

Efficient and Dynamic Routing Topology Inference From End-to-End Measurements

Jian Ni, *Member, IEEE*, Haiyong Xie, *Member, IEEE*, Sekhar Tatikonda, *Member, IEEE*, and Yang Richard Yang, *Member, IEEE*

Abstract—Inferring the routing topology and link performance from a node to a set of other nodes is an important component in network monitoring and application design. In this paper, we propose a general framework for designing topology inference algorithms based on additive metrics. The framework can flexibly fuse information from multiple measurements to achieve better estimation accuracy. We develop computationally efficient (polynomial-time) topology inference algorithms based on the framework. We prove that the probability of correct topology inference of our algorithms converges to one exponentially fast in the number of probing packets. In particular, for applications where nodes may join or leave frequently such as overlay network construction, application-layer multicast, and peer-to-peer file sharing/streaming, we propose a novel sequential topology inference algorithm that significantly reduces the probing overhead and can efficiently handle node dynamics. We demonstrate the effectiveness of the proposed inference algorithms via Internet experiments.

Index Terms—Network measurement, network monitoring, network tomography, routing topology inference.

I. INTRODUCTION

DEVELOPING a scalable tool to infer the routing topology and link performance from a node to a set of other nodes is an important challenge. In *network monitoring*, this tool can help a network operator obtain routing information and network internal characteristics (e.g., loss rate, delay, utilization) from its network to a set of other collaborating networks that are separated by nonparticipating autonomous networks. In *application design*, this tool can be particularly useful for peer-to-peer (P2P) style applications where a node communicates with a set of other nodes (called *peers*) for file sharing and multimedia streaming. For example, a node may want to know the routing topology to other nodes so that it can select peers with low or no route overlap to improve resilience against network failures

Manuscript received February 24, 2008; revised November 18, 2008 and March 24, 2009; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor N. Duffield. First published September 22, 2009; current version published February 18, 2010. An earlier version of this paper was presented at the 27th IEEE INFOCOM, Phoenix, AZ, April 2008.

J. Ni is with the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL 61801 USA (e-mail: jianni@illinois.edu).

H. Xie is with Akamai Technologies, San Mateo, CA 94402 USA (e-mail: hxie@akamai.com).

S. Tatikonda is with the Department of Electrical Engineering, Yale University, New Haven, CT 06520 USA (e-mail: sekhar.tatikonda@yale.edu).

Y. R. Yang is with the Department of Computer Science, Yale University, New Haven, CT 06520 USA (e-mail: yry@cs.yale.edu).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TNET.2009.2022538

[1]. As another example, a streaming node using multipath may want to know both the routing topology and the link-loss rates so the selected paths have low loss correlation [2].

So far, there are two primary approaches to infer the routing topology and link performance in a communication network. Both have their limitations. One approach is to use tools based on measurements or feedback messages of the internal nodes (e.g., routers). Such an approach is limited, as today's communication networks are evolving towards more decentralized and private administration. For example, a common approach to obtain the routing topology from a source node to a destination node in the Internet is to use *traceroute*. Traceroute relies on internal routers' responding to traceroute requests and returning Internet Control Message Protocol messages. However, an increasing number of routers in the Internet today will block traceroute requests due to privacy and security concerns. These routers are known as *anonymous routers* [29], and their existence makes the routing topology inferred by traceroute-like tools inaccurate. Furthermore, traceroute-like tools cannot discover layer-2 switches and multiprotocol label switching (MPLS) paths that are increasingly being deployed.

The other approach, known as *network tomography*, utilizes end-to-end packet probing measurements (such as packet loss and delay measurements) conducted by the end hosts and does not require extra cooperation from the internal nodes (except the basic packet forwarding functionality). Under a network tomography approach, a source node will send probes to a set of destination nodes. The basic idea is to utilize the correlations among the observed losses and delays of the probes at the destination nodes to infer the network structure and internal characteristics. Due to its flexibility and reliability, network tomography has attracted many recent studies [7], [11]. Many previous network tomography studies are based on multicast probing because of its effectiveness and probing efficiency (e.g., [6], [13]–[15], [19], [21], [22]). Since IP multicast is not widely deployed in the current Internet, unicast network tomography approaches based on back-to-back unicast packet pairs or strings have also been investigated (e.g., [3], [10], [12], [16], [24], and [27]).

Two fundamental challenges of network tomography approaches include *computational complexity* and *probing scalability* (especially under unicast probing). These limit the number of destination nodes that a source node can infer. In addition, the focus of previous studies is on a relatively stable set of nodes, while in many applications and networks (e.g., overlay network construction, application-layer multicast, P2P file sharing and streaming, and wireless ad-hoc and sensor networks), nodes may join or leave a session frequently [26]. To handle node dynamics efficiently, we need fast and scalable

inference procedures/algorithms that have low computational complexity, fast convergence rate, and small probing overhead.

In this paper, we study the problem of inferring the network routing topology from a source node to a set of destination nodes,¹ where the set can be dynamic. We summarize our contributions as follows.

- We present a general framework for designing network routing topology inference algorithms based on additive metrics. We show how to construct additive metrics and estimate the (shared) path lengths using end-to-end multicast and unicast packet probing measurements as well as traceroute type measurements. The framework can flexibly fuse information available from multiple measurements to achieve better estimation accuracy and faster convergence rate.
- Based on the framework, we develop two computationally efficient (polynomial-time) topology inference algorithms. In particular, we propose a novel sequential topology inference algorithm, which significantly reduces the probing overhead under unicast probing. In addition, it can efficiently handle dynamic node joining and leaving and thus is particularly desirable for applications and networks where node dynamics are prevalent.
- Under some assumptions, we prove that the probability of correct topology inference of our algorithms converges to one exponentially fast in the sample size (number of probing packets). We also demonstrate the effectiveness of our algorithms via Internet experiments. For the most effective inference algorithm (a hybrid scheme that incorporates both network tomography measurements and traceroute measurements), the inferred topology is approximately 100% correct when no more than 20% of the internal routers do not respond to traceroute probing. It can still correctly identify approximately 50% of the internal nodes, solely from network tomography measurements, even when none of the internal routers respond to traceroute probing.

We organize this paper as follows. In Section II, we review some related work. In Section III, we introduce the network model and the inference problems. In Section IV, we describe how to construct additive metrics and estimate the (shared) path lengths from end-to-end measurements. In Sections V and VI, we propose and analyze a neighbor-joining based topology inference algorithm and a sequential topology inference algorithm, which can be applied to any additive metric. We design Internet routing tree topology inference schemes and evaluate their performance via Internet experiments in Section VII. This paper is concluded in Section VIII.

II. RELATED WORK

Multicast routing tree topology inference was studied in [13]–[15] and [22]. Ratnasamy *et al.* [22] proposed a grouping algorithm to infer the tree topology based on shared losses observed at the destination nodes. Duffield *et al.* [15] extended

¹We use destination nodes for simplicity, while the nodes can be relay nodes or peer nodes of the source node in real applications.

the grouping algorithm and also proposed a maximum-likelihood approach and a Bayesian approach to estimate the tree topology. They further generalized the grouping algorithm to any estimable and monotonic performance metrics [13]. Similar generalization was made in [3]. The rooted neighbor-joining (RNJ) algorithm proposed in this paper is also a grouping type algorithm that recovers the tree topology by recursively joining the neighbors on the tree. This agglomerative joining/grouping idea has been used in *clustering* for building cluster trees (e.g., [18]) and in *evolutionary biology* for building phylogenetic trees (e.g., [17] and [23]).

Unicast routing tree topology inference was studied in [8], [12], and [24]. Coates *et al.* [12] introduced a *sandwich* probing technique to conduct delay measurements and proposed a Markov chain Monte Carlo procedure to search the most likely tree topologies. Castro *et al.* [8] and Shih *et al.* [24] formulated the inference problem as a *hierarchical clustering* problem and developed several hierarchical clustering algorithms to recover the tree topology.

The limitations of existing topology inference algorithms are summarized in the beginning of Section VI. To the best of our knowledge, the sequential topology inference algorithm proposed in this paper is a first effort to address the issues of node dynamics and probing scalability for network routing topology inference.

III. NETWORK MODEL AND INFERENCE PROBLEMS

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote the topology of the network, which is a directed graph with node set \mathcal{V} (end hosts, internal switches and routers, etc.) and link set \mathcal{E} (communication links that join the nodes). For any nodes i and j in the network, if the underlying routing algorithm returns a sequence of links that connect j to i , we say j is *reachable* from i . We assume that during the measurement period, the underlying routing algorithm determines a unique path from a node to another node that is reachable from it. Hence, the *physical routing topology* from a source node to a set of (reachable) destination nodes is a (directed) tree.

From the physical routing topology, we can derive a *logical routing tree*, which consists of the source node, the destination nodes, and the branching nodes (internal nodes with at least two outgoing links) of the physical routing tree [15], [22]. A logical link may comprise more than one consecutive physical links, and the degree of an internal node on the logical routing tree is at least three. An example is shown in Fig. 1. In this paper, we consider topology inference of logical routing trees and use the *routing tree* to express the *logical routing tree* for simplicity.

Suppose s is a source node in the network and D is a set of destination nodes that are reachable from s . Let $T(s, D) = (V, E)$ denote the routing tree from s to nodes in D with node set V and link set E . Let $U = s \cup D$ be the set of terminal nodes (e.g., end hosts), which are nodes of degree one.

Every node $k \in V$ has a *parent* $f(k) \in V$ and a set of children $c(k) = \{j \in V : f(j) = k\}$, except that the source node (root of the tree) has no parent and the destination nodes (leaves of the tree) have no children. For notational simplification, we also use e_k to denote link $(f(k), k)$. We use $\mathcal{P}(i, j)$ to denote the sequence of links that connect j to i on the routing tree.

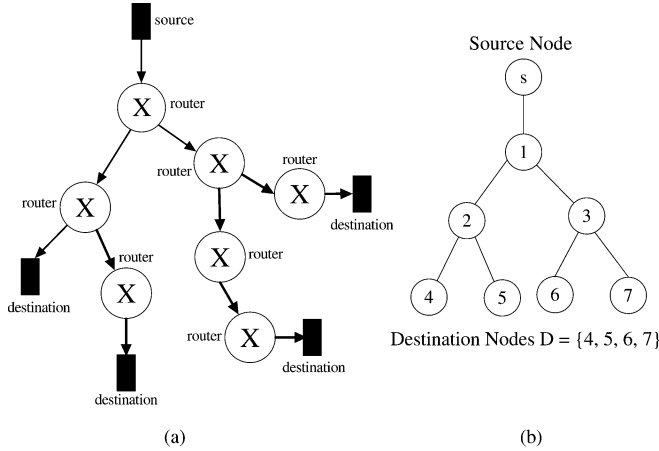


Fig. 1. (a) The physical routing topology and (b) the associated logical routing tree with a single source node and multiple destination nodes.

Each link $e \in E$ is associated with a parameter θ_e (e.g., success rate, delay distribution, utilization, etc.). The network inference problems involve using measurements taken at the terminal nodes to infer:

- (1) the topology of the (logical) routing tree;
- (2) link parameters θ_e of the links on the routing tree.

In this paper, we focus on routing tree topology inference. Link parameter estimation with known routing tree topology was studied in [6], [10], [20], [21], and [27].

A. Probing Model

A source node can employ different probing techniques to send probes (packets) to a set of destination nodes. Under *multicast probing*, when an internal node on the routing tree receives a packet from its parent, it will send a copy of the packet to all its children on the tree. Hence, the packets of the same probe received by different destination nodes have exactly the same network experience (loss, delay, etc.) in the shared links.

Under *unicast probing*, the source node sends a string of back-to-back unicast packets to the destination nodes, one packet for each destination node, respectively (to mimic the transmission of a multicast probe). We call it a $1 \times k$ *packet string probing* if the string size (i.e., number of probed destination nodes) is k . Since the back-to-back packets are very close to each other, it is normally assumed that these packets have the same network experience in the shared links just like a multicast probe. We will relax this assumption in Section IV-B.

For a probe sent by source node s to the destination nodes in D , we define a set of *link state variables* Z_e for all links $e \in E$ on the routing tree $T(s, D)$. Z_e takes value in a set \mathcal{Z} . The distribution of Z_e is parameterized by θ_e , e.g., $\mathbb{P}(Z_e = z) = \theta_e(z), \forall z \in \mathcal{Z}$.

The transmission of a probe from s to nodes in D will induce a set of *outcome variables* on the routing tree. For each node $k \in V$, we use X_k to denote the (random) outcome of the probe at node k . X_k takes value in a set \mathcal{X} . By *causality*, the outcome of the probe at node k (i.e., X_k) is determined by the outcome

of the probe at node k 's parent $f(k)$ (i.e., $X_{f(k)}$) and the state of link $e_k = (f(k), k)$ (i.e., Z_{e_k})

$$X_k = g(X_{f(k)}, Z_{e_k}). \quad (1)$$

In network tomography studies, it is normally assumed that the link states are independent from link to link (*spatial independence*) and are *stationary* during the measurement period [6], [11]. (Note that these assumptions may not hold in real networks like the Internet. We develop a hybrid scheme in Section VII to improve the estimation accuracy of the pure network tomography scheme for Internet routing tree topology inference.) Under those assumptions, we can show that the outcome variables X_k induced by the transmission of a probe form a *Markov random field* (MRF) on the routing tree [19]. Specifically, for each node $k \in V$, the conditional distribution of X_k given other random variables ($X_j : j \neq k$) on $T(s, D)$ is the same as the conditional distribution of X_k given just its neighboring random variables ($X_j : j \in f(k) \cup c(k)$) on $T(s, D)$. For MRFs on trees, under mild conditions, the tree topology and the link parameters can be *identified* (uniquely determined) by the joint distributions of the outcome variables at pairs and triples of the terminal nodes on the tree [9], [19].

In actual network inference problems, however, the joint distributions of the outcome variables at the terminal nodes are not given. We can estimate the joint distributions based on measurements taken at the terminal nodes. Specifically, the source node will send a sequence of n probes, and there are in total n outcomes $X_V^{(t)} = (X_k^{(t)} : k \in V), t = 1, 2, \dots, n$, one for each probe. For the t th probe, only the outcomes $X_U^{(t)} = (X_k^{(t)} : k \in U = s \cup D)$ at the terminal nodes can be measured and observed. We can estimate the joint distributions of the outcome variables at the terminal nodes using the empirical distributions, which will converge to the actual stationary distributions almost surely if the link state processes are stationary and ergodic during the measurement period.

B. Network Tomography Examples

1) *Example 1: Link Loss Inference* [6]: The link state variable Z_e is a Bernoulli random variable that takes value one with probability α_e if the probe can go through link e and value zero with probability $1 - \alpha_e \triangleq \bar{\alpha}_e$ if the probe is lost on the link. α_e is called the *success rate* or *packet delivery rate* of link e , and $\bar{\alpha}_e$ is called the *loss rate* of link e . The outcome variable L_k is also a Bernoulli random variable, which takes value one if the probe successfully reaches node k . For this example, we have ($L_s \equiv 1$)

$$L_k = L_{f(k)} \cdot Z_{e_k} = \prod_{e \in \mathcal{P}(s, k)} Z_e. \quad (2)$$

2) *Example 2: Link Utilization Inference* [14]: The link state variable Z_e is a Bernoulli random variable that takes value one with probability γ_e if the probe does not experience any queuing delay on link e and value zero with probability $1 - \gamma_e \triangleq \bar{\gamma}_e$ otherwise. $\bar{\gamma}_e$ can be viewed as the utilization of link e . The outcome variable U_k is also a Bernoulli random variable, which takes value one if the packet reaches node

k with no queueing delay. For this example, we also have ($U_s \equiv 1$)

$$U_k = U_{f(k)} \cdot Z_{e_k} = \prod_{e \in \mathcal{P}(s,k)} Z_e. \quad (3)$$

3) *Example 3: Link Delay Inference [21]*: The link state variable Z_e is a random variable denoting the random (queueing) delay of link e . θ_e can be a certain moment of Z_e , e.g., $\theta_e = \text{var}(Z_e)$; or the distribution of Z_e is parameterized by θ_e , e.g., $\theta_e(i) = \mathbb{P}(Z_e = i)$, $i \in \mathcal{Z}$. The outcome variable T_k is the cumulative (queueing) delay experienced by the probe from s to node k . For link delay inference, we have ($T_s \equiv 0$)

$$T_k = T_{f(k)} + Z_{e_k} = \sum_{e \in \mathcal{P}(s,k)} Z_e. \quad (4)$$

IV. CONSTRUCT ADDITIVE METRICS

Let $T(s, D) = (V, E)$ be a routing tree with source node s and destination nodes D . We say d is an *additive metric* on $T(s, D)$ if

- a) $0 < d(e) < \infty$, $\forall e \in E$
- b) $d(i, j) = \sum_{e \in \mathcal{P}(i,j)} d(e)$, $\forall i, j \in V$.

$d(e)$ can be viewed as the *length* of link e and $d(i, j)$ can be viewed as the *distance* between nodes i and j . Remember $U = s \cup D$ is the set of terminal nodes on the tree. We use $d(U^2) = \{d(i, j) : i, j \in U\}$ to denote the distances between the terminal nodes. It is known that the topology and link lengths of a tree are uniquely determined by the distances between the terminal nodes under an additive metric [5].

Suppose the source node s is fixed. For any destination node $i \in D$, let $\rho(i) = d(s, i)$ denote the *path length* from s to i (under additive metric d).

For any pair of destination nodes $i, j \in D$, let \underline{ij} denote their *nearest common ancestor* on $T(s, D)$ (i.e., the ancestor of both nodes i and j that is closest to i and j on the routing tree). For example, in Fig. 1(b), the nearest common ancestor of destination nodes 4 and 5 is node 2, and the nearest common ancestor of destination nodes 4 and 6 is node 1. Let $\rho(i, j) = d(s, \underline{ij})$ denote the *shared path length* from s to i and j (i.e., the distance between s and the nearest common ancestor of i and j).

Let $\rho(s, D) = \{\rho(i) : i \in D\}$ denote the path lengths from s to nodes in D and $\rho(s, D^2) = \{\rho(i, j) : i, j \in D\}$ denote the shared path lengths from s to pairs of nodes in D . Note that there is a one-to-one mapping between $d(U^2)$ and $\rho(s, D) \cup \rho(s, D^2)$. We can recover the topology of the routing tree if we know either $d(U^2)$ or $\rho(s, D) \cup \rho(s, D^2)$. The key is to construct an additive metric for which we can derive/estimate $d(U^2)$ or $\rho(s, D) \cup \rho(s, D^2)$ from end-to-end measurements.

A. Additive Metrics Based on Multicast Probing

1) *Loss-Based Additive Metric*: For Example 1 in Section III-B, if $0 < \alpha_e < 1$, $\forall e$, then we can construct an additive metric d_l with link length

$$d_l(e) = -\log \alpha_e, \quad \forall e \in E. \quad (5)$$

Under the spatial independence assumption that the link states are independent from link to link, $\rho_l(s, D) \cup \rho_l(s, D^2)$ can be obtained by

$$\begin{aligned} \rho_l(i) &= -\log \mathbb{P}(L_i = 1), \quad i \in D \\ \rho_l(i, j) &= -\log \frac{\mathbb{P}(L_i = 1)\mathbb{P}(L_j = 1)}{\mathbb{P}(L_i L_j = 1)}, \quad i, j \in D. \end{aligned} \quad (6)$$

2) *Utilization-Based Additive Metric*: Similarly for Example 2 in Section III-B, if $0 < \gamma_e < 1$, $\forall e$, then we can construct an additive metric d_u with link length

$$d_u(e) = -\log \gamma_e, \quad \forall e \in E. \quad (7)$$

$\rho_u(s, D) \cup \rho_u(s, D^2)$ can be obtained by

$$\begin{aligned} \rho_u(i) &= -\log \mathbb{P}(U_i = 1), \quad i \in D \\ \rho_u(i, j) &= -\log \frac{\mathbb{P}(U_i = 1)\mathbb{P}(U_j = 1)}{\mathbb{P}(U_i U_j = 1)}, \quad i, j \in D. \end{aligned} \quad (8)$$

3) *Delay-Based Additive Metric*: For Example 3 in Section III-B, if $0 < \text{var}(Z_e) < \infty$, $\forall e$, then we can construct an additive metric d_v with link length

$$d_v(e) = \text{var}(Z_e), \quad \forall e \in E. \quad (9)$$

$\rho_v(s, D) \cup \rho_v(s, D^2)$ can be obtained by

$$\begin{aligned} \rho_v(i) &= \text{var}(T_i), \quad i \in D \\ \rho_v(i, j) &= \text{cov}(T_i, T_j), \quad i, j \in D. \end{aligned} \quad (10)$$

As in (6), (8), and (10), if we know the pairwise joint distributions of the outcome variables at the terminal nodes, then we can construct an additive metric and derive $\rho(s, D) \cup \rho(s, D^2)$. In actual network inference problems, we are not given such distributions. We can use measurements taken at the terminal nodes to estimate the distributions.

Let s send a sequence of n probes to (a subset of) destination nodes in D . For any probed node i , let $T_i^{(t)}$ be the measured (one-way) delay of the t th probe from s to i , with $T_i^{(t)} = \infty$ meaning that i does not receive the t th probe. We use $T_i^{\min} = \min_t T_i^{(t)}$ to approximate the propagation delay from s to i .

The loss outcomes can be derived from the delay measurements as follows:

$$L_i^{(t)} = \begin{cases} 1, & T_i^{(t)} < \infty \\ 0, & T_i^{(t)} = \infty. \end{cases}$$

As in [14], the utilization outcomes can be derived from the delay measurements as follows:

$$U_i^{(t)} = \begin{cases} 1, & T_i^{(t)} - T_i^{\min} \leq \epsilon \\ 0, & T_i^{(t)} - T_i^{\min} > \epsilon \end{cases}$$

where ϵ is a small value, e.g., 0.1 ms, to account for possible measurement error.

We can construct explicit estimators for the path lengths and shared path lengths in (6), (8) as follows:

$$\hat{\rho}_l(i) = -\log \bar{L}_i, \quad \hat{\rho}_l(i, j) = -\log \bar{L}_i \bar{L}_j / \bar{L}_{ij} \quad (11)$$

$$\hat{\rho}_u(i) = -\log \bar{U}_i, \quad \hat{\rho}_u(i, j) = -\log \bar{U}_i \bar{U}_j / \bar{U}_{ij} \quad (12)$$

where

$$\begin{aligned}\bar{L}_i &= \frac{1}{n} \sum_{t=1}^n L_i^{(t)}, \quad \bar{L}_{ij} = \frac{1}{n} \sum_{t=1}^n L_i^{(t)} L_j^{(t)} \\ \bar{U}_i &= \frac{1}{n} \sum_{t=1}^n U_i^{(t)}, \quad \bar{U}_{ij} = \frac{1}{n} \sum_{t=1}^n U_i^{(t)} U_j^{(t)}.\end{aligned}$$

Similarly, we can construct explicit estimators for the path lengths and shared path lengths in (10) using sample variances and sample covariances

$$\hat{\rho}_v(i) = v\hat{a}r(T_i), \quad \hat{\rho}_v(i, j) = c\hat{o}v(T_i, T_j) \quad (13)$$

where

$$\begin{aligned}v\hat{a}r(T_i) &= \frac{1}{n-1} \sum_{t=1}^n (T_i^{(t)} - \bar{T}_i)^2 \\ c\hat{o}v(T_i, T_j) &= \frac{1}{n-1} \sum_{t=1}^n (T_i^{(t)} - \bar{T}_i)(T_j^{(t)} - \bar{T}_j) \\ \bar{T}_i &= \frac{1}{n} \sum_{t=1}^n T_i^{(t)}.\end{aligned}$$

In the above equations, we assume $T_i^{(t)}, T_j^{(t)} < \infty$ (i.e., there is no packet loss). For lost packets, we will not count them in the computation.

Notice that possible clock asynchronization between the source node and the destination nodes will not affect the estimators in (11)–(13).

A convex combination of several additive metrics is still an additive metric. In order to fuse information from multiple measurements, we can construct a new additive metric using a convex combination of d_l, d_u, d_v : $d_t = a_l d_l + a_u d_u + a_v d_v$ with $a_l + a_u + a_v = 1$. The (estimated) path lengths and shared path lengths under the new additive metric can be easily computed: $\hat{\rho}_t = a_l \hat{\rho}_l + a_u \hat{\rho}_u + a_v \hat{\rho}_v$. In practice, we can select the coefficients based on the current network state or to minimize the variance of the new estimator $\hat{\rho}_t$.

B. Additive Metrics Based on Unicast Packet Pair Probing

The validity of (6), (8), and (10) depends on the fact that the packets of the same multicast probe received by different destination nodes have the same network experience (loss, delay, etc.) in the shared links, which may not hold for a unicast packet pair/string probe. Can we still construct additive metrics from unicast probing? The answer is yes, if the packets are positively correlated (not necessarily perfect correlated) in the shared links.

Suppose the source node s sends two back-to-back packets to destination nodes i and j , for which the first packet (denoted by a) is sent to node i and the second packet (denoted by b) is sent to node j . Let Z_e^a and Z_e^b be the link state variables experienced by packet a and packet b in link e , respectively.

First consider link-loss (or utilization) inference. Let $\alpha_e = \mathbb{P}(Z_e^x = 1)$ for $x = \{a, b\}$ be the *marginal* success rate of link e . Let $\beta_e = \mathbb{P}(Z_e^b = 1 | Z_e^a = 1)$ be the *conditional* success rate of link e , i.e., β_e is the conditional probability of the second packet

b successfully goes through link e given that the first packet a successfully goes through link e .

If $0 < \alpha_e < \beta_e \leq 1$ for all links, then $0 < \frac{\alpha_e}{\beta_e} < 1$, and we can construct an additive metric d_l^i with link length

$$d_l^i(e) = -\log \frac{\alpha_e}{\beta_e}, \quad \forall e \in E.$$

In real networks, we would expect $\alpha_e < \beta_e$ because the fact that the first packet successfully goes through a link indicates that the link is in good state and the second packet, which closely follows the first packet, can also go through the link. This phenomenon was observed in real Internet measurements (e.g., [4] and [28]).

Let L_i^a and L_j^b be the loss outcome variable of packet a and b at node i and j , respectively. Under the spatial independence assumption, we have

$$\begin{aligned}\mathbb{P}(L_i^a = 1) &= \prod_{e \in \mathcal{P}(s, i)} \alpha_e, \\ \mathbb{P}(L_j^b = 1) &= \prod_{e \in \mathcal{P}(s, j)} \alpha_e \\ \mathbb{P}(L_i^a L_j^b = 1) &= \prod_{e \in \mathcal{P}(s, ij)} \alpha_e \beta_e \prod_{e \in \mathcal{P}(ij, i)} \alpha_e \prod_{e \in \mathcal{P}(ij, j)} \alpha_e.\end{aligned}$$

Hence, $\rho_l^i(s, D) \cup \rho_l^j(s, D^2)$ can be obtained by

$$\begin{aligned}\rho_l^i(i) &= -\log \frac{\mathbb{P}(L_i^a = 1)\mathbb{P}(L_i^b = 1)}{\mathbb{P}(L_i^a L_i^b = 1)}, \quad i \in D \\ \rho_l^i(i, j) &= -\log \frac{\mathbb{P}(L_i^a = 1)\mathbb{P}(L_j^b = 1)}{\mathbb{P}(L_i^a L_j^b = 1)}, \quad i, j \in D.\end{aligned} \quad (14)$$

Now consider link delay inference. If $\text{cov}(Z_e^a, Z_e^b) > 0$ for all links (which we would expect to hold in real networks because the two back-to-back packets are very close, hence, their experienced delays in a shared link are positively correlated), then we can construct an additive metric d_v^i with link length

$$d_v^i(e) = \text{cov}(Z_e^a, Z_e^b), \quad \forall e \in E.$$

Let T_i^a and T_j^b be the delay outcome variable of packet a and b at node i and j , respectively. We have $T_i^a = \sum_{e \in \mathcal{P}(s, i)} Z_e^a$, $T_j^b = \sum_{e \in \mathcal{P}(s, j)} Z_e^b$.

Under the spatial independence assumption, we have

$$\begin{aligned}\text{cov}(T_i^a, T_j^b) &= \sum_{e \in \mathcal{P}(s, ij)} \text{cov}(Z_e^a, Z_e^b) \\ \text{cov}(T_i^a, T_i^b) &= \sum_{e \in \mathcal{P}(s, i)} \text{cov}(Z_e^a, Z_e^b).\end{aligned}$$

Hence, $\rho_v^i(s, D) \cup \rho_v^j(s, D^2)$ can be obtained by

$$\begin{aligned}\rho_v^i(i) &= \text{cov}(T_i^a, T_i^b), \quad i \in D \\ \rho_v^i(i, j) &= \text{cov}(T_i^a, T_j^b), \quad i, j \in D.\end{aligned} \quad (15)$$

Similarly, as in (11)–(13), we can construct explicit estimators for the path lengths and shared path lengths in (14) and (15) using empirical distributions measured by the terminal nodes.

C. Additive Metric Based on Traceroute-Like Probing

Using traceroute-like probing, a source node can obtain the unique labels (IP addresses) of the internal nodes (routers) in the path from it to any destination node. We can construct an additive metric d_h by defining the *link length* $d_h(e)$ to be the number of hops (physical links) contained in logical link e .

The *path length* $\rho_h(i)$ is the number of hops contained in the path from s to i . The *shared path length* $\rho_h(i, j)$ is the number of hops contained in the shared portion of the paths from s to i and j . The shared portion of two paths can be determined by comparing the labels of the internal nodes in the two paths.

If some internal nodes do not respond to traceroute-like probing (e.g., anonymous routers, layer-2 switches, MPLS switches), then the derived path lengths and shared path lengths can be distorted. We use $\hat{\rho}_h(s, D)$ and $\hat{\rho}_h(s, D^2)$ to denote the estimated path lengths and shared path lengths with possible measurement errors.

V. TREE TOPOLOGY INFERENCE BASED ON NEIGHBOR JOINING

We first present a topology inference algorithm using (estimated) path lengths and shared path lengths as the input. The algorithm is a grouping type algorithm as in [15] and [22]. It can be viewed as a rooted version of the widely used *neighbor-joining* algorithm for constructing phylogenetic trees from distances [17], [23]. The algorithm begins with a leaf set including all the destination nodes. In each step, it selects a group of nodes that are likely to be *neighbors* (i.e., *siblings*, nodes with the same parent on the tree), deletes them from the leaf set, creates a new node as their parent, and adds that node to the leaf set. The whole process is iterated until there is only one node left in the leaf set, which will be the child of the root (source node). To avoid trivial cases, we assume $|D| \geq 2$.

Algorithm 1: Rooted Neighbor-Joining (RNJ) Algorithm

Input: Source s , Destinations D , $\hat{\rho}(s, D)$, $\hat{\rho}(s, D^2)$, $\Delta > 0$.

1. $V = \{s\} \cup D$, $E = \emptyset$.
- 2.1 Find $i^*, j^* \in D$ with the largest $\hat{\rho}(i, j)$ (break the tie arbitrarily). Create a node f as the parent of i^* and j^* .
 $D = D \setminus \{i^*, j^*\}$,
 $V = V \cup \{f\}$,
 $E = E \cup \{(f, i^*), (f, j^*)\}$.
 (+) $\hat{d}(f, i^*) = \hat{\rho}(i^*) - \hat{\rho}(i^*, j^*)$,
 (+) $\hat{d}(f, j^*) = \hat{\rho}(j^*) - \hat{\rho}(i^*, j^*)$.
- 2.2 For every $k \in D$ such that $\hat{\rho}(i^*, j^*) - \hat{\rho}(i^*, k) \leq \frac{\Delta}{2}$:
 $D = D \setminus k$,
 $E = E \cup (f, k)$.
 (+) $\hat{d}(f, k) = \hat{\rho}(k) - \hat{\rho}(i^*, j^*)$.
- 2.3 For each $k \in D$, compute:

$$\hat{\rho}(k, f) = \frac{1}{2}[\hat{\rho}(k, i^*) + \hat{\rho}(k, j^*)]. \quad (16)$$

$D = D \cup f$.

(+) $\hat{\rho}(f) = \hat{\rho}(i^*, j^*)$.

3. If $|D| = 1$, for the $k \in D$: $E = E \cup (s, k)$.

Otherwise, repeat Step 2.

Output: Tree $\hat{T} = (V, E)$, and link length $\hat{d}(e)$ for all $e \in E$.

Note that in (16) of step 2.3, we compute the shared path length between nodes k and f , $\hat{\rho}(k, f)$, using measurements from only two children of f . If f has more than two children, we could utilize measurements from all of them as follows:

$$\hat{\rho}(k, f) = \frac{1}{|c(f)|} \sum_{i \in c(f)} \hat{\rho}(k, i). \quad (17)$$

This modification improves the accuracy of the RNJ algorithm in our simulation.²

The computational complexity of the RNJ algorithm is $O(N^2 \log N)$ for a routing tree with N destination nodes. Note that the RNJ algorithm only requires (estimated) shared path lengths $\hat{\rho}(s, D^2)$ to infer the tree topology (steps without (+)). If the (estimated) path lengths $\hat{\rho}(s, D)$ are also available, then the RNJ algorithm can infer the link lengths as well (steps with (+)). If there is a one-to-one mapping between the link performance parameters (e.g., success rate, utilization, delay variance) and the link lengths, as in (5), (7), and (9), then we can use the link lengths returned by the RNJ algorithm to estimate the link performance parameters.

A. Analysis of RNJ Algorithm

Let T be the true topology of the routing tree. Let $d(e)$ s be the true link lengths and $\rho(s, D^2)$ be the true shared path lengths under additive metric d on T .

Proposition 1: Let $\Delta \leq \min_{e \in E} d(e)$ (the minimum link length on the routing tree) be the input parameter. A sufficient condition for the RNJ algorithm to return the correct tree topology is

$$|\hat{\rho}(i, j) - \rho(i, j)| < \frac{\Delta}{4}, \quad \forall i, j \in D. \quad (18)$$

Proof: We prove by induction on the cardinality of D .

- 1) If $|D| = 2$, then clearly the RNJ algorithm will return the correct tree topology.
- 2) Assume the RNJ algorithm returns correct topology under condition (18) for $|D| \leq N$. Now consider $|D| = N + 1$.

Claim 1— i^, j^* Found in Step 2.1, Which Maximize $\hat{\rho}(i, j)$, Are Siblings:* If i^* and j^* are not siblings, then $\exists k \in D$ such that either $\underline{i^*k}$ or $\underline{j^*k}$ is descended from $\underline{i^*j^*}$. Without loss of generality, suppose $\underline{i^*k}$ is descended from $\underline{i^*j^*}$. This implies $\rho(i^*, k) > \rho(i^*, j^*)$. Since link lengths $\geq \Delta$, we have $\rho(i^*, k) \geq \rho(i^*, j^*) + \Delta$. Then, under condition (18)

$$\hat{\rho}(i^*, k) > \rho(i^*, k) - \frac{\Delta}{4} > \rho(i^*, j^*) + \frac{\Delta}{4} > \hat{\rho}(i^*, j^*)$$

a contradiction to the maximality of $\hat{\rho}(i^*, j^*)$.

Claim 2— k Will Be Selected in Step 2.2 if and Only if it is a Sibling of i^ and j^* :* If k is a sibling of i^* and j^* , then $\rho(i^*, j^*) = \rho(i^*, k)$. This, together with condition (18), implies $\hat{\rho}(i^*, j^*) - \hat{\rho}(i^*, k) < \frac{\Delta}{2}$. Hence, k will be selected in step 2.2.

If k is not a sibling of i^* and j^* , and since i^* and j^* are siblings, then $\underline{i^*j^*}$ is descended from $\underline{i^*k}$. Since link lengths

²We thank an anonymous reviewer for this suggestion.

$\geq \Delta$, we have $\rho(i^*, j^*) \geq \rho(i^*, k) + \Delta$. Then, under condition (18)

$$\hat{\rho}(i^*, j^*) > \rho(i^*, j^*) - \frac{\Delta}{4} \geq \rho(i^*, k) + \frac{3\Delta}{4} > \hat{\rho}(i^*, k) + \frac{\Delta}{2}$$

which implies k will not be selected in step 2.2.

Claim 3—Condition (18) Is Maintained After Step 2.3: We have $|\hat{\rho}(k, i^*) - \rho(k, i^*)| < \frac{\Delta}{4}$ and $|\hat{\rho}(k, j^*) - \rho(k, j^*)| < \frac{\Delta}{4}$. Since $\hat{\rho}(k, f) = \frac{1}{2}(\hat{\rho}(k, i^*) + \hat{\rho}(k, j^*))$, $\rho(k, f) = \frac{1}{2}(\rho(k, i^*) + \rho(k, j^*))$, by triangular inequality, we have $|\hat{\rho}(k, f) - \rho(k, f)| < \frac{\Delta}{4}$.

From claims 1–3, after one iteration of step 2, the RNJ algorithm will correctly find out a pair of siblings and all their other siblings (if any), and condition (18) is maintained for the new set of leaf nodes. Then $|D|$ is decreased at least by one. By induction assumption, the algorithm will return the correct topology of the rest of the tree. This completes our proof of the proposition. ■

Therefore, if the estimated shared path lengths are close enough to the true values, the RNJ algorithm will return the correct tree topology. We can derive exponential error bounds for the shared path length estimators in (11) and (12) using Chernoff bounds [20].

Proposition 2: For any pair of nodes $i, j \in D$, a sample size of n (number of probes to estimate $\hat{\rho}_l$ or $\hat{\rho}_u$), and any small $\epsilon > 0$

$$\mathbb{P}\{|\hat{\rho}_l(i, j) - \rho_l(i, j)| \geq \epsilon\} \leq e^{-c_{ij}(\epsilon)n} \quad (19)$$

$$\mathbb{P}\{|\hat{\rho}_u(i, j) - \rho_u(i, j)| \geq \epsilon\} \leq e^{-b_{ij}(\epsilon)n} \quad (20)$$

where $c_{ij}(\epsilon)$ s and $b_{ij}(\epsilon)$ s are some constants.

Let \hat{T}_n be the inferred tree topology returned by the RNJ algorithm with a sample size n . Let $P_n = \mathbb{P}\{\hat{T}_n = T\}$ denote the probability of correct topology inference of the RNJ algorithm.

Proposition 3: Let $\Delta \leq \min_{e \in E} d(e)$ be the input parameter of the RNJ algorithm. If

$$\mathbb{P}\left\{|\hat{\rho}(i, j) - \rho(i, j)| \geq \frac{\Delta}{4}\right\} \leq e^{-c_{ij}(\Delta)n}, \quad \forall i, j \in D$$

where n is the sample size and $c_{ij}(\Delta)$ is some constant. Then, for a routing tree with N destination nodes

$$P_n \geq 1 - N^2 e^{-c(\Delta)n} \quad (21)$$

i.e., the probability of correct topology inference of the RNJ algorithm converges to one exponentially fast in the sample size.

Proof: By Proposition 1 and union bound, we have

$$\begin{aligned} P_n &\geq \mathbb{P}\left\{\bigcap_{i, j \in D} |\hat{\rho}(i, j) - \rho(i, j)| < \frac{\Delta}{4}\right\} \\ &= 1 - \mathbb{P}\left\{\bigcup_{i, j \in D} |\hat{\rho}(i, j) - \rho(i, j)| \geq \frac{\Delta}{4}\right\} \\ &\geq 1 - \sum_{i, j \in D} e^{-c_{ij}(\Delta)n} \geq 1 - N^2 e^{-c(\Delta)n} \end{aligned}$$

where $c(\Delta) = \min_{i, j \in D} c_{ij}(\Delta)$. ■

B. Comparison With Previous Grouping Algorithms

The grouping algorithms in [15] and [22] aggregate the measurement data from the destination nodes up the tree, which are particularly designed for multicast probing. In contrast, the RNJ algorithm only requires (estimated) shared path lengths between pairs of the destination nodes, which is applicable to both multicast probing and unicast packet pair probing.

Under multicast probing, for general (nonbinary) routing trees, the RNJ algorithm has a much lower computational complexity, while it may also require a larger sample size to achieve the same level of accuracy compared to the maximum-likelihood based grouping algorithm in [15]. Nevertheless, we have shown that the probability of correct topology inference of the RNJ algorithm converges to one exponentially fast in the sample size.

We compare the accuracy of the RNJ algorithm with the BLTP algorithm (the reference grouping algorithm in [15], which has best accuracy and complexity) via model simulation. For each experiment, we first randomly generate the tree topology and select the link-loss rates in a certain range. We compare the inferred tree topology returned by RNJ and BLTP with the true tree topology. Each experiment is repeated 200 times. For each inference algorithm, we compute the *fraction* of correctly inferred trees among all 200 trials (which can be viewed as the *probability* of correct topology inference of the algorithm).

The results are shown in Figs. 2 and 3. The x axis is in log scale, i.e., it is $\log_2 n$ for a sample size of n probes. Both the RNJ algorithm and the BLTP algorithm are *consistent*: the fraction of correctly inferred trees of both algorithms goes to one exponentially fast as we increase the sample size. When the link-loss rates are within the range of [1%, 10%], the BLTP algorithm has a noticeably better accuracy than the RNJ algorithm while, when the link-loss rates are within the range of [0.1%, 1%], the difference is small (this is consistent with our analysis in [20], where we show that the simple estimator for shared path lengths (11) is as efficient as the maximum likelihood estimator for networks with small loss rates). We conduct experiments for trees with different sizes and ranges of link-loss rates and observe the same pattern of the results.

VI. DYNAMIC TREE TOPOLOGY INFERENCE

In practice, the RNJ algorithm (and other existing topology inference algorithms) may have some limitations. First, the focus of previous studies is on a relatively stable set of nodes. In real applications (e.g., P2P applications), the destination nodes that a source node communicates with will often change over time. Hence, the routing tree topology will also change over time. When an existing destination node leaves, it is straightforward to derive the updated routing tree topology. When a new destination node joins, running the RNJ algorithm over the new set of destination nodes to infer the updated routing tree topology is not efficient when the nodes join and leave frequently.

The second limitation is the *probing scalability problem* under unicast probing. The RNJ algorithm requires estimated shared path lengths from the source node to all pairs of the

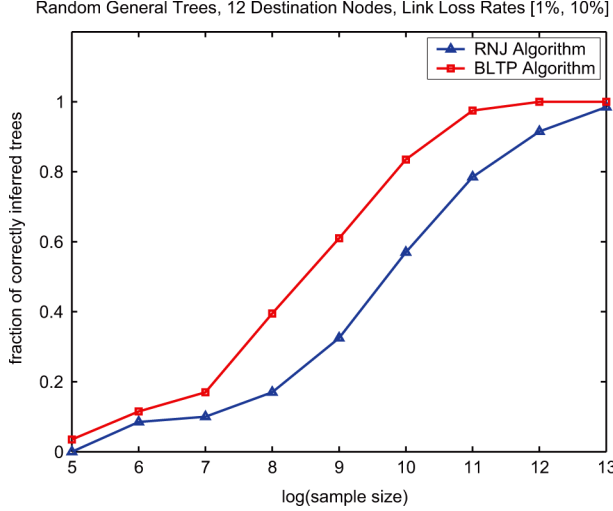


Fig. 2. Comparison of RNJ and BLTP under moderate link-loss rates.

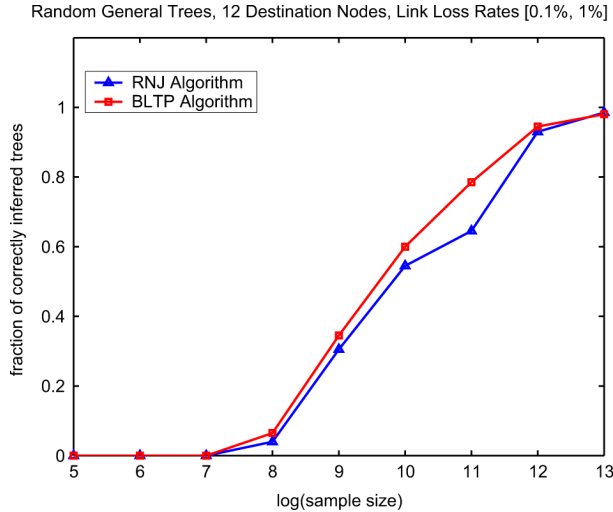


Fig. 3. Comparison of RNJ and BLTP under small link-loss rates.

destination nodes as the input. Suppose there are N destination nodes. If multicast probing is available, then the source node can use a $1 \times N$ multicast probing to obtain the required measurements. The probing overhead is $O(N)$. On the other hand, if multicast probing is not supported and N is large, then it is difficult to obtain $\hat{\rho}(s, D^2)$ using a single $1 \times N$ unicast packet string probing without violating the assumption that the string of packets has the same or even positively correlated network experiences in the shared links.

The source node could use back-to-back (unicast) packet pair probings. This requires $O(N^2)$ 1×2 probings. The probing overhead is $O(N^2)$. If these probings are conducted in parallel, then this will quickly consume the outgoing bandwidth of the source node; while if these probings are conducted in sequence, then it will take a long time to obtain the measurements, and it is likely that the network states (routing topology, link performance metrics) will change during the measurement period, which will violate the stationarity assumption. We tested the RNJ algorithm via Internet experiments, and we found that it only has decent accuracy for a small number of destination

nodes (less than six). Therefore, poor *probing scalability* of unicast packet pair probing will limit the number of destination nodes that a source node can infer when multicast probing is not supported.

We address these issues in this section. We design procedures to add a node to (`add_node`) and delete a node from (`delete_node`) a routing tree. These procedures can handle node joining and leaving efficiently and are particularly useful for applications where node dynamics are prevalent. Based on the `add_node` procedure, we propose a novel sequential topology inference algorithm, which greatly reduces the probing overhead under unicast packet pair probing.

A. Procedure `add_node`

`add_node`(T, k, j, Δ) is a recursive procedure that adds a new destination node j to the routing tree $T = (V, E)$ via an existing node k on the tree, with the initial condition that j is a sibling or descendant of node k . Δ is the (estimated) minimum link length. Let $f(k)$ be the parent of k on the (old) tree T .

Procedure: `add_node`(T, k, j, Δ)

IF k is a leaf node on the tree $T = (V, E)$:

(j will be a sibling of k on the new tree.)

1. Create a node p as the parent of k and j .

$$V = V \cup \{p, j\},$$

$$E = E \setminus (f(k), k) \cup \{(f(k), p), (p, k), (p, j)\}.$$

ELSE **Suppose** k has l children c_1, \dots, c_l .

2. Select a destination node d_i descended from c_i .

3. Measure/estimate $\hat{\rho}(d_1, d_2)$ and $\hat{\rho}(j, d_i)$ for $i = 1, \dots, l$.

4. Find d_{i^*} with the largest $\hat{\rho}(j, d_i)$.

Case (a) $\hat{\rho}(d_1, d_2) - \hat{\rho}(j, d_{i^*}) \geq \frac{\Delta}{2}$:

(j will be a sibling of k on the new tree.)

5. Create a node p as the parent of k and j .

$$V = V \cup \{p, j\},$$

$$E = E \setminus (f(k), k) \cup \{(f(k), p), (p, k), (p, j)\}.$$

Case (b) $|\hat{\rho}(d_1, d_2) - \hat{\rho}(j, d_{i^*})| < \frac{\Delta}{2}$:

(j will be a child of k on the new tree.)

6. $V = V \cup j, E = E \cup (k, j)$.

Case (c) $\hat{\rho}(j, d_{i^*}) - \hat{\rho}(d_1, d_2) \geq \frac{\Delta}{2}$:

(j will be a sibling or descendant of c_{i^*} on the new tree.)

7. Execute `add_node`(T, c_{i^*}, j, Δ).

By running `add_node`(T, s, j, Δ), we add a new destination node j to the routing tree T rooted at s .

In step 3 of `add_node`(T, k, j, Δ), in order to estimate the shared path lengths $\hat{\rho}(d_1, d_2)$ and $\hat{\rho}(j, d_i)$ for $i = 1, \dots, l$, s can use a $1 \times (l + 1)$ (multicast) probing by sending probes to destination nodes j, d_1, \dots, d_l ; alternatively, s can use $l + 1$ (unicast) packet pair probings by sending probes to node pairs $(d_1, d_2), (j, d_1), \dots, (j, d_l)$.

For an l -ary (balanced) tree with N destination nodes, the depth of the tree is $O(\log_l N)$. In the worst case, the `add_node` procedure needs to be executed $O(\log_l N)$ times in order to add a new destination node to the tree. Under unicast packet pair probing, if we apply the `add_node` procedure to infer the

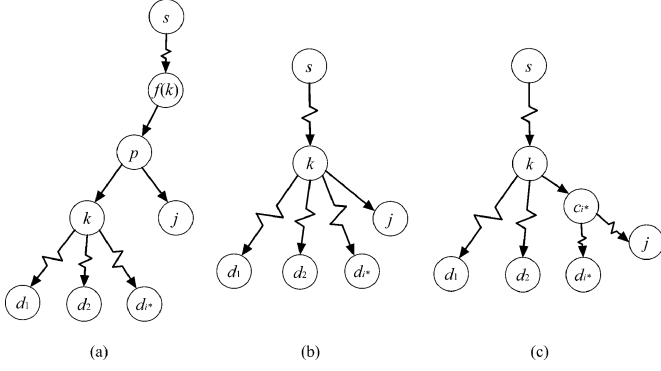


Fig. 4. Three cases of adding a new node j to the tree via a node k on the tree. (a) j is a sibling of k . (b) j is a child of k . (c) j is a sibling or descendant of c_{i^*} .

topology of the new tree, we need $O(l \log_l N)$ packet pair probings, and the computational complexity is $O(l \log_l N)$. If we apply the RNJ algorithm to infer the topology of the new tree, we need $O(N^2)$ packet pair probings, and the computational complexity is $O(N^2 \log N)$.

B. Analysis of Procedure `add_node`

If the estimated shared path lengths in step 3 are close enough to the true values, then `add_node`(T, s, j, Δ) will correctly add a new destination node to the tree.

Proposition 4: Let Δ be less than or equal to the minimum link length in the new routing tree (including existing destination nodes and the new destination node j). A sufficient condition for the recursive procedure `add_node`(T, s, j, Δ) to return the correct tree topology (after adding node j) is that for all the nodes k visited by the recursive procedure

$$\begin{aligned} |\hat{\rho}(d_1, d_2) - \rho(d_1, d_2)| &< \frac{\Delta}{4} \\ |\hat{\rho}(j, d_i) - \rho(j, d_i)| &< \frac{\Delta}{4}, \quad i = 1, 2, \dots, l. \end{aligned} \quad (22)$$

Proof: We prove that if k is a sibling or ancestor of j in the new routing tree and condition (22) is satisfied, then procedure `add_node`(T, k, j, Δ) will either correctly add j to the tree or find a child c of k that is a sibling or ancestor of j and execute `add_node`(T, c, j, Δ) recursively. Hence, `add_node`(T, s, j, Δ) finally returns the correct new routing tree topology.

Now assume k is a sibling or ancestor of j and condition (22) is satisfied. If k is a leaf node (i.e., k is a destination node), then k and j must be siblings so step 1 of `add_node`(T, k, j, Δ) correctly adds j to the tree. Otherwise, suppose k has l children c_1, \dots, c_l , and d_i is a destination node descended from c_i selected in step 2. In step 3, $\hat{\rho}(d_1, d_2)$ and $\hat{\rho}(j, d_i)$ for $i = 1, \dots, l$ are measured and estimated. There are three cases to consider.

- j is a sibling of k in the new routing tree, as shown in Fig. 4(a). In this case, for the d_{i^*} found in step 4, we have $\rho(d_1, d_2) - \rho(j, d_{i^*}) \geq \Delta$. Under (22), this implies $\hat{\rho}(d_1, d_2) - \hat{\rho}(j, d_{i^*}) > \frac{\Delta}{2}$, so step 5 will be executed, which correctly adds j to the tree.
- j is a child of k , as shown in Fig. 4(b). In this case, $\rho(d_1, d_2) = \rho(j, d_{i^*})$. Under (22), this implies

$|\hat{\rho}(d_1, d_2) - \hat{\rho}(j, d_{i^*})| < \frac{\Delta}{2}$, so step 6 will be executed, which correctly adds j to the tree.

- j is a sibling or descendant of a child of k , as shown in Fig. 4(c). Suppose c_{i^*} is the child and d_{i^*} is the selected destination node descended from c_{i^*} in step 2. Then $\rho(j, d_{i^*}) - \rho(j, d_{i'}) \geq \Delta$ for $i' \neq i^*$ and $\rho(j, d_{i^*}) - \rho(d_1, d_2) \geq \Delta$. Under (22), this implies $\hat{\rho}(j, d_{i^*}) > \hat{\rho}(j, d_{i'})$, so d_{i^*} will be selected in step 4 and $\hat{\rho}(j, d_{i^*}) - \hat{\rho}(d_1, d_2) > \frac{\Delta}{2}$; hence, `add_node`(T, c_{i^*}, j, Δ) will be executed in step 7. \blacksquare

Proposition 5: Let Δ be less than or equal to the minimum link length in the new routing tree. If, for all the nodes k visited by the recursive procedure `add_node`(T, s, j, Δ), we have

$$\begin{aligned} \mathbb{P} \left\{ |\hat{\rho}(d_1, d_2) - \rho(d_1, d_2)| \geq \frac{\Delta}{4} \right\} &\leq e^{-c_{d_1 d_2}(\Delta)n} \\ \mathbb{P} \left\{ |\hat{\rho}(j, d_i) - \rho(j, d_i)| \geq \frac{\Delta}{4} \right\} &\leq e^{-c_{j d_i}(\Delta)n}, \\ &i = 1, \dots, l \end{aligned}$$

where n is the sample size and $c_{d_1 d_2}(\Delta)$, $c_{j d_i}(\Delta)$ s are some constants, then the probability of correct topology inference of `add_node`(T, s, j, Δ) for an l -ary tree with N destination nodes satisfies

$$P_n \geq 1 - (l+1)(\log_l N)e^{-c(\Delta)n}. \quad (23)$$

Proof: The proof is similar to the proof of Proposition 3. \blacksquare

C. Procedure `delete_node`

Procedure `delete_node`(T, j) deletes a destination node j from routing tree T . It will first remove node j and link $(f(j), j)$ from the tree. If $f(j)$ has only one child left after deleting j , it will then further remove node $f(j)$ and connect the child of $f(j)$ to the parent of $f(j)$, so that the new routing tree maintains the property that each internal node has at least two children.

Procedure: `delete_node`(T, j)

- $V = V \setminus j$, $E = E \setminus (f(j), j)$.
- If $f(j)$ has only one child c left:

$$V = V \setminus f(j)$$

$$E = E \setminus \left\{ (f(f(j)), f(j)), (f(j), c) \right\} \cup (f(f(j)), c).$$

D. Sequential Topology Inference Algorithm

For a source node s and a set of destination nodes D , we can apply the `add_node` procedure over the nodes in D in sequence to construct the routing tree topology incrementally, as described in Algorithm 2.

We compare the RNJ algorithm and the sequential topology inference algorithm in Table I. We assume all probings have the same sample size and time interval between two consecutive probes. Under multicast probing, the RNJ algorithm is more efficient (for building the whole tree); while under unicast packet pair probing, the sequential topology inference algorithm is more efficient, in terms of the probing traffic and probing time. In both cases, the sequential topology inference

TABLE I
COMPARISON OF RNJ ALGORITHM AND SEQUENTIAL TOPOLOGY INFERENCE ALGORITHM

		N Destination Nodes, l -ary Tree with Depth $O(\log_l N)$				
		Multicast Probing		Unicast Packet Pair Probing		Computational Complexity
		Probing Traffic	Probing Time	Probing Traffic	Probing Time	
Add Node	RNJ	$O(N)$	$O(1)$	$O(N^2)$	$O(N^2)$	$O(N^2 \log N)$
	Sequential	$O(l \log_l N)$	$O(\log_l N)$	$O(l \log_l N)$	$O(l \log_l N)$	$O(l \log_l N)$
Build Tree	RNJ	$O(N)$	$O(1)$	$O(N^2)$	$O(N^2)$	$O(N^2 \log N)$
	Sequential	$O(Nl \log_l N)$	$O(N \log_l N)$	$O(Nl \log_l N)$	$O(Nl \log_l N)$	$O(Nl \log_l N)$

algorithm is more computationally efficient than the RNJ algorithm.

Algorithm 2: Sequential Topology Inference Algorithm

Input: Source Node s , Destination Nodes $D = \{1, 2, \dots, N\}$, $\Delta > 0$.

1. $V_0 = \{s\}$, $E_0 = \emptyset$, $T_0 = (V_0, E_0)$.

2. For $j = 1$ to N :

$T_j = \text{add_node}(T_{j-1}, s, j, \Delta)$.

Output: Tree $\hat{T} = T_N$.

VII. INTERNET ROUTING TREE TOPOLOGY INFERENCE

A. Schemes For Internet Routing Tree Topology Inference

In this section, we design schemes for Internet routing tree topology inference. We consider the following schemes.

1) *Traceroute-Based Inference Scheme (TR)*: We use traceroute measurements to construct additive metric d_h and derive the shared path lengths $\hat{\rho}_h(s, D^2)$, as we described in Section IV.

2) *Tomography-Based Inference Scheme (Tomo)*: We use unicast packet pair/string measurements to construct additive metrics d_l, d_u, d_v and estimate the shared path lengths $\hat{\rho}_l(s, D^2), \hat{\rho}_u(s, D^2), \hat{\rho}_v(s, D^2)$, as we described in Section IV. We construct a new additive metric using a convex combination of the additive metrics to fuse information from different measurements: $d_t = a_l d_l + a_u d_u + a_v d_v$ with $a_l + a_u + a_v = 1$.

We have shown that if the estimated shared path lengths are close enough to the true values [e.g., condition (18) or (22) is satisfied], then the RNJ algorithm and the sequential topology inference algorithm will return the correct routing tree topology.

For traceroute measurements, the estimated shared path lengths can be distorted due to the existence of anonymous routers, layer-2 switches, and MPLS switches. For network tomography measurements, the assumption of *independent* and *stationary* link states can be violated, so a larger sample size with longer measurement period may not return more accurate estimation of shared path lengths. Hence, the condition for correct topology inference (18) or (22) may not hold for both types of measurements.

In order to utilize information collected from both traceroute measurements and network tomography measurements, we propose the following *hybrid scheme* for Internet routing topology inference.

3) *Traceroute + Tomography Inference Scheme (TRTomo)*: We use both traceroute measurements and network tomography measurements to construct additive metrics d_h and d_t , respectively, and we construct a new additive metric $d_{ht} = A \cdot d_h + d_t$

with a large constant A , which makes traceroute measurements dominate network tomography measurements. The reason for selecting a large A is that traceroute measurements have certain “consistent” property. An anonymous router will affect all the paths passing that router (i.e., the path lengths of those paths are all reduced by one). Hence, if $\hat{\rho}_h(i, j) > \hat{\rho}_h(i, k)$, then we know for sure that ij is descended from ik on the routing tree. The reverse, however, may not be true. Even if ij is descended from ik , we may have $\hat{\rho}_h(i, j) = \hat{\rho}_h(i, k)$ due to anonymous routers; hence, network tomography measurements are needed to further determine the topology.

For a large number of destination nodes, we propose to infer the routing tree topology using a *two-step procedure*: first use traceroute measurements ($\hat{\rho}_h$) (or other heuristics, e.g., round trip times, AS information) to build a skeleton of the tree; then apply tomography measurements ($\hat{\rho}_t$ or $\hat{\rho}_{ht}$) on subtrees (with relatively a small number of destination nodes) to determine the topology of the subtrees. We find this hybrid approach significantly reduces the probing scalability problem of a pure tomography-based approach. It also leads to better accuracy than a pure traceroute-based approach or pure tomography-based approach via information fusion.

We refer to the above schemes as *TR*, *Tomo*, and *TRTomo* for short hereafter. We evaluate their performance via Internet experiments.

B. Evaluation Methodology

We choose an idle host in our local network as the source node and two sets of PlanetLab³ nodes as the destination nodes. We have implemented a *sender utility program* (running at the source node) that can send unicast probing packet pairs/strings and a *receiver utility program* (running at the destination nodes) to receive the probing packets and measure their one-way delays. We collect the measured one-way delays from the destination nodes using the sender utility program.

The first destination node set, referred to as *US nodes*, consists of 30 hosts in the United States (most of them are located at U.S. universities). The second set, referred to as *international nodes*, consists of 30 international nodes (ten in North America, ten in Europe, and ten in East Asia). The reliability of the chosen nodes is important to the experiments; hence, we choose nodes that have low CPU load and long running time.

Each probing from the source node to a subset of the destination nodes consists of a sequence of 1200 packet strings. Each probing packet is of size 80 bytes. The probing interval between two consecutive strings is 10 ms (contributing to a probing rate of 64 kbps per destination node).

³See <http://www.planet-lab.org>.

We evaluate the performance of the three topology inference schemes by artificially varying the *anonymization ratio*, which is the fraction of the underlying routers not responding to traceroute probing. For each anonymization ratio, we test the topology inference schemes for 20 rounds.

In each round, we first obtain the sequence of the underlying routers from the source node to every destination node using traceroute. The destination nodes we choose have the property that the paths from the source node to them contain no or very few anonymous routers, so we can obtain the *ground-truth topology* in order to test the topology inference schemes. We count the total number of unique routers we have seen for all destination nodes and compute how many of the routers in total should be anonymized according to the anonymization ratio. We then iteratively choose a destination node randomly and anonymize the last m routers along its route,⁴ where m is computed as the anonymization ratio times the route length. We also keep track of the number of unique routers we have anonymized in each iteration and terminate the anonymization procedure once the total number of unique anonymized routers reaches the number we compute a priori.

We use the following two metrics to evaluate the performance of the topology inference schemes.

- **Correctness ratio**, which is the fraction of the internal nodes in the ground-truth topology that are correctly inferred averaged over all rounds. An internal node in the ground-truth topology is *correctly inferred* if and only if there is an internal node in the inferred topology with the same set of destination nodes descending from it. A higher correctness ratio means better accuracy of the inference scheme.
- **Node ratio**, which is the ratio of the number of internal nodes in the inferred topology to the number of internal nodes in the ground-truth topology, averaged over all rounds. An accurate inference scheme has a node ratio close to one. If the node ratio is larger than one (or less than one), then the inference algorithm returns more internal nodes (or less internal nodes) in the inferred topology.

C. Experimental Results

We run experiments using the US nodes and international nodes and refer to them as US experiments and international experiments, respectively. We plot the correctness ratios (Figs. 5 and 6) and node ratios (Figs. 7 and 8) of different schemes with varying levels of underlying routers being anonymized.

1) *Correctness Ratio*: As shown in Figs. 5 and 6, both TR and TRTomo can correctly infer most of the internal nodes in the ground-truth topology when the anonymization ratio is small. As the anonymization ratio increases, the correctness ratio of TR decreases to zero because TR heavily relies on routers' support for traceroute probing (note that it is not exactly zero because the source node must be attached to an access router and we

⁴When choosing the PlanetLab nodes, we find that a lot of them are behind routers that do not respond to traceroute probing. Most of these routers are edge routers or access routers of the network in which the destination nodes are located. This suggests that traceroute probings are likely to be discarded in enterprise networks to protect their internal hosts; hence, the routers in the last few hops to a destination node are more likely to be anonymous.

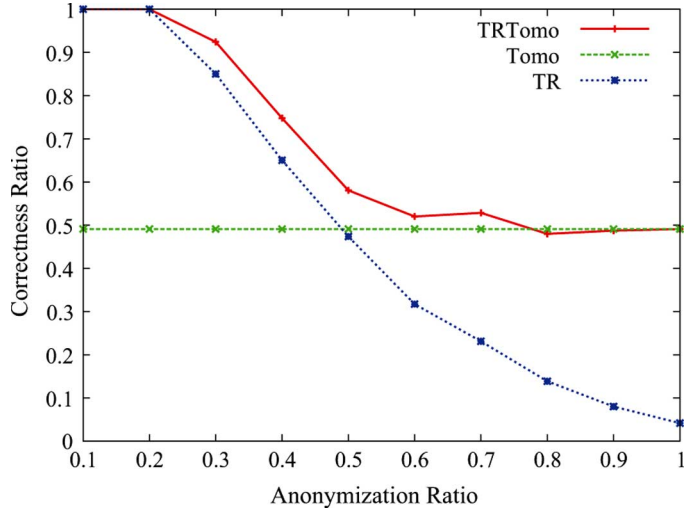


Fig. 5. US experiment: correctness ratio of inferred topology.

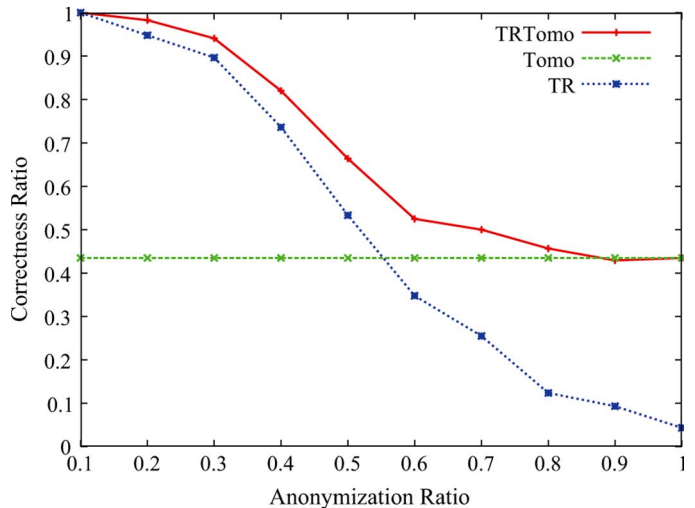


Fig. 6. International experiment: correctness ratio of inferred topology.

always include that router in the inferred routing tree topology). The correctness ratio of TRTomo stabilizes around 0.5 because TRTomo can improve TR's accuracy by utilizing both traceroute measurements and tomography measurements.

When the anonymization ratio is one (no routers respond to traceroute probing), TRTomo becomes the pure tomography-based scheme (Tomo), so we determine the correctness ratio of Tomo using the correctness ratio of TRTomo at anonymization ratio one, which is around 0.5.

From our experiences, we would like to comment on why the pure Tomo scheme alone can only infer about 50% of the internal nodes but cannot infer all the internal nodes in the ground-truth topology. First, the routing topology and link states may be *time-varying* instead of *stationary* during the measurement period. Secondly, there are several limitations of the PlanetLab testbed. We observed that the network connections from the source node to the PlanetLab nodes are pretty good most of the time; hence, the shared path lengths derived from loss and delay metrics (the *signals*) are quite small and can be easily distorted by measurement noises. In addition, most PlanetLab nodes are

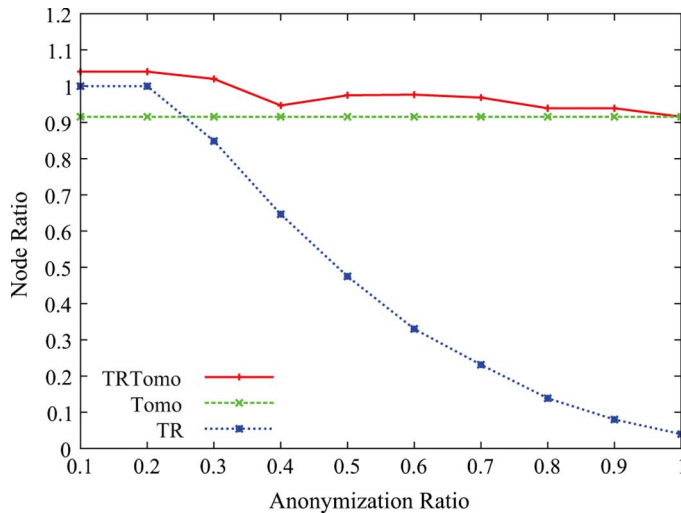


Fig. 7. US experiment: node ratio of inferred topology.

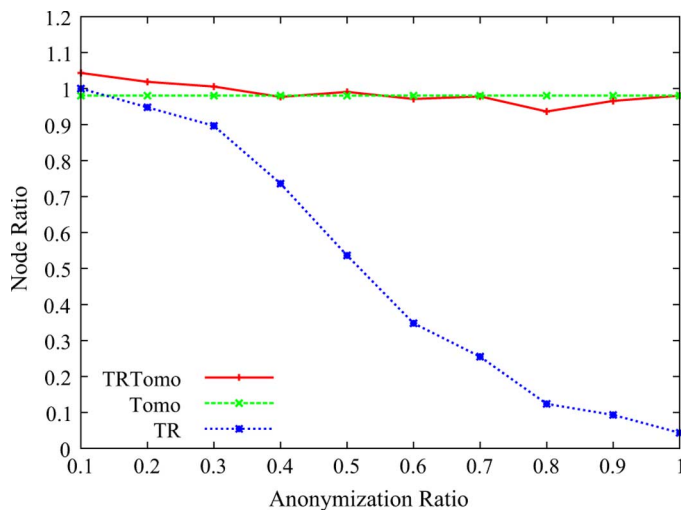


Fig. 8. International experiment: node ratio of inferred topology.

often running multiple applications and processes. This introduces nonnegligible node delays to the delay measurements, which will affect the delay and utilization measurements. (Such a phenomenon has been observed and addressed in [25].)

2) *Node Ratio*: As shown in Figs. 7 and 8, the node ratio of TR is close to one when the anonymization ratio is small, but it decreases to zero with an increasing anonymization ratio. In contrast, TRTomo has a node ratio close to one in all experiments regardless of anonymization ratio, although it may introduce a few more or less internal nodes in the inferred tree topology. The node ratio of Tomo is determined by the node ratio of TRTomo at anonymization ratio one.

VIII. CONCLUSION

In this paper, we developed fast and scalable algorithms for network routing tree topology inference using a framework based on additive metrics. In particular, we proposed a sequential topology inference algorithm to address the probing scalability problem and handle dynamic node joining and

leaving efficiently. We proved the correctness of our algorithms and demonstrated their effectiveness via Internet experiments. The proposed algorithms provide powerful tools for large-scale network inference in communication networks. In the future, we will study how to utilize the inferred information and extend the framework for efficient and effective network monitoring and application design.

ACKNOWLEDGMENT

The authors would like to thank Dr. N. Duffield and the anonymous reviewers for their helpful comments and suggestions.

REFERENCES

- [1] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris, "Resilient overlay networks," in *Proc. SOSP*, Banff, AB, Canada, Oct. 2001, pp. 131–145.
- [2] D. Antonova, A. Krishnamurthy, Z. Ma, and R. Sundaram, "Managing a portfolio of overlay paths," in *Proc. NOSSDAV*, Kinsale, Ireland, Jun. 2004, pp. 30–35.
- [3] A. Bestavros, J. Byers, and K. Harfoush, "Inference and labeling of metric-induced network topologies," in *Proc. IEEE INFOCOM*, New York, Jun. 2002, pp. 628–637.
- [4] J.-C. Bolot, "End-to-end packet delay and loss behavior in the Internet," in *Proc. ACM SIGCOMM*, San Francisco, CA, Sep. 1993, pp. 189–199.
- [5] P. Buneman, "The recovery of trees from measures of dissimilarity," in *Mathematics in the Archaeological and Historical Sciences*. Edinburgh, Scotland: Edinburgh Univ. Press, 1971, pp. 387–395.
- [6] R. Caceres, N. G. Duffield, J. Horowitz, and D. Towsley, "Multicast-based inference of network-internal loss characteristics," *IEEE Trans. Inf. Theory*, vol. 45, no. 7, pp. 2462–2480, Nov. 1999.
- [7] R. Castro, M. Coates, G. Liang, R. Nowak, and B. Yu, "Network tomography: Recent developments," *Statist. Sci.*, vol. 19, no. 3, pp. 499–517, 2004.
- [8] R. Castro, M. Coates, and R. Nowak, "Likelihood based hierarchical clustering," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2308–2321, Aug. 2004.
- [9] J. T. Chang, "Full reconstruction of Markov models on evolutionary trees: Identifiability and consistency," *Math. Biosci.*, vol. 137, pp. 51–73, 1996.
- [10] M. Coates and R. Nowak, "Network loss inference using unicast end-to-end measurement," in *Proc. ITC Conf. IP Traffic, Model. Manage.*, Monterey, CA, Sep. 2000, pp. 28-1–28-9.
- [11] M. Coates, A. O. Hero, III, R. Nowak, and B. Yu, "Internet tomography," *IEEE Signal Process. Mag.*, vol. 19, no. 3, pp. 47–65, May 2002.
- [12] M. Coates, R. Castro, M. Gadhio, R. King, Y. Tsang, and R. Nowak, "Maximum likelihood network topology identification from edge-based unicast measurements," in *Proc. ACM SIGMETRICS*, Jun. 2002, pp. 11–20.
- [13] N. G. Duffield, J. Horowitz, F. L. Presti, and D. Towsley, "Multicast topology inference from End-to-End measurements," *Adv. Perform. Anal.*, vol. 3, pp. 207–226, 2000.
- [14] N. G. Duffield, J. Horowitz, and F. L. Presti, "Adaptive multicast topology inference," in *Proc. IEEE INFOCOM*, Anchorage, AK, Apr. 2001, pp. 1636–1645.
- [15] N. G. Duffield, J. Horowitz, F. L. Presti, and D. Towsley, "Multicast topology inference from measured end-to-end loss," *IEEE Trans. Inf. Theory*, vol. 48, no. 1, pp. 26–45, Jan. 2002.
- [16] N. G. Duffield, F. Lo Presti, V. Paxson, and D. Towsley, "Network loss tomography using striped unicast probes," *IEEE/ACM Trans. Netw.*, vol. 14, no. 4, pp. 697–710, Aug. 2006.
- [17] O. Gascuel and M. Steel, "Neighbor-joining revealed," *Mol. Biol. Evol.*, vol. 23, no. 11, pp. 1997–2000, 2006.
- [18] J. Hartigan, *Clustering Algorithms*. New York: Wiley, 1975.
- [19] J. Ni and S. Tatikonda, "A Markov random field approach to multicast-based network inference problems," in *Proc. IEEE ISIT*, Seattle, WA, Jul. 2006, pp. 2769–2773.
- [20] J. Ni and S. Tatikonda, "Explicit link parameter estimators based on end-to-end measurements," in *Proc. Allerton Conf. Commun., Contr., Comput.*, Monticello, IL, Sep. 2007, pp. 174–181.

- [21] F. L. Presti, N. G. Duffield, J. Horowitz, and D. Towsley, "Multicast-based inference of network-internal delay distributions," *IEEE/ACM Trans. Netw.*, vol. 10, no. 6, pp. 761–775, Dec. 2002.
- [22] S. Ratnasamy and S. McCanne, "Inference of multicast routing trees and bottleneck bandwidths using end-to-end measurements," in *Proc. IEEE INFOCOM*, Mar. 1999, pp. 353–360.
- [23] N. Saitou and M. Nei, "The neighbor-joining method: A new method for reconstruction of phylogenetic trees," *Mol. Biol. Evol.*, vol. 4, no. 4, pp. 406–425, 1987.
- [24] M. Shih and A. O. Hero, III, "Hierarchical inference of unicast network topologies based on End-to-End measurements," *IEEE Trans. Signal Process.*, vol. 55, no. 5, pp. 1708–1718, May 2007.
- [25] J. Sommers and P. Barford, "An active measurement system for shared environments," in *Proc. ACM IMC*, San Diego, CA, Oct. 2007, pp. 303–314.
- [26] D. Stutzbach and R. Rejaie, "Understanding churn in peer-to-peer networks," in *Proc. ACM SIGCOMM Conf. Internet Meas.*, Rio de Janeiro, Brazil, 2006, pp. 189–202.
- [27] Y. Tsang, M. Coates, and R. Nowak, "Network delay tomography," *IEEE Trans. Signal Process.*, vol. 51, no. 8, pp. 2125–2136, Aug. 2003.
- [28] M. Jain, S. Moon, J. Kurose, and D. Towsley, "Measurement and modelling of the temporal dependence in packet loss," in *Proc. IEEE INFOCOM*, New York, Mar. 1999, pp. 345–352.
- [29] B. Yao, R. Viswanathan, F. Chang, and D. Waddington, "Topology inference in the presence of anonymous routers," in *Proc. IEEE INFOCOM*, San Francisco, CA, Apr. 2003, pp. 353–363.



Jian Ni (S'02–M'09) received the B.Eng. degree in automation from Tsinghua University, Beijing, China, in 2001, the M.Phil. degree in electrical and electronic engineering from the Hong Kong University of Science and Technology, Hong Kong SAR, China, in 2003, and the Ph.D. degree in electrical engineering from Yale University, New Haven, CT, in 2008.

He is currently a Post-Doctoral Researcher with the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign. His research interests include performance analysis, algorithm design, statistical inference and learning, and control and optimization of communication networks.



Haiyong Xie (S'05–M'09) received the B.S. degree from the University of Science and Technology of China, Hefei, China, and the Ph.D. degree in computer science from Yale University, New Haven, CT.

He proposed the idea of proactive provider participation in P2P (aka P4P, coordinating network providers and peer-to-peer applications), laid the theoretical foundations, designed and implemented the novel P4P network architecture, and led and conducted original research and large-scale tests on P4P. Encouraged by and based upon his research

and results on P4P, the P4P Working Group (P4PWG) was formed to promote academic studies and industrial adoptions of P4P. He was the Principal Researcher for the P4PWG and Distributed Computing Industry Association. He also has conducted research in general computer network areas including network traffic engineering, enterprise network traffic optimization, routing in overlay networks, and network equilibria.



Sekhar Tatikonda (S'92–M'00) received the Ph.D. degree in electrical engineering and computer science from Massachusetts Institute of Technology, Cambridge, in 2000.

He is currently an Associate Professor of electrical engineering in the Department of Electrical Engineering, Yale University, New Haven, CT. From 2000 to 2002, he was a Post-Doctoral Fellow in the Department of Computer Science, University of California, Berkeley. His current research interests include communication theory, information theory,

stochastic control, distributed estimation and control, statistical machine learning, and inference.



Yang Richard Yang (M'01) received the B.E. degree in computer science and technology from Tsinghua University, Beijing, China, in 1993, and the M.S. and Ph.D. degrees in computer science from the University of Texas at Austin in 1998 and 2001, respectively.

He is an Associate Professor of computer science and electrical engineering at Yale University, New Haven, CT. His current research interests include computer networks, mobile computing, wireless networking, sensor networks, and network security. He leads the Laboratory of Networked Systems at Yale.

He has served as a Committee Member of many conferences, as a Panelist of several funding agencies, and as an Advisor of several industrial and academic organizations. He is the Conference Cochair of IWQoS 2006. His research is supported by both government funding agencies such as the National Science Foundation (NSF) and industrial companies such as Microsoft.

Dr. Yang's recent awards include an NSF CAREER Award and a Schlumberger Foundation Award.