

Network Delay Tomography Using Flexicast Experiments

Earl Lawrence

Los Alamos National Laboratory, Los Alamos, NM 87545, USA

George Michailidis

University of Michigan, Ann Arbor MI, USA

Vijayan N. Nair

University of Michigan, Ann Arbor MI, USA

Summary. Estimating and monitoring the quality of service of computer and communications networks is a problem of considerable interest. This paper focuses on estimating link-level delay distributions from end-to-end path-level data collected using active probing experiments. This is an interesting large-scale statistical inverse (deconvolution) problem. We describe a flexible class of probing experiments (flexicast) for data collection and develop conditions under which the link-level delay distributions are identifiable. Maximum likelihood estimation using the EM algorithm is studied. It does not scale well for large trees, so a faster algorithm based on solving for local MLEs and combining their information is proposed. The usefulness of the methods is illustrated on real Voice-over-IP data collected from the University of North Carolina campus network.

Keywords: Deconvolution, EM algorithm, Internet, Inverse problem, Tree-structured graphs

1. Introduction

Computer and communications networks form the backbone of our information society. Over the last decade, these networks have experienced an exponential growth in terms of the number of users, the amount of traffic, and the complexity of applications. It is important for network engineers and internet service providers to be able to estimate and monitor the quality of service (QoS) parameters such as link delays, dropped packet rates, and available bandwidth. However, the decentralized nature of modern computer and communications networks has made it difficult to assess performance. Traditional methods based on queueing analysis focus on the behavior of one or a small number of routers and are inadequate in characterizing the complexities of such networks. This has led to the emergence of *network tomography* – an area that uses *active* and *passive* traffic measurement schemes to quantify the performance and QoS of networks. A good review of the area and challenges can be found in [1].

The term network tomography was introduced in [2], which dealt with the estimation of origin-destination traffic intensities based upon the total measured intensities along individual links. See [3], [4], and [5] for related work on this problem. Active tomography deals with estimating link-level characteristics, such as loss rates and delay distributions, by actively *probing* the network. This involves sending probe packets from one or more sender nodes to a set of receiver nodes and measuring the end-to-end characteristics. The goal then is to estimate (“recover”) the link-level information from the end-to-end path level data. This paper focuses on delay distributions.

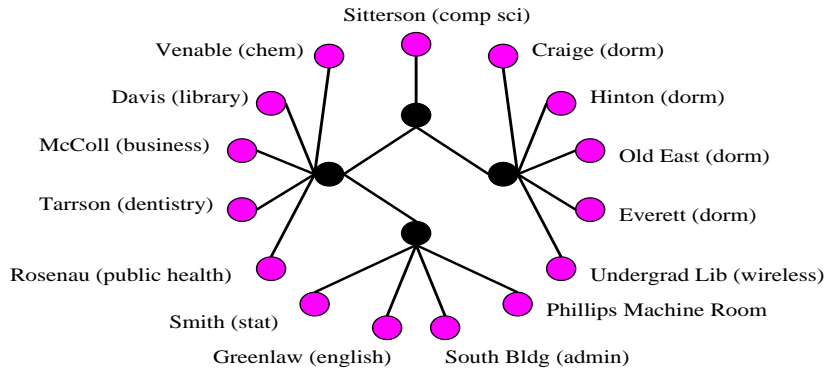


Fig. 1. Left panel: Schematic of the UNC network; Right panel: Logical topology of the UNC network.

As an example, consider the emerging application of Internet or Voice over IP (VoIP) telephony. VoIP is a technology that turns voice signals into packets and transmits them over the Internet to the intended receivers. The main difference from classical telephony is that the call does not use a dedicated connection with reserved bandwidth, but instead packets carrying the voice data are multiplexed in the network with other traffic. For this application, the QoS requirements in terms of packet losses and delays are significantly more stringent than non-real time applications such as e-mail. The University of North Carolina (UNC) has been in the planning phase of deploying VoIP telephony and wanted to assess its campus network to determine if it is capable of supporting the technology. To do this, monitoring equipment and software capable of placing such phone calls was installed throughout the campus network. The software allowed the emulation of VoIP calls between the monitoring devices. It can then synchronize the clocks and obtain very accurate packet loss and delay measurements along the network paths.

Fifteen monitoring devices from Avaya Labs were deployed in a variety of buildings and on a range of different capacity links through the UNC network. The locations included dorms, libraries, and various academic buildings. The links included large capacity gigabit links, smaller 100 megabit links, and one wireless link. Monitoring VoIP transmissions between these buildings allowed us to examine traffic influenced by the physical conditions of the link and the demands of various groups of users. The left panel of Figure 1 gives the physical connectivity of the UNC network. Each of the nodes on the circle has a basic machine that can place a VoIP phone call to any of the other endpoints. The three nodes in the middle are part of the core (main routers) of the network. One of these internal nodes, the upper router linked to Sitterson Hall, also connects to the gateway that exchanges traffic with the rest of the Internet. The measured data consist of end-to-end delays and losses. We will use data collected from this study to illustrate our methodology in Section 8.

Although the physical structure of a network can be arbitrary (left panel of Figure 1), the logical topology for the probing experiment can be represented more simply (right panel of Figure 1). We will follow the common practice in the literature and focus attention on logical topologies that can be described by trees: *acyclic graphs with one vertex designated as the root* (see right panel of Figure 2). Formally, let $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ be a tree with node set \mathcal{V} and link set \mathcal{E} . The nodes follow a canonical numbering scheme, starting from the root node 0. All links will be named after the node at their terminus, so in Figure 2, link 1 refers

to the link connecting nodes 0 and 1. The parent of node $k \in \mathcal{V}$ will be written $f(k)$. In a tree, all nodes have a parent except for the root node. Define $f^i(k)$ recursively as follows: $f^i(k) = f(f^{i-1}(k))$, where $f^1(k) = f(k)$. Node k is said to be in layer L if $f^L(k) = 0$. Let $\mathcal{D}(k)$ denote the children of node k , which is the collection of nodes whose parent is k . Let \mathcal{R} denote the set of leaf or receiver nodes, i.e., nodes with no children. Finally, an internal node is one with both a parent and a set of children. The left panel of Figure 2 shows a simple binary tree with three layers. We will use it later in the paper to illustrate some of the techniques.

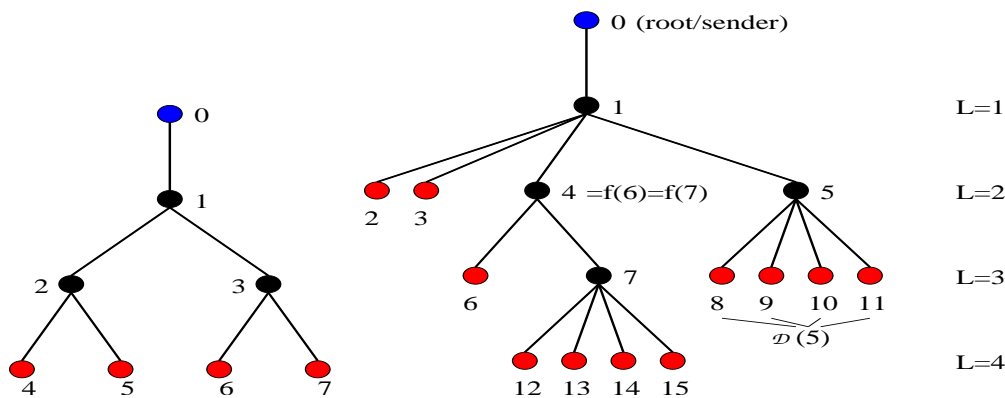


Fig. 2. Left panel: Three-layer, binary, symmetric tree; Right panel: A general tree with notation.

Packets can be sent from a source to a destination using two basic transmission protocols: a unicast scheme that sends a packet from a source to one receiver at a time, and a multicast scheme that sends a packet simultaneously to a set of specified receivers. The situation where all the receiver nodes in a network are probed simultaneously using a single multicast scheme is called a multicast experiment in the literature. We will refer to it instead as an omnicastr probing experiment. The flexicast experiments in Section 3 are also based on multicast transmission, although they do not involve sending the packet to all receivers in the network.

The estimation of loss rates based on active probing has been studied by several authors (see, for example, [?], [?], [?]) and references therein). Link delay tomography was first studied in [?] in which the authors developed a heuristic estimator for omnicastr experiments based on solving polynomial equations, which can be very inefficient. [?] developed a pseudo-likelihood approach by considering all possible pairwise results from each individual full omnicastr result. We will compare these two techniques with our methods later in the paper. [?] presented an estimator that models link delay using a point mass at zero and a finite mixture of Gaussian distributions.

The rest of the paper is organized as follows. Section 2 describes the models and assumptions. The following section describes the framework for probing studies, introduces the flexicast probing experiments, and studies conditions under which the link-level delay parameters are estimable from end-to-end path-level data. Section 4 deals with maximum likelihood estimation using the EM algorithm and develops its computational and theoretical aspects. Alternative, heuristic algorithms that are faster and scale up well to large-scale networks are discussed in Section 5. An extensive numerical investigation of the procedures is also presented using the *ns-2* network simulation software. Finally, the methods are

illustrated on real data collected from the UNC campus network.

2. Models and Assumptions

Following ?) and ?), we will consider nonparametric estimation of the delay distributions using a discrete distribution with a fixed, universal bin size. Specifically, let X_k be the delay accumulated on link k , taking values in the set $\{0, q, \dots, bq\}$. Here q is the bin size and b is the maximum discrete delay (assumed to be common for all the links). Although this framework seems restrictive, it is useful for a number of reasons. First, experience with real network traffic data shows that behavior tends to vary with the particular network being studied, so selecting a particular parametric family for modeling the link delay distribution is difficult. Moreover, the delay data typically exhibit bursty behavior, in which case the tails of the distribution are also of considerable interest. The discrete model makes no assumptions about where the mass is located, so the tails of heavy-tailed distributions can be estimated provided we have a sufficient number of probes. The bin size can be chosen adaptively after the data are collected. Smaller bin sizes can be used to estimate detailed information about the distribution. Large bin sizes can be used to obtain tail information. Examples of this will be discussed in the data analysis section.

Throughout the paper, we will ignore losses or infinite delays. One can always estimate the loss rates and the finite delay distributions separately and combine the results to estimate the overall network behavior. In addition, we make the assumption (common in the network tomography literature) that the packet delays are temporally independent and that the delay of a packet on a link is independent of the delay on the other links in the path. The assumption of temporal independence is reasonable as long as the interval between probes is large enough. Temporal stationarity is reasonable as long as the probing period is short enough to avoid major network changes. The adequacy of the spatial assumption will depend on the particular network being studied and whether there are other physical links connecting the nodes.

The data are collected by recording the total delay that a packet experiences as it travels from the root node to the receiver nodes. For example, in the right panel of Figure 2, probe packets would be sent from node 0 to various collections of nodes 2, 3, 6, 8, 9, 10, 11, 12, 13, 14, and 15 and the delays experienced along their corresponding paths would be recorded. Physically, each end-to-end delay is the sum of the individual link delays along the path. If a scheme has a collection of k receivers, a single observation would be a k -tuple of delays.

Let $\mathcal{P}_{0,k}$ denote the path from node 0 to node k , and let $Y_k = \sum_{i \in \mathcal{P}_{0,k}} X_i$ be the cumulative delay accumulated from the root node to node k . For example, $Y_3 = X_1 + X_3$ in the right panel of Figure 2. The measurements obtained from a delay tomography experiment consist of cumulative delays Y_r , $r \in \mathcal{R}$. Let $\alpha_k(i) = \mathbb{P}\{X_k = iq\}$, $i = 1, \dots, b$. Our objective is to estimate this set of values for $k \in \mathcal{E}$ and i in $\{0, 1, \dots, b\}$ using the Y_r measurements.

In the following, we will use the notation $\vec{\alpha}_k = [\alpha_k(0), \alpha_k(1), \dots, \alpha_k(b)]'$ and $\vec{\alpha} = [\vec{\alpha}'_0, \vec{\alpha}'_1, \dots, \vec{\alpha}'_{|\mathcal{E}|}]'$. Let $\pi_{j,k}(i)$ be the probability that the delay accumulated on path $\mathcal{P}_{j,k}$ is equal to i units. This is a function of $\vec{\alpha}$.

3. Flexicast Probing Experiments

While the omnicast probing experiment is easy to implement, it suffers from several disadvantages. Suppose there are R receivers and the total number of possible bins associated with the path-level delay distribution of the i th receiver is B_i . Then, the total number of possible outcomes is of order $\prod_{i=1}^R B_i$ which can be a huge number. As we will see later, the computational efficiency of most estimation methods depends on this number. As an example, let each link-level delay distribution have the same number of bins: $b = 4$. Further, suppose we have a symmetric binary tree with 5 layers. Then, there are $R = 16$ receivers and $B_i = 19$ for the path-level delay distributions of all the receivers, so the approximate number of possible outcomes is 2.8×10^{20} .

A second, perhaps more important, issue is the lack of flexibility as it involves probing the entire network each time. In practice, we are interested in monitoring the network over a period of time, and so we need a flexible scheme that allows us to probe different parts of the network with different degrees of intensity depending on where there are bottlenecks or quality of service problems. We consider a flexible class of probing experiments called flexicast experiments in order to address these problems in the context of delay tomography.

Define first a k -cast scheme as one that sends probes from a receiver to specified set of k receiver nodes. For example, for the tree on the left panel of Figure 2, $\langle 2, 3 \rangle$ is a two-cast (or bicast) scheme while $\langle 6, 12, 13, 14, 15 \rangle$ is a five-cast scheme. A flexicast experiment \mathcal{C} is a combination of k -cast schemes \mathcal{C}_j , $j = 1, \dots, M$, with possibly different values of k , that allows one to estimate all of the link-level parameters. Returning to Figure 2, one flexicast experiment made up of a collection of bicast and unicast schemes is $\mathcal{C} = \{\langle 2, 3 \rangle, \langle 6, 12 \rangle, \langle 13, 14 \rangle, \langle 8, 9 \rangle, \langle 10 \rangle, \langle 11 \rangle, \langle 15 \rangle\}$. Another that uses larger k -casts is $\mathcal{C} = \{\langle 2, 3 \rangle, \langle 6, 12, 13, 14, 15 \rangle, \langle 8, 9, 10, 11 \rangle\}$.

A natural question is: When will a flexicast experiment lead to identifiability? We provide below a necessary and sufficient condition. The proof is based on the idea that an individual k -cast probe can identify all of the path distributions between branching nodes on its subtree. It suffices for the collection to be rich enough in terms of subtrees that the individual links can be expressed as functions of paths from different schemes. We formalize this intuition in the following Proposition. The proof is deferred to the Appendix.

PROPOSITION 1. *Let \mathcal{T} be a general tree network, and suppose its link delay distributions are discrete. Let \mathcal{C} be a collection of k -cast schemes \mathcal{C}_j , $j = 1, \dots, M$. The link-level delay distributions are identifiable if and only if: (a) For every internal node $s \in \mathcal{T} \setminus \{0, \mathcal{R}\}$, there is at least one k -cast scheme $\mathcal{C}_j \in \mathcal{C}$, with $k > 1$, such that s is a branching node for \mathcal{C}_j , and (b) every receiver $r \in \mathcal{R}$ is covered by at least one $\mathcal{C}_j \in \mathcal{C}$.*

Remark 1: We have restricted attention to discrete distributions as they are the focus of the present paper, but the result holds more generally. First, the result can be shown to hold as long as the distribution has at least one point mass. It will also hold for purely continuous distributions under some conditions (such as higher order moments depending on the mean). It does not, however, hold for arbitrary continuous distributions. This can be seen easily using a two-layer tree (the top two layers of the tree in the left panel of Figure 2) with a source node 0, internal node 1 and receiver nodes 2 and 3. Let the link-level delay random variables be X_1 , X_2 , and X_3 and the path-level delay random variables at receiver nodes 2 and 3 be $Y_2 = X_1 + X_2$ and $Y_3 = X_1 + X_3$. Suppose the X_i 's are independent $N(\mu_i, 1)$. Then, we cannot recover the μ_i 's from the joint distribution of Y_2 and Y_3 .

4. Maximum Likelihood Estimation

Active delay tomography is an instance of a large-scale inverse problem. For example, consider the omnicast problem for the topology given in the right panel of Figure 2. Here we must use the 11-dimensional end-to-end measurements to estimate the 15 link delay distributions. The key is the dependence among the 11-dimensional data induced by the simultaneous probing. This dependence gives us additional information that allows us to deconvolve the path-level delay into link-level information. We consider maximum likelihood estimation first.

We need some additional notation. Let $\mathcal{T}^{\mathcal{C}_j}$ be the subtree probed by scheme $\mathcal{C}_j \in \mathcal{C}$, with node set $\mathcal{V}^{\mathcal{C}_j}$ and link set $\mathcal{E}^{\mathcal{C}_j}$. Let $\mathcal{X}^{\mathcal{C}_j} = \{0, 1, \dots, b\}^{|\mathcal{E}^{\mathcal{C}_j}|}$ be the set of all possible link delay combinations that could arise from this scheme. Each $x \in \mathcal{X}^{\mathcal{C}_j}$ is an $|\mathcal{E}^{\mathcal{C}_j}|$ -tuple giving a possible link-delay combination. Let the function $y(x, \mathcal{T}^{\mathcal{C}_j})$ give the end-to-end delay arising in scheme \mathcal{C}_j from link outcome $x \in \mathcal{X}^{\mathcal{C}_j}$. Define the set of all possible end-to-end delays as $\mathcal{Y}^{\mathcal{C}_j} = \{y(x, \mathcal{T}^{\mathcal{C}_j}) | x \in \mathcal{X}^{\mathcal{C}_j}\}$. Let $\gamma_{\mathcal{C}_j}(y) = P\{Y^{\mathcal{C}_j} = y\}$, the probabilities for the end-to-end experimental outcomes.

We illustrate this notation using the right panel of Figure 2. Suppose we probe the pair $\langle 2, 3 \rangle$. Let $b = 1$ so $X_k \in \{0, 1\}$. The link set is $\mathcal{E}^{\langle 2, 3 \rangle} = \{1, 2, 3\}$. Assume that only a single probe packet is used for this scheme, and it experiences link delays of 0, 1, and 1 on each link, respectively. We then have $x = (0, 1, 1)$ and $y = (1, 1)$. The probability of this link delay set is $P\{X^{\langle 2, 3 \rangle} = (0, 1, 1)\} = \alpha_1(0)\alpha_2(1)\alpha_3(1)$. The probability of this end-to-end outcome is $P\{Y^{\langle 2, 3 \rangle} = (1, 1)\} = \alpha_1(0)\alpha_2(1)\alpha_3(1) + \alpha_1(1)\alpha_2(0)\alpha_3(1)$ which is the sum of the probabilities for the link outcomes which can give rise to this end-to-end outcome.

4.1. EM Algorithm

The discrete nonparametric distribution framework results in multinomial outcomes for path-level data. Specifically, the observations consist of the number of times that one observes each individual outcome \vec{y} from the set of outcomes $\mathcal{Y}^{\mathcal{C}_j}$ for a given scheme. Denote these counts as $N_{\vec{y}}^{\mathcal{C}_j}$. Consider the likelihood equation

$$l(\vec{\alpha}; \mathbf{Y}) = \sum_{\mathcal{C}_j \in \mathcal{C}} \sum_{\vec{y} \in \mathcal{Y}^{\mathcal{C}_j}} N_{\vec{y}}^{\mathcal{C}_j} \log[\gamma_{\mathcal{C}_j}(\vec{y}; \vec{\alpha})]. \quad (1)$$

This expression is difficult to maximize directly. However, it is a classical example of a missing data problem: if the counts for the unobserved link delays were known, the maximization would be fairly straightforward as the link outcomes are also simple multinomial experiments. The EM algorithm is a natural candidate for computing the maximum likelihood estimates and has been used in the area. For exponential families, we need to impute just the sufficient statistics for each link: the counts for the number of times that X_k took on each value.

The E-step can be broken into two parts. Assume that we have some parameter vector $\vec{\alpha}^{(q-1)}$. First, we compute the expected number of times each link delay vector, \vec{x} , occurred as

$$N_{\vec{x}}^{\mathcal{C}_j (q)} = \frac{P\{\vec{X}^{\mathcal{C}_j} = \vec{x}\}^{(q)}}{P\{\vec{Y}^{\mathcal{C}_j} = \vec{y}(\vec{x})\}^{(q)}} N_{\vec{y}}^{\mathcal{C}_j}. \quad (2)$$

Then, we use these values to compute the number of times that probes on link k had a

delay of i units as

$$M_{k,i}^{(q)} = \sum_{\mathcal{C}_j \in \mathcal{C}: k \in \mathcal{T}^{\mathcal{C}_j}} \sum_{\vec{x} \in \mathcal{X}^{\mathcal{C}_j}: x_k = i} N_{\vec{x}}^{\mathcal{C}_j (q)}. \quad (3)$$

We also need to keep track of m_k which is the total number of probes that crossed link k .

The M-step is quite simple once the sufficient statistics have been imputed.

$$\alpha_k(i)^{(q)} = \frac{1}{m_k} M_{k,i}^{(q)} \quad (4)$$

4.2. Partitioning

The computationally challenging aspect in our setting is to partition the observed end-to-end delays into the set of possible link delay combinations. These details are given next.

Consider the left panel of Figure 3. Suppose that this is the probing tree for a five-cast experiment and that the maximum link delay is $b = 2$. Suppose further that a single probe results in the observed delay vector $Y = [2 \ 3 \ 3 \ 4 \ 3]$. We need to systematically partition this end-to-end delay vector into the complete list of all possible link delay vectors that give this result. We move top to bottom, identifying possible delays for links starting at the top of the tree and moving downward. We begin by listing possible link delays for the first link, between nodes 0 and 1, and leaving the rest of the delays as path delays. This amounts to imagining the tree takes the form of the *shrub* shown in the right panel of Figure 3 with each branch of the shrub having a maximum delay determined by b and the number of links from node 1 to each of the receivers.

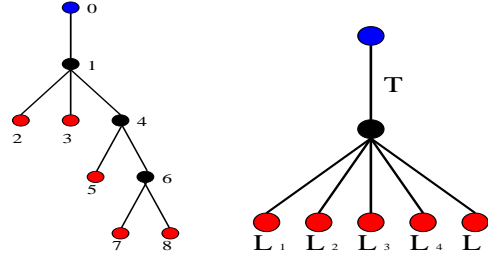


Fig. 3. Partitioning example: probing subtree (left panel) and its corresponding shrub (right panel).

To get the lower bound of the possible delays for the first link, consider the minimum delay possible on this link that will give the observed values. The lower bound is the maximum of a set containing zero and each observed value minus the maximum delay that could be obtained on its branch, $Y_r - g_r b$ where g_r is the number of links hidden in the branch of the shrub connecting receiver r to the splitting node. For this example, the value is one. The upper bound is simply the minimum of b and the set of observed values. Here the value is two. This allows us to expand the observed delay into the set of link 1 delays and the remaining delays:

$$[2 \ 3 \ 3 \ 4 \ 3] \rightarrow \begin{bmatrix} 1 & 1 & 2 & 2 & 3 & 2 \\ 2 & 0 & 1 & 1 & 2 & 1 \end{bmatrix}. \quad (5)$$

We have now isolated the delays that could occur on the first link. We have also isolated the delays that could occur on the second and third links. Now we need to expand the

triplets [2 3 2] and [1 2 1]. This is done exactly as before by considering only the portion of the tree rooted on the link between nodes 1 and 4. Each triplet is an end-to-end observation from this portion of the tree. For each set, we imagine the tree to be a three-branch shrub and expand the observation on the possible values that could occur on link 4:

$$\begin{bmatrix} 1 & 1 & 2 & 2 & 3 & 2 \\ 2 & 0 & 1 & 1 & 2 & 1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 1 & 2 & 0 & 2 & 3 & 2 \\ 1 & 1 & 2 & 1 & 1 & 2 & 1 \\ 1 & 1 & 2 & 2 & 0 & 1 & 0 \\ 2 & 0 & 1 & 0 & 1 & 2 & 1 \\ 2 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix}. \quad (6)$$

Partitioning the second shrub gave us the range of values for links 4 and 5 leaving all the pairs comprising the last two columns. Each pair can be partitioned into the three component parts of this remaining shrub to give us the full partition for the observed delay on this tree:

$$\begin{bmatrix} 1 & 1 & 2 & 0 & 2 & 3 & 2 \\ 1 & 1 & 2 & 1 & 1 & 2 & 1 \\ 1 & 1 & 2 & 2 & 0 & 1 & 0 \\ 2 & 0 & 1 & 0 & 1 & 2 & 1 \\ 2 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 1 & 2 & 0 & 2 & 1 & 2 & 1 \\ 1 & 1 & 2 & 0 & 2 & 2 & 1 & 0 \\ 1 & 1 & 2 & 1 & 1 & 0 & 2 & 1 \\ 1 & 1 & 2 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 2 & 2 & 0 & 0 & 1 & 0 \\ 2 & 0 & 1 & 0 & 1 & 0 & 2 & 1 \\ 2 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 2 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}. \quad (7)$$

Formally, let a shrub be any tree graph with a single internal node which has one or more children that are all receivers. Partitioning the shrub is all that is required to partition any tree or subtree. By moving downward from the top and expanding one link at a time, we can ignore any structure below the link of interest. The tree becomes a shrub by considering each receiver descended from the link of interest to be on a separate branch. After expanding the desired link, the remaining delay can again be partitioned using the shrub algorithm. A single recursive function is all that is required: it would partition the tree as if it were a shrub and then call itself to partition the sub-shrubs.

The algorithm for the general shrub with r receivers is quite simple. Let $Y = [Y_1, \dots, Y_r]$ be the delay observed on the shrub. Further, let t be the maximum delay that can be observed on the trunk and let l_i be the maximum delay that can be observed on leaf i . We have

$$a = \max \left\{ 0, \max_i \{Y_i - l_i\} \right\} \text{ and} \quad (8)$$

$$z = \min \left\{ t, \min_i \{Y_i\} \right\} \quad (9)$$

as the minimum and maximum possible values for the trunk delay. Thus, in one for-loop it is easy to make the expansion:

$$Y = [Y_1, \dots, Y_r] \rightarrow X = \begin{bmatrix} a & Y_1 - a & \dots & Y_r - a \\ a + 1 & Y_1 - (a + 1) & \dots & Y_r - (a + 1) \\ \vdots & \vdots & \ddots & \vdots \\ z & Y_1 - z & \dots & Y_r - z \end{bmatrix}. \quad (10)$$

4.3. EM Complexity

To study the complexity of a single EM iteration, consider first a specific k -cast scheme. There are $b^{|\mathcal{T}^{C_j}|}$ link delay outcomes for this probe. For each of these, there are $|\mathcal{T}^{C_j}|$ multiplications to compute the probability of the link delay outcome. For each outcome, there is also a single addition to tally up the end-to-end probabilities and a single division to compute the conditional probability of each outcome given the end-to-end outcome. Finally, there are $|\mathcal{T}^{C_j}|$ additions to tally up the sufficient statistics. Overall, this gives us $\mathcal{O}\{b^{|\mathcal{T}^{C_j}|}\}$ operations. The largest subtree sets the complexity for the E-step at $\mathcal{O}\{b^{|\mathcal{T}^{C_{large}}|}\}$ where C_{large} is the largest experiment. The M-step is trivial consisting of $|\mathcal{E}b|$ divisions.

There are a few things to note. First, mixtures of bicast ($k = 2$) and unicast schemes offer the best complexity while meeting the identifiability conditions. Additionally, they scale better than k -casts with larger values of k . In particular, an omnicast scheme does not scale well.

If the tree grows in size but not in depth, then the bicast schemes should scale well because this will simply result in more bicast schemes rather than more complicated bicast schemes. This property does not hold for omnicast schemes which have complexity $\mathcal{O}\{b^{|\mathcal{E}|}\}$.

Also of note, the complexity stated here is the extreme worst case based on observing every possible delay combination from every probing experiment. In practice, both the partitioning and estimation only consider the observed delays which will significantly reduce the average-case complexity.

Unfortunately, the EM-algorithm does not scale well as the tree gets deeper for any k -cast scheme. For such topologies, we explore alternative fast estimators in a later section. Note, however, that the EM algorithm can be made computationally more efficient through parallelization. Notice that the E-step involves computing a sum that ranges first over the schemes and then over the outcomes for that experiment. This sum can be broken down into component pieces which can be computed simultaneously and combined.

4.4. Numerical Investigation of EM-algorithm

This section studies effects of the tree size and the number of bins (in the discrete delay distribution) on the convergence of the EM algorithm. These two factors determine the number of model parameters. In practice, one has flexibility over the number of bins but has limited control over the tree size. For example, in a monitoring situation, a coarse distribution can be estimated quickly and still provide enough information to detect anomalies in the network. On the other hand, one can only obtain a smaller tree by lumping several links together and eliminating some of the receivers.

Consider first the effect of the tree size, in terms of both number of links and layers. We start with a two-layer symmetric binary tree so that there is a total of three links. Then, we add two children at a time. So the five-link tree corresponds to adding two children to one of the receiver nodes. The seven-link tree adds two children to the other receiver nodes (so that this is just a three-layer symmetric binary tree). We proceed in this manner until we get the four-layer symmetric binary tree with 15 links (see x -axis of left panel of Figure 4). The remaining model components are held fixed. In particular, each link has a five-bin uniform delay distribution chosen because it provides maximum entropy, thus it is the most difficult to resolve. We use a flexicast experiment that is a minimal set of bicast schemes that satisfy the identifiability conditions. Some additional studies indicated that convergence for the EM algorithm does not seem to depend greatly on the number

of probes used. Hence, in the investigations here, each bicast scheme had 1000 probes for each links in its subtree. For each tree size, 50 sets of data were generated and used for estimation. The convergence criteria was a change in the log-likelihood of less than 10^{-4} . The left panel of Figure 4 shows the number of iterations required for convergence for each data set. The average number of iterations for each size is plotted with standard deviation error bars. This suggests that the average number of iterations seems to be increasing at a faster than linear rate (perhaps exponential) with the number of links. This is a further indication that the EM algorithm does not scale well to larger networks.

Next, consider the effect of the number of bins in each link with uniform delay distributions. The number of bins on each link was varied from two to 15. We considered both two-layer and three-layer binary symmetric trees with three and seven links respectively. Again, a minimal bicast experiment with 1000 probes per scheme was used. The results for both trees are shown in the right panel of Figure 4. In this scenario, the average number of iterations seems to grow approximately linearly with the number of bins on each link. This is an important observation that we will exploit later in developing a faster algorithm.

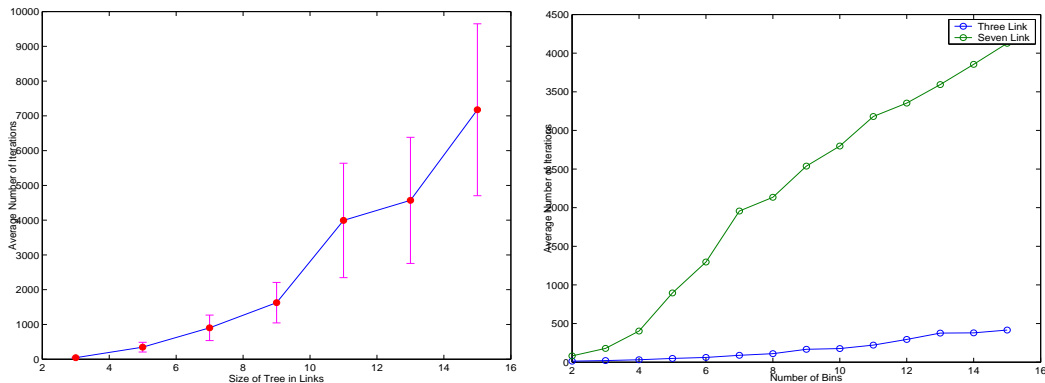


Fig. 4. Left panel: Number of iterations versus tree size (number of links); Right panel: Number of iterations versus bin size for 2- and 3-layer trees

4.5. Asymptotic Properties of the MLE

Given the multinomial nature of the underlying k -cast schemes, the asymptotic properties of the MLE mostly follow from general principles. The difference arises because the flexicast experiments mean that individual probes are not *i.i.d.* The proposition below establishes that Fisher information matrix is positive definite at the true value $\vec{\alpha}_0$. Thus, the likelihood has a unique maximum in a local neighborhood of the true value $\vec{\alpha}_0$.

PROPOSITION 2. *The Fisher information matrix $\mathcal{I}(\vec{\alpha}_0)$ for the maximum likelihood estimator based on end-to-end quantized measurements from a flexicast experiment \mathcal{C} is finite and positive definite.*

The proof can be found in (?). The main idea is to treat the Fisher information as the covariance of the score function and then consider various hypothetical data sets to establish that it must be positive definite.

PROPOSITION 3. Let $\frac{n^{c_j}}{n} \rightarrow \lambda^{c_j}$ as $n \rightarrow \infty$ with $0 < \lambda^{c_j} < 1$ for $j = 1, \dots, M$. Then, $\vec{\alpha}_{\text{MLE}} \rightarrow \vec{\alpha}_0$, a.s., and $\sqrt{\mathbf{n}}(\vec{\alpha}_{\text{MLE}} - \vec{\alpha}_0) \Rightarrow Z$, where $Z \sim N\{\vec{0}, \mathcal{I}^{-1}(\vec{\alpha})\}$.

Proposition 3 follows from Proposition 2 using standard arguments.

5. Faster, Heuristic Algorithms

5.1. Grafting

We propose a method called grafting which computes the “local MLE” on each subtree and uses peeling to combine the results. Shown in Figure 5, peeling is the process of using a known path distribution and a known distribution from one end of that path to solve for the distribution on the other end (see the proof of Proposition 1). In essence, grafting treats each k -cast scheme as an omnicast experiment on the probing subtree. It uses EM to solve for the MLE of the logical links on this subtree and then uses peeling to get estimates for individual links. For collections of bicast and unicast scheme, this technique scales very well because the EM algorithm is applied to a series of three-link, two-layer trees. Thus the complexity is a cubic polynomial in the number of bins, a vast improvement over the standard MLE complexity. Based on the investigations discussed previously, increasing the number of bins on the links increases the average iterations approximately linearly while adding links increases the required iterations exponentially. This local scheme takes advantage of this fact by trading links for bins.

Fig. 5. Peeling is the process of using a known path distribution and the known distribution for one end of the path to solve for the distribution of the other end.

We will explain the details using just a flexicast experiment with bicast and unicast scheme. First, consider a bicast scheme and the corresponding subtree. Let the trunk have t links and the two branches have l_1 and l_2 links respectively. The subtree has just three logical links with varying numbers of bins on each: the trunk has $tb + 1$ bins and the branches have $l_1b + 1$ and $l_2b + 1$ bins. We apply the EM algorithm to this logical subtree and solve for its MLE. This is done for all of the bicast schemes. This gives the estimates for the trunks and branches of all the bicast subtrees.

Individual links can be now be obtained in one of several ways. Consider first the simple peeling from the proof of Proposition 1. This is straightforward and non-iterative although not very statistically efficient as only a some of the bin probabilities from each known distribution are used in computing the unknown distribution. At least one pair must split at node 1, so at least one of the local MLEs must give us an estimate for link 1. Now, at least one scheme gives us the local MLE for the path from the root node to every child of node 1. So the individual links up to these points can be identified through peeling. This process continues down the tree identifying each link. The receivers covered by bicast

experiments can be identified as the branches in a subtree or by peeling from the branches. The receivers covered by only unicast experiments can also be identified by peeling.

We now propose a more sophisticated peeling mechanism. This is a fixed-point type algorithm that arises from postulating an EM algorithm for imaginary data. Imagine that we send n probes across the path. Form data by setting $n_d = n\pi_{0,2}(d)$. The data are counts of the number of times delay d was observed on the path for all possible d . In the E step, we want to compute M_i , the expected number of times that delay i was seen on the unknown link. After the q -th iteration, this is given by:

$$M_i^{(q+1)} = \sum_{j=0}^b \frac{\alpha_2^{(q)}(i)\alpha_1(j)}{\pi_{0,2}^{(q)}(i+j)} n_{i+j}, \quad (11)$$

where $\pi_{0,2}^{(q)}$ is updated with each update of $\bar{\alpha}_2^{(q)}$. Note that this is not the quantity used to generate the data. Since we obtain our estimates by dividing M_i by n , we get the following equation:

$$\alpha_2(i)^{(q+1)} = \sum_{j=0}^b \frac{\alpha_2^{(q)}(i)\alpha_1(j)}{\pi_{0,2}^{(q)}(i+j)} \pi_{0,2}(i+j). \quad (12)$$

This equation is no longer based on our imaginary data. It is run until $\bar{\alpha}_2$ approaches its fixed point. Note that this fixed-point algorithm is simply an EM algorithm for computing one link-level distribution from the path-level and other link-level distributions and meets standard conditions for convergence. Unlike the simple method, this peeling function uses all of the information from the two known distributions.

The peeling method can lead to multiple estimates for some links. The easiest way to address this problem is to combine them, using either a simple average or a weighted average. For the latter, if we have two estimates of $\bar{\alpha}_1$ from experiments \mathcal{C}_1 and \mathcal{C}_2 , we can combine them as follows to get

$$\tilde{\bar{\alpha}}_1 = \frac{n^{\mathcal{C}_1}\bar{\alpha}_1^{\mathcal{C}_1} + n^{\mathcal{C}_2}\bar{\alpha}_1^{\mathcal{C}_2}}{n^{\mathcal{C}_1} + n^{\mathcal{C}_2}}. \quad (13)$$

It can be shown that the grafting algorithm yields estimators that are consistent and asymptotically normal. The computation of the asymptotic variance is complicated and the simplest way to compute the variance is to use bootstrap or other resampling methods.

5.2. A Comparison of the Various Estimators

5.2.1. Other Estimators in the Literature

Two other estimation procedures have been proposed in the literature for the delay tomography problem. Both are based on omnicastr probing and rely on the same modeling assumptions as those presented here: discrete delay with temporal and spatial independence. The first, discussed by (?), depends on solving polynomials. At some link k , the estimator uses the data from the subtree rooted at the link to create a polynomial for each unit of delay, i . The degree of the polynomial is $|\mathcal{D}(k)|$. The second root of this polynomial give us the cumulative probability of delay i on link k . The principal drawback of this estimator is that it does not use all of the information available. End-to-end delays that are larger than the largest allowable link delay are ignored. Additionally, the nature of the estimator allows inappropriate results from the polynomial solution such as negative values or values greater than one.

The estimator by ?) is based on a pseudo-likelihood approach. The complexity of the omnicast experiment is reduced by looking at just bicast projections – all pairwise combinations – and using a pseudo-likelihood that treats them as independent. For example, the network in the left panel of Figure 2 has 11 receivers, so the omnicast experiment result in 11-dimensional delay observations. There are 55 possible pairs of receivers, so the pseudo-likelihood scheme treats all the possible pairs of delays as 55 independent bicast observations. The motivating idea is that processing the data as pairs is computationally much more efficient than processing the omnicast data. This can be justified if the gain in computational speed offsets the loss in statistical efficiency.

5.2.2. Computational Efficiency

Computational speed of the estimators is clearly a major consideration in real applications. Network monitoring requires the ability to solve the inverse estimation problem very quickly. In this section, we investigate the computational efficiencies of the various methods for several different tree structures. Specifically, we compare the efficiencies of the MLEs based on omnicast probing, all-pairs bicast experiment, and $min+1$ bicast experiment.

Recall that a minimal flexicast refers to a combination of bicast and unicast probing schemes that satisfy the identifiability condition. For a symmetric binary tree, this consists of just bcasts. To see this, consider the three-layer, binary, symmetric tree in the right panel of Figure 2. The minimal bicast experiment is $\{\langle 4, 5 \rangle, \langle 5, 6 \rangle, \langle 6, 7 \rangle\}$. This experiment is unbalanced as receiver links 4 and 7 are only covered once while links 5 and 6 are covered twice. A more balanced approach is obtained by adding pair $\langle 4, 7 \rangle$. This ensures that each link on a particular layer of a binary, symmetric tree is covered by the same number of experiments. We refer to such experiments as $min+1$ flexicast experiments.

In addition to the MLEs, we also consider the pseudo-likelihood estimator, the polynomial estimator from ?), and grafting for all-pairs bicast and $min+1$ bicast. All of the estimators were implemented using Matlab with the combinatorial partitioning components of the likelihood-based methods written in C. The link delays follow a five-bin truncated geometric distribution. The parameter of the distributions were varied in a manner to keep the situations realistic: the interior links have high probability of zero as compared with edge links to simulate the difference between internal links with large bandwidth and smaller local links. The efficiency comparisons were based on 100 simulated data sets and are shown in Table 1.

The polynomial estimator is, of course, the fastest. This is partially driven by the fact that it is solving quadratic equations in this example (binary tree) and the formulas for the estimates are obtained explicitly. The effect of having a large number of children on the polynomial estimator will be investigated later. We will also see later that this algorithm can be considerably inefficient in a statistical sense. As to be expected, the PLE is faster than the MLE based on the full EM; however, it does not gain as much over the MLE as does the pure bicast algorithm. The all-pairs bicast is more than twice as fast as the likelihood-based multicast estimators while the PLE does not seem to benefit from an order of magnitude gain.

Unbalanced Tree: Here we consider the tree structure shown in the right panel of Figure 2. We investigate only the pseudo-likelihood estimator, the minimum-pairs bicast MLE, the all-pairs grafting procedure, and the minimum pairs grafting procedure. The polynomial estimator was dropped due to the difficulty in implementing it for general trees. It is studied

Table 1. Three-Layer Tree Comparison: Average computing time (in seconds).

Estimator	MLE	PLE	Polynomial	All-Pairs MLE	Min+1 MLE	All-Pairs Graft	Min+1 Graft
Time	46.93s	32.68s	3.96s	17.19s	11.11s	9.11s	5.59s

Table 2. Computational speeds of estimators applied to data from the right panel of Figure 2

Estimator	PLE	Min Pairs MLE	All Pairs Graft	Min Pairs Graft
Avg. Time	401.27s	10.12s	41.61s	4.99s

below however for another situation. The link delay distributions were again chosen to be a five-bin, truncated geometric. Table 2 shows the results of the comparisons. The pseudo-likelihood is the slowest. The all-pairs grafting requires a tenth of the computation time of the PLE despite sharing similar amounts of data. The flexicast experiment with minimum bicasts has a smaller number of pairs, so it should be expected to save in computational time. The full MLE for this minimum pairs experiment is about 40 times faster than the PLE in this example. The grafting procedure with minimum pairs is extremely fast, comparable in speed to the time achieved by the polynomial estimator on the simpler tree discussed previously.

Shrub Comparison: Here we investigate only the two fastest estimators: the grafting procedure with minimum bicast pairs and the polynomial estimator. We study at a set of simple cases: shrubs with increasing numbers of children to see how the performance varies. For each configuration, we generated 1000 data sets from truncated geometric distributions on each link. Table 3 lists the average computation times for shrubs with two, size, and 10 children. The grafting procedure is uniformly faster on this test, even when it has to combine information from five trees in the 10-child example. The polynomial estimator performs at its best on small trees with small numbers of bins. However, when the true bin probability is small, we found that it can lead to negative estimates in a significant number of cases.

5.2.3. Statistical Efficiency

Statistical efficiency has received little attention in the literature, perhaps because of the inherent assumption that a large number of probes can be generated easily. In reality, however, active experimentation perturbs the network, and so too much probing in a short period of time can end up causing delay and losses on the network. If we spread the probing over an extended period, it will invalidate the stationarity assumption. As a result, one has to limit the number of probes, so any effective estimator must be reasonably efficient.

Table 3. Comparison of computation time for Grafting and the Polynomial estimator on shrubs with varying numbers of children.

Children	2	6	10
Grafting	.48s	.84s	.99s
Polynomial	1.02s	1.08s	1.13s

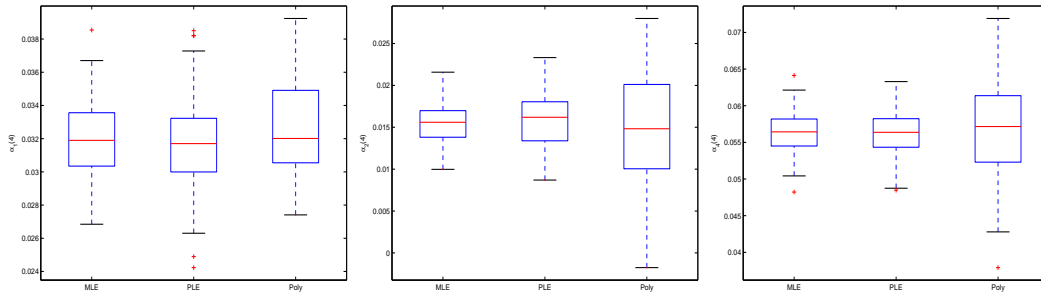


Fig. 6. Boxplots of the estimates for $\alpha_1(4)$, $\alpha_2(4)$, and $\alpha_4(4)$ for three multicast-based estimators.

We also note that it is difficult to compare the statistical efficiencies of estimates based on bicast or other flexicast experiments with those based on an omnicast experiment as they are not on equal footing. If the total number of probes is fixed, the total amount of expected traffic is different for omnicast and flexicast experiments. Even if we fix the total amount of expected traffic on all links under the schemes, the different links will have different expected numbers of probes. It is not possible to make the expected number of probes in each link be the same under the different schemes. Moreover, omnicast experiments contain information about all higher-order moments while the flexicast experiments are designed to sacrifice the higher-order moments to reduce data complexity.

To keep the comparisons meaningful, we will examine the efficiencies of estimation methods based on omnicast and bicast experiments separately. The comparisons here are based on a three-layer symmetric binary tree. The link delay distributions were chosen to be truncated geometric with five bins.

Figure 6 shows the performance of the full EM-based MLE, PLE, and the polynomial estimator for the last bin for three links: $\alpha_1(4)$, $\alpha_2(4)$, and $\alpha_4(4)$. The size of the omnicast experiment was 20,000 total probes. Recall that α_1 is the first link on the tree while α_4 corresponds to one of the receiver nodes. The figure suggests that the bias is small (medians close to the true values). The performance of the PLE is close to that of the MLE on links 1 and 4 with IQR ratios of 1.01 and 1.06. The performance on the interior link is somewhat worse with an IQR ratio of 1.38. This conclusion seems to be true in general; i.e., the relative performance of the PLE gets worse as one moves to the interior of the tree. So we would expect the performance to be poor for internal nodes in the middle of a large tree with many layers. The polynomial estimator, on the other hand, is considerably less efficient in all three cases with IQR ratios of 1.36, 3.18, and 2.47. Mean squared error (MSE, shown on the plots beneath the labels) tells a similar story. The MSE for the PLE is 1.4, 1.7 and 1.1 times as large as that of the MLE and the MSE for the polynomial estimator 1.5, 7.4, and 6.1 times as large as that of the MLE. We also compared the performance of the estimators for other bins and links. In general, the performance of the polynomial estimator is quite good for the first link ($\bar{\alpha}_1$) but gets progressively worse as we move deeper down the tree. This is because the polynomial estimator uses a lot of the data in estimating the first link but uses less and less data as we move down.

For the bicast-based estimators, we considered two different experiments: (i) all possible pairs and (ii) $\mathit{min}+1$. The estimators include all-pairs MLE, all-pairs grafting, $\mathit{min}+1$ MLE, and $\mathit{min}+1$ grafting. The comparisons of the estimators for the first and last bins are shown in Figures 7 and 8 respectively. In terms of precision, the $\mathit{min}+1$ MLE is very comparable

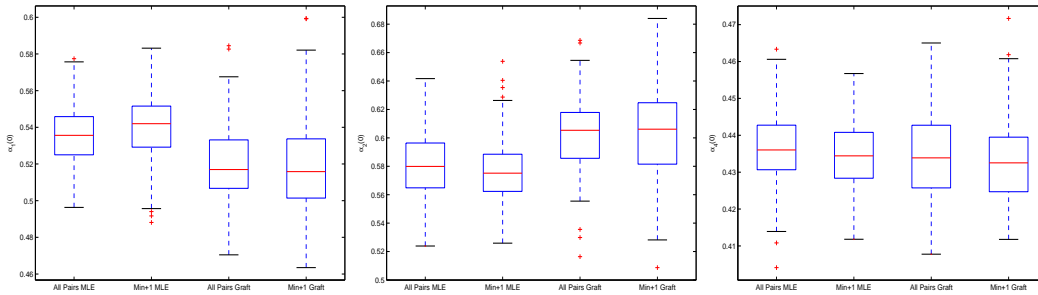


Fig. 7. Boxplots of the estimates for $\alpha_1(0)$, $\alpha_2(0)$, and $\alpha_4(0)$ for four bicast-based estimators.

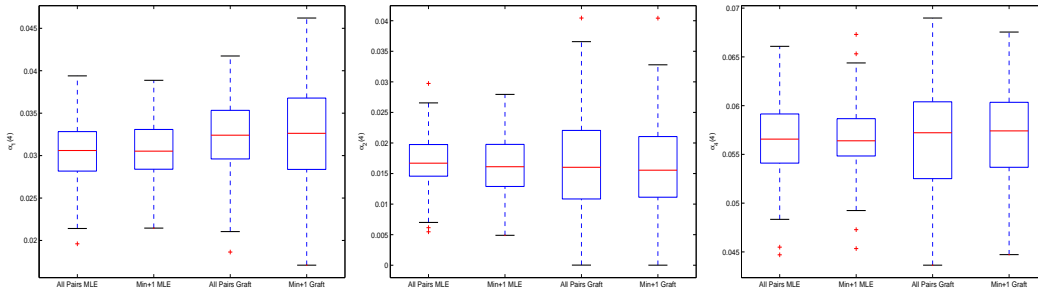


Fig. 8. Boxplots of the estimates for $\alpha_1(4)$, $\alpha_2(4)$, and $\alpha_4(4)$ for four bicast-based estimators.

to the all-pairs MLE except for estimating $\alpha_2(4)$. For $\alpha_4(4)$, it is actually slightly better. This can be explained by the fact that the *min+1* experiment allocates more probes for a scheme, $\langle 4, 5 \rangle$, that isolates link 4 thereby giving us a more precise estimate for this link. The grafting algorithms do slightly worse than the MLEs. For the all-pairs experiment, the IQR ratios of the grafting estimates to the MLEs for the zero bin on links 1, 2, and 4 are 1.267, 1.0231, and 1.4053. For the four bin, they are 1.2309, 2.1632, and 1.5634. For the *min+1* experiment, the IQR ratios for the zero bins on links 1, 2, and 4 are 1.4448, 1.6537, and 1.1899. For the four bin they are 1.7956, 1.4418, and 1.7397. In general the less precise algorithms do not perform as well in the interior of the tree at the tails of the distributions.

Examining the mean-squared error gives us further information that includes the apparent bias observed in the plots for some of the zero bins. The MSEs reflect this bias as the grafting algorithms have smaller MSEs than the MLE, despite having a larger spread. This effect disappears on the later bins and we see the same general trend as that indicated by the IQRs.

6. Optimal Allocation of Probes

We now turn to an important issue in designing flexicast experiments, viz., how to optimally allocate the number of probes among the various schemes within a flexicast experiment. The question of interest is the following: given a fixed budget of probes, how should they be allocated among the various flexicast schemes?

It turns out that the optimal allocations depend on the values of the unknown

delay distributions and the tree topology. This is called *local optimality* in the design literature (?). We describe results from a limited study based on binary, symmetric trees and bicast schemes to provide some insights and suggest suggest how one could go about studying the problem in general. A comprehensive study of this problem is part of on-going work.

We conduct our study on a three-layer binary, symmetric tree (right panel of Figure 2). For the link delay distributions, we use a geometric distributions truncated to five bins. Based on our experience with real data and the network simulator, the geometric distribution is a reasonable choice with its large mass at zero and decaying tails. We let the parameter of the distributions range from $p = 0.1$ to $p = 0.9$ with a step size of 0.1. This range allows us to consider very good links with light tails ($p = 0.9$) and more congested links with heavier tails ($p = 0.1$).

We use the following bicast experiment: $\mathcal{C} = \{\langle 4, 5 \rangle, \langle 6, 7 \rangle, \langle 5, 6 \rangle, \langle 4, 7 \rangle\}$. Note that the last two schemes split at node 1 while first two schemes split at a lower level. We view links 1, 4, 5, 6, and 7 as edge links while links 2 and 3 will be considered internal/back bone links. Links of the same type (edge, backbone) will have the same distribution. Based on this symmetry, the optimal proportion of probes sent $\langle 4, 5 \rangle$ and $\langle 6, 7 \rangle$ should be equal and that to $\langle 5, 6 \rangle$ and $\langle 4, 7 \rangle$ should be equal. Let τ refer to the total proportion of probes sent to first group; then the second group will get a proportion $1 - \tau$. The design problem is to identify the optimal value of τ .

The criterion we use here is D-optimality, commonly used in the experimental design literature. Specifically, the optimal value of τ is obtained by maximizing the determinant of the Fisher information matrix. As noted before, this value depends on the unknown parameters of the delay distributions, in addition to the tree topology. This is referred to as local D-optimality.

First, we consider the optimal τ for the situation in which all the link-level distributions are identical. Interestingly, the optimal value of τ is constant (around .75) as p ranges from .1 to about .8 and then decreases slightly to about 0.7 as p increases to 0.9. Based on this, the bicast pairs $\langle 4, 5 \rangle$ and $\langle 6, 7 \rangle$, which split at a lower level in the tree, should get about 35 – 37% of the probes each while the bicasts $\langle 5, 6 \rangle$ and $\langle 4, 7 \rangle$, which split at a higher level, should receive only about 12 – 15% of the probes. Note the pairs that split at the lower level provide the most information for estimating the receiver links. Further, the total number of probes at each link varies: under the above optimal setting, all the probes pass through the link at the top layer, links in layer two (the “backbone” links) each see about 3/4 of the probes, and those at layer three (receiver links) each see only 1/4 of the probes.

Figure 9 shows the optimal allocations for a two-dimensional situation: the edge links (link 1 and the receiver links) have the same truncated geometric distribution with “failure” probability p_1 (x-axis) while the backbone links 2 and 3 have the same distribution with probability p_2 (y-axis). The z-axis shows the values of τ , the optimal allocation. For most of the $p_1 - p_2$ values, the optimal value of τ is between 0.6 – 0.8, again indicating that a higher proportion should be sent to the bicast pairs that split at the lower level. The exception is when the failure probability p_2 of the backbone links become larger than about 0.8 and p_1 is in the range 0.1 – 0.5, the values of τ decrease, implying that the pairs that split at a higher level should get a larger proportion of the probes. This is likely due to the higher congestion on link 1 as compared with the backbone links. Probes splitting at node 1 help to provide a good estimate of link 1 which is important in this case for sorting out the links below it.

The above results provide some limited insights into the optimal allocation issue. As

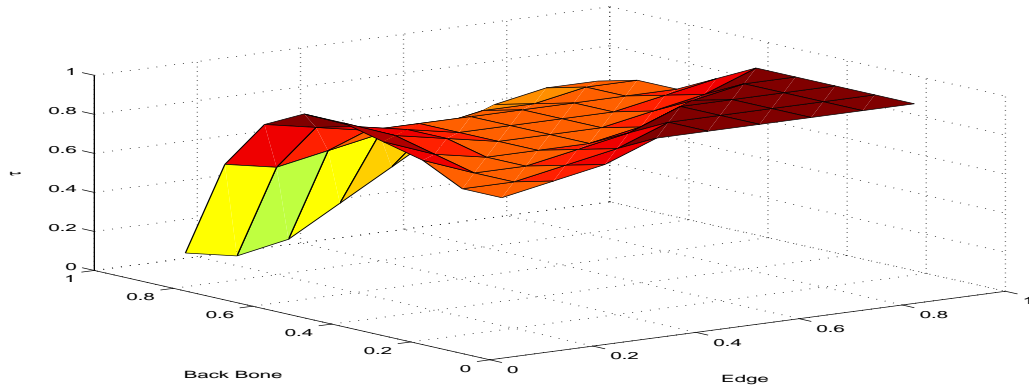


Fig. 9. The optimal allocation of probes will all links the same (left panel) and the optimal allocation of probes with interior links the same and edge links the same (right panel).

noted before, the optimal results depend on the unknown delay distributions, so one cannot obtain universally optimal results for all distributions. In practice, one can take a number of different approaches in getting efficient (or close to efficient) probe allocations. The simplest is to use prior knowledge based on historical information and treat it as the truth. A better alternative that uses prior information more formally is the use of Bayesian optimal design techniques (see, for example, ?). However, this is a non-linear optimal design problem that is complex. Perhaps the most practical alternative is to use a two-stage experimentation where the estimates from the first stage are used to obtain the optimal estimates for the second stage probing.

7. Simulation Studies

In this section, we use simulation to assess the performance of the estimation methods under two scenarios: a) under the stochastic model assumptions in Section 2.3 and b) using the ns-2 network simulator.

7.1. Model-Based Simulation

Our simulation studies showed that if the true link-level distributions are discrete, the MLE as well as grafting methods are able to recover the link-level estimates well without any bias. When the true distributions are continuous, however, the binning seems to introduce some bias. The problem arises from the fact that the end-to-end data are grouped into bins, so we have discretized sums instead of sums of discrete values from each link. The extent of the bias depends on the bin size, the link, and other variables.

To develop some insights, we considered a three-layer symmetric binary tree (right panel of Figure 2) and focused on the MLE for a minimum bicast experiment. Each link distribution was taken to be a mixture of exponential with mean one and point mass at 0. The point mass, corresponding to no delay, is common in many real situations. Various bin sizes and point mass probabilities were considered in the study. Figure 10 show the results from links 1, 2, and 4 (link 3 has the same behavior as 2 and links 5, 6, and 7 have the same behavior as 4) with a bin size of $q = 0.25$ and a point mass with probability $p = 0.2$.

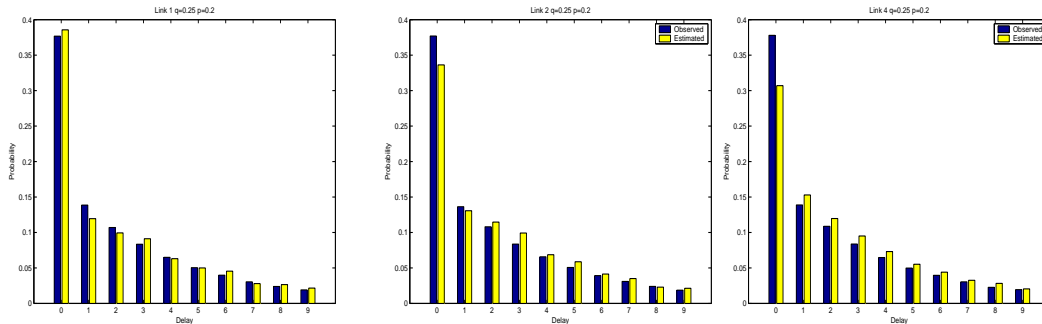


Fig. 10. Observed and estimated distributions for links 1, 2, and 4 showing bias when the estimation is applied to binned end-to-end data. The distributions are exponential with mean 1 mixed with point mass at 0 with probability .2. The bin size is .25

In general, the largest bias occurs for the zero bin ($\alpha_k(0)$); if point-mass probabilities or bin sizes increase, this bias gets smaller. For instance, for a bin size of $q = 0.25$ and point-mass probabilities of 0.1, 0.2, and 0.4, the underestimate of the zero probability on link 4 is 34%, 24%, and 9% respectively. For a bin size of $q = 4$, the corresponding underestimates reduce to 8%, 6%, and 4%. Note also from Figure 10 that the bias is greatest for the receiver node (link 4) and decreases as one goes up to the source node.

The effect of this bias on the other bins is much smaller. For the most part, the bias at the zero bin seems to be spread across the rest of the distribution. The estimate at bin 1 seems to compensate somewhat more than the other bins. There is also some compensation across links; the zero bin for link 1 is overestimated while the zero bins for other links are underestimated. We are currently exploring some methods for correcting this bias.

7.2. Network Simulation

We now examine the performance of the proposed estimators in a realistic network environment by using the ns-2 (?) simulation package. This allows one to construct any topology and generate traffic and transmit packets using real network protocol. It gives users control over the hardware and software aspects of a network including bandwidth, propagation delay, traffic volume, and traffic protocol. We constructed the topology shown in Figure 1 to mimic the UNC network. For links between core routers, we used 500 megabit links and for links to endpoints, we used 50 megabit links. Background traffic on the core links consists of 27 TCP connections and 5 UDP connections. TCP connections acknowledge reception of packets by the receiver. Lost packets are retransmitted by the sender and result in a slower transmission rate. Therefore, TCP connections are responsive to congestion. UDP connections do not have any of the above features and continue to send packets at a constant rate, thus being unresponsive to congestion patterns. On the edge links, the background consists of 6 TCP connections and 1 UDP connection. The probe traffic consists of 40 bit UDP packets using the multicast protocol.

Every one tenth of a second, the probing mechanism selects a scheme at random from $\mathcal{C} = \{\langle 4, 5 \rangle, \langle 6, 7 \rangle, \langle 8, 10 \rangle, \langle 11, 12 \rangle, \langle 13, 14 \rangle, \langle 15, 16 \rangle, \langle 17, 18 \rangle\}$ and sends a packet to its receivers. Probing lasts for 700 seconds resulting in about 1000 packets sent to each pair. This is

approximately the length of a session that we would use for monitoring a real network. The end-to-end delays are discretized using a bin size of $q = .00005s$. This is an extremely fine scale resulting in a maximum link delay setting of $b = 155$. Furthermore, the ns-2 package allows us to record the true link delays and hence directly obtain the link delay distributions for verification.

Figures 11 and 12 show the fitted distributions along with the observed distributions for selected links. We see the effect of the bias from binning discussed in the last section. Aside from this, the estimation procedure does a very good job of capturing the distribution despite a violation of the temporal and spatial independence assumptions in the model. The results show that the estimation procedure performs well in a real network setting.

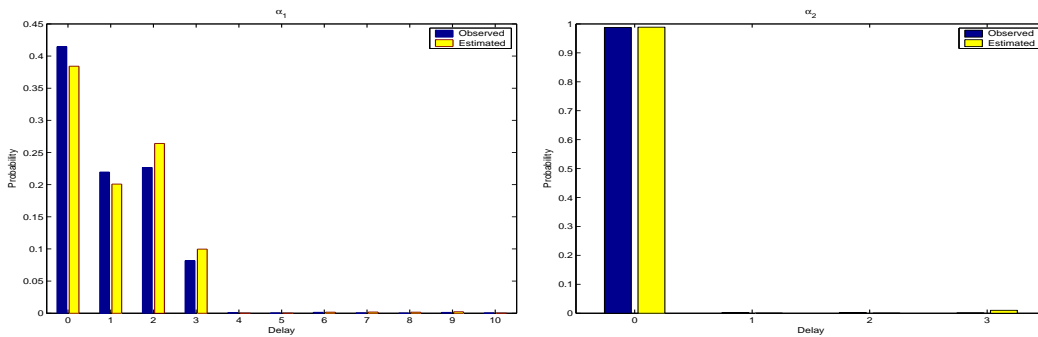


Fig. 11. Observed and estimated delays for links 1 and 2 of the ns-2 example.

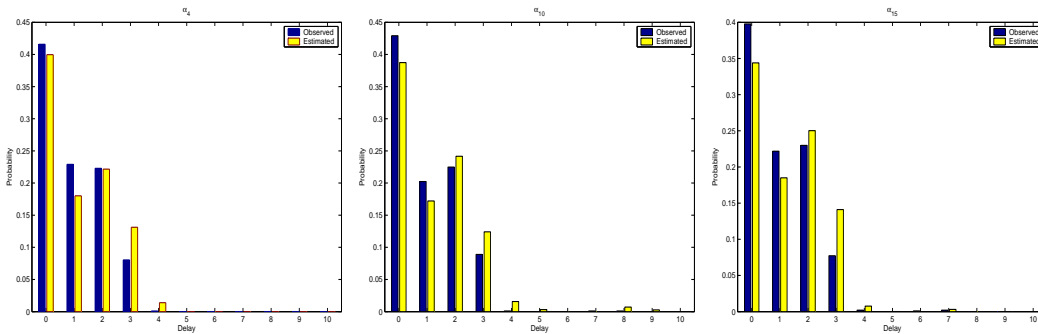


Fig. 12. Observed and estimated delays for links 6, 11 and 15 of the ns-2 example.

8. Application to the UNC Campus Network Data

We now illustrate the usefulness of the results developed here by applying them to the UNC campus network to assess VoIP capabilities. As discussed before, this real-time application requires excellent link quality in order to be successful. In particular, the presence of any large delays can significantly reduce the quality of the phone calls. In this section, we validate the usefulness of the methods developed in the paper by applying them to real

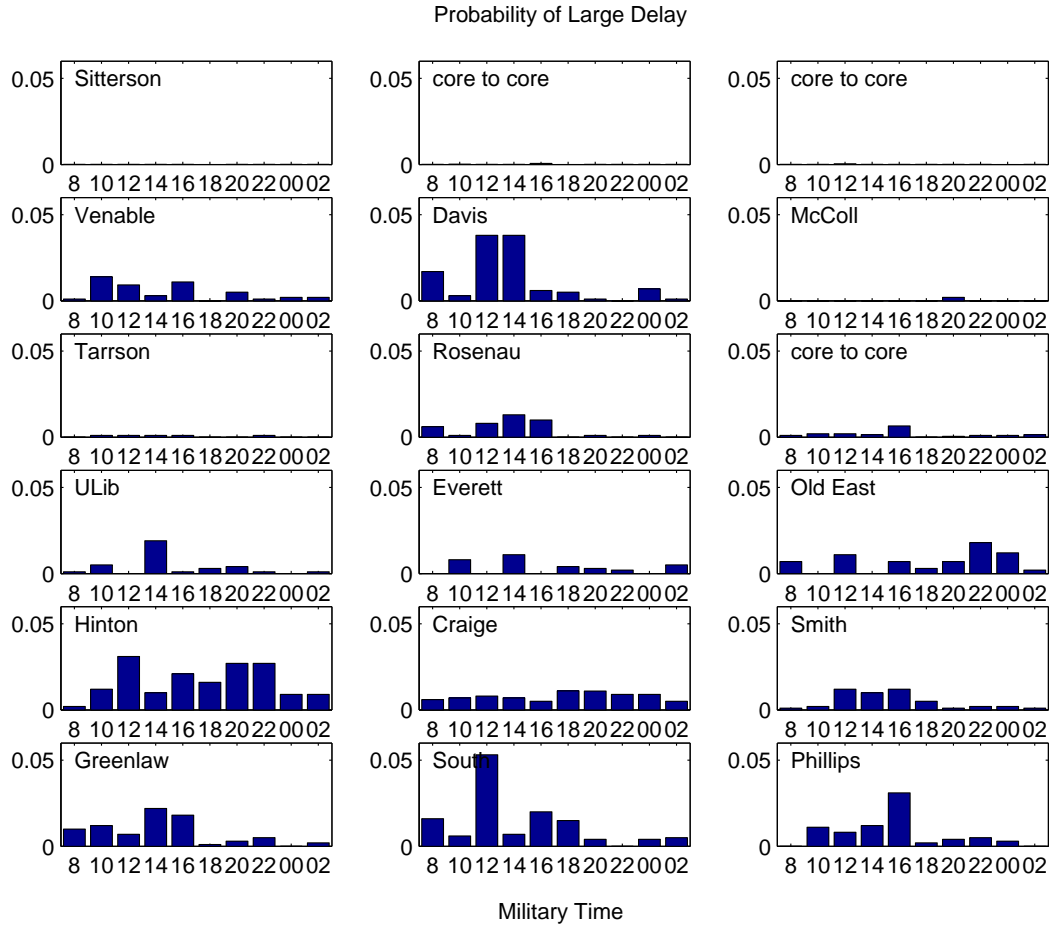


Fig. 13. Probability of large delay on each link throughout the day.

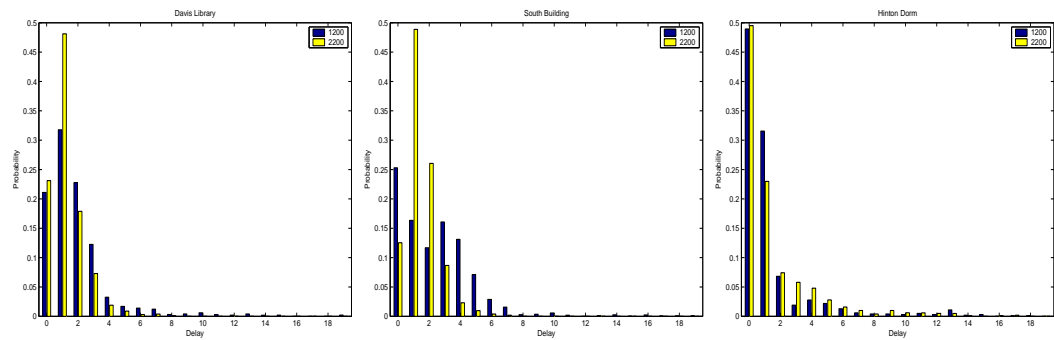


Fig. 14. Delay distribution for Davis Library at 12:00 p.m. and 10:00 p.m. (left panel) for South Bldg. (middle panel) and Hinton Dorm (right panel).

data. We have collected extensive amounts of data but report here only selected results based on data collected at two-hour intervals starting at 8:00 a.m. and ending at 2:00 a.m. on a typical school day.

The data were collected using a tool designed by Avaya Labs for testing a network's readiness for VoIP. There are two parts to the tool. First, there are the monitoring devices that are computers deployed throughout the network with the capability of exchanging VoIP-style traffic. These devices run an operating system that allows them to accurately measure the time at which packets are sent and received. The machines collect these time stamps and report them back to the second part of the system: the collection software. This software remotely controls the devices and determines all the features of each call, such as source-destination devices, start time, duration, and protocol which includes the inter-packet time intervals. The software collects the time stamps when the calls are finished and processes them. The processing consists of adjusting the time stamps to account for the difference among the machines' clocks, and then calculating the one-way end-to-end delays.

For the data collection, Sitterson served as the root, and we used seven bicast pairs to cover the 14 receiver nodes: $\mathcal{C} = \{\langle 4, 5 \rangle, \langle 6, 7 \rangle, \langle 8, 10 \rangle, \langle 11, 12 \rangle, \langle 13, 14 \rangle, \langle 15, 16 \rangle, \langle 17, 18 \rangle\}$. The network only allowed unicast transmission protocol, so back-to-back probing was used to simulate multicast transmissions. The span of time between the two packets comprising the back-to-back probe was on the order of a few nanoseconds while the time between successive probes was one tenth of a second. Prior experimentation using the call synthesis tool and this probing method leads us to believe that the correlation between the two packets on the shared links is close to one. Each probing session consisted of two passes through the pairs in the order presented. On each pass, each pair was probed for 50 seconds. Thus, we have 1000 probes for each pair during each monitoring session. Maximum likelihood estimation was used to deconvolve the link distributions.

In this section, we present results related only to discovering links that have significant probabilities of large delays. For this reason, we used a bin size of $q = .0002s$. Above this threshold, delays can become detrimental to call quality. We expect that most links in this network would have distributions with most of the mass on the zero bin. Nonetheless, a mass as low as .01 on the rest of the delays could prove troublesome.

Figure 13 shows the probability of delays larger than .0002 seconds at various times throughout the day. First, note that the links between the main core routers are of very high quality. The Sitterson outgoing link is also extremely good. The rest of the links do experience some congestion, varying over the course of the day. Many of the school buildings show a diurnal effect with lots of activity contributing to higher delays starting around noon and continuing throughout the afternoon. In particular, there is a bit of a spike around 4:00 p.m. This spike is evident on link 9 which is a larger core-to-core router. The dormitory links show more consistent traffic throughout the entire day with some elevated delays in the later evening. Links that show 1% or more large delays would likely require an upgrade in order to be able to handle the increased load placed upon them by VoIP which uses a more aggressive protocol than the prevalent TCP traffic. Several receiver links already show close to 5% large delays without a strong VoIP presence. Even the large link 9 seems to be problematic as it needs to perform almost flawlessly in order to handle considerably more traffic than the receiver links.

To look at some aspects of the analysis in more detail, we solved the inverse problem using a bin size of $q = .00002s$ for the time periods 12:00 p.m. and 10:00 p.m. This gave us ten times the resolution of the above analysis. Further, it allowed us to break down the previous analysis to see where the delays fall within the smallest bin. Figure 14 show the

first 20 bins of these detailed results for Davis Library, South Building, and Hinton Dorm respectively. The first thing to note is that most of the mass is still on the lowest bins so the vast majority of packets experience very little delay. Both Davis and South exhibit a strong diurnal effect. Unlike the dorm, the traffic in these buildings dies off late at night.

The analyses of other similar data sets collected on the network over a period of time showed remarkable stability in the results and conclusions. These delay probabilities indicated to the UNC IT group that the current network is not capable of supporting the VoIP application. From our point of view, the results are qualitatively consistent with the overall behavior that is to be expected for this network. This serves as a validation of the techniques of the techniques studied here, both from statistical and implementation perspectives.

9. Concluding Remarks

We have introduced a flexible class of probing experiments for active network tomography and studied the properties of several methods for estimating the link-level delay distributions of computer and communications networks. Both simulation and real data were used to illustrate the usefulness of the methodology. The use of the full EM algorithm is practical for small to moderate trees, especially with minimum-identifiable flexicast experiments (bicast and unicast schemes). With larger networks, the grafting algorithm proposed in the paper provides a practical alternative. The fast algorithm is especially useful for monitoring the networks to detect degradation in quality of service and quickly localize the problem to links or small regions of the network. We are currently studying monitoring and diagnostic procedures for this problem.

We have followed the literature in the area in making a number of simplifying assumptions. For example, the logical topology of the network has been assumed to be single source, known and fixed. Extensions to multi-source topologies, although conceptually straightforward, nevertheless introduce technical challenges that are currently under investigation. Further, in practice, the network's topology can be to a certain degree unknown or changing periodically. There is on-going work in the network engineering literature to address these topics. The assumption of spatio-temporal stationarity is also commonly made in this area. As we have noted, the temporal stationarity assumption is not critical as the time-between-probes is on the order of milliseconds. The spatial assumption is more problematic although more realistic models can be developed only in the context of specific real networks. There is also additional work required to assess the performance of the methods for larger networks. We note, however, that even if the actual network is large, one typically aggregates many of the links and focuses on a smaller topology for studying network performance.

Acknowledgments: The authors would like to thank the Editor, the Associate Editor and two referees for useful comments and suggestions. The paper is part of the first author's Ph.D. dissertation. The research was supported in part by NSF grants IIS-9988095, CCR-0325571, DMS-0204247, and DMS-0505535. The authors are grateful to: Jim Landwehr, Lorraine Denby and Jean Meloche of Avaya Labs for making their ExpertNet tool available for VoIP data collection and for many useful discussions on network monitoring; Yinghan Yang for assistance with data collection; Jim Gogan and his team from the IT Division at UNC for their technical support in deploying the ExpertNet tool on their campus network, for troubleshooting hardware problems and for providing information about the structure

and topology of the network; Don Smith of the CS Department at UNC for helping us establish the collaboration with the UNC IT group; and Steve Marron for many helpful comments during the course of this research.

A. Proof of Proposition 1

We first establish sufficiency of an omnicast experiment. This will show that an individual k -cast scheme identifies all of the distributions on the paths between source, branching nodes, and receivers of its subtrees. We can then show that the above conditions guarantee that we have enough subtrees to solve for every link delay distribution. The proof of necessity will proceed by contradiction.

For omnicast probing, we consider two cases:

Case 1 – Receiver node k : Consider all omnicast probes that result in zero delay on all remaining receivers except k . This set of probes allow us to estimate $\vec{\alpha}_k$ as these probes consist of direct observations from link k .

Case 2 – Internal node k : We proceed by induction. Suppose we have identified the distributions for all links that are descendants of k . Let $\mathcal{R}(k)$ represent the receivers descended from k . Consider probes that result in zero delay on all nodes except those in $\mathcal{R}(k)$ which all experience i delay. Let $\gamma^k(i)$ be the probability that each $r \in \mathcal{R}(k)$ has an end-to-end delay of i . From this, we can estimate $\gamma^k(i)$ for each i . Since we have estimates of link delay distributions of the descendants of k , we can now estimate $\vec{\alpha}_k$.

This proof implies that a single k -cast scheme will identify the following distributions: the path between the source and the first splitting node, the paths between any two splitting nodes, and the paths between a splitting node and a receiver. Now we focus on a collection of flexicast schemes. Here we consider three cases.

Case 1: There is some k -cast scheme \mathcal{C}_j in which branching occurs at node 1, the only child of the root node. Based on the omnicast identifiability proof, this experiment identifies the delay distribution for link 1, $\vec{\alpha}_1$.

Case 2: Let s be some internal node. Assume that we have identified all of the delay distributions for links $k \in \mathcal{P}_{0,f(s)}$. There is a scheme \mathcal{C}_j for which branching occurs at node s . This scheme identifies the path-level distribution $\vec{\pi}_{0,s}$. We can construct $\vec{\pi}_{0,f(s)}$ and solve for $\vec{\alpha}_s$:

$$\begin{aligned} \alpha_s(0) &= \pi_{0,s}(0)/\pi_{0,f(s)}, \\ \alpha_s(d) &= \frac{1}{\pi_{0,f(s)}(0)} \left[\pi_{0,s}(d) - \sum_{\delta=\max(0,d-B_{f(s)})}^{d-1} \alpha_s(\delta)\pi_{0,f(s)}(d-\delta) \right] \quad \forall d = 1, \dots, b, \end{aligned}$$

where $B_{f(s)}$ is the maximum delay up to this node. We call this solution peeling since we are peeling the unknown distribution from the path-level distributions. It can be used more generally and take other functional forms.

Case 3: Let r be some receiver node. Assume that we have identified all of the delay distributions for links $k \in \mathcal{P}_{0,f(r)}$. There is some scheme \mathcal{C}_j which probes receiver r . From this, we can estimate the path probability $\pi_{0,r}$. We can construct $\vec{\pi}_{0,f(r)}$ and use peeling to get $\vec{\alpha}_r$.

It is easy to see the necessity of covering all of the receivers: if we do not probe a receiver, we can never estimate its link delay distribution. To see the necessity of branching at each

internal node, consider a collection of schemes in which branching occurs at all internal nodes except some node s . Each link $d \in \mathcal{D}(s)$ will always occur as part of a logical link that also includes link s . We will be able to obtain estimates for $\vec{\pi}_{f(s),d}$ for each $d \in \mathcal{D}(s)$ but we will have no information with which to peel the two apart. In essence, these estimates are like unicast measurements which are not sufficient for estimation.

Add references to (?), (?), (?), (?), (?); Remove reference to our tech report.