Network Tomography Based on Additive Metrics

Jian Ni, *Member, IEEE*, and Sekhar Tatikonda, *Member, IEEE*

***Abstract*—Network tomography studies the inference of network structure and dynamics based on indirect measurements when direct measurements are unavailable or difficult to collect. In this paper, we design and analyze routing tree topology and link performance inference algorithms for communication networks using tools from phylogenetic inference in evolutionary biology. We develop polynomial-time distance-based inference algorithms and derive sufficient conditions for the correctness of the algorithms. We show that the algorithms are consistent and robust. In particular, the algorithms achieve the optimal l_∞-radius 1/2 for binary trees and 1/4 for general trees when a threshold neighbor selection criterion is used.**

***Index Terms*—Link performance estimation, neighbor-joining, network tomography, phylogenetic inference, routing topology inference.**

I. Introduction

NETWORK tomography (network inference) [6], [10], [31] studies the inference and learning of network structure and dynamics (e.g., routing topology, link performance, traffic matrices) of communication networks based on *indirect* measurements when *direct* measurements are unavailable or difficult to collect. As modern communication networks continue to grow in size and complexity, scalable and accurate network inference algorithms and techniques are becoming increasingly important for many network design and optimization tasks. These include service provision and resource allocation, traffic engineering, network monitoring, and application design.

Several types of network inference problems have been investigated recently: (1) link performance estimation based on end-to-end traffic measurements (e.g., [5], [9], [22], [24], [30]); (2) network routing topology inference based on end-to-end traffic measurements (e.g., [3], [11], [12], [23], [25], [28]); (3) source-destination traffic matrix estimation based on link-level measurements (e.g., [19], [21], [31]–[33]).

In this paper we focus on the first two types of network inference problems. A direct approach to measure the routing topology and link performance in a network is to use tools based on measurements or feedback messages of the internal nodes (e.g., routers). Such an approach, however, is limited as today's communication networks are evolving towards more decentralized and private adminstration. For example, *traceroute* is a tool to discover the intermediate routers from a source node to a destination node in the Internet (http://en.wikipedia.org/wiki/Traceroute). However, an increasing number of routers in the Internet will block traceroute probing messages due to privacy and security concerns. These routers, known as anonymous routers, cannot be discovered by traceroute type tools.

The *network tomography* approach, in contrast, utilizes end-to-end packet probing measurements (such as packet loss and delay measurements) conducted by the end hosts to infer the routing topology and link performance. Since it does not require extra cooperation of the internal nodes, the network tomography approach is more flexible and reliable. 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 routing topology and link performance from the source node to the destination nodes. Both multicast probing based approaches (e.g., [5], [12], [20], [24], [25]) and unicast probing based approaches (e.g., [9], [11], [13], [28], [30]) have been investigated. The main challenges of existing network tomography approaches include:

- *computational complexity*;
- *information fusion*: how to fuse information from different measurements to achieve the best estimation accuracy;
- *probing scalability*: how to reduce probing traffic, especially under unicast probing;
- *node dynamics*: how to conduct inference under dynamic node joining and leaving efficiently.

We found that the network inference problem is similar to the phylogenetic inference problem in evolutionary biology [20]. The phylogenetic inference problem is to determine the evolutionary relationship among a set of species. Such relationship is represented by a phylogenetic tree, in which the leaf nodes represent extant species and the internal nodes represent extinct common ancestors of the extant species. Many methods have been developed to reconstruct phylogenetic trees from biological information such as biomolecular sequence data observed at the leaf nodes [14], [26], [27]. The mathematical models of these two problems are quite similar: under certain assumptions both models are Markov random fields on trees [8], [20].

In this paper we use a framework based on *additive metrics* from phylogenetic inference to address the network inference problem in communication networks. Under an additive metric, the path metric (path length) is expressed as the summation of the link metrics (link lengths) of the links along the path. The basic idea is to use (estimated) distances between the terminal nodes (i.e., end hosts) to infer the routing tree topology and link metrics, where the distances incorporate the correlation information among the measurements at the end hosts. Since a linear combination of several additive metrics is still an additive metric, the framework can flexibly fuse information from multiple measurements to improve accuracy. Based on the frame-

Manuscript received August 17, 2008; revised November 07, 2010; accepted March 18, 2011. Date of current version December 07, 2011.

J. Ni is with the IBM T. J. Watson Research Center, Yorktown Heights, NY 10598 USA (e-mail: nij@us.ibm.com).

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

Communicated by S. Ulukus, Associate Editor for Communication Networks.

Digital Object Identifier 10.1109/TIT.2011.2168901

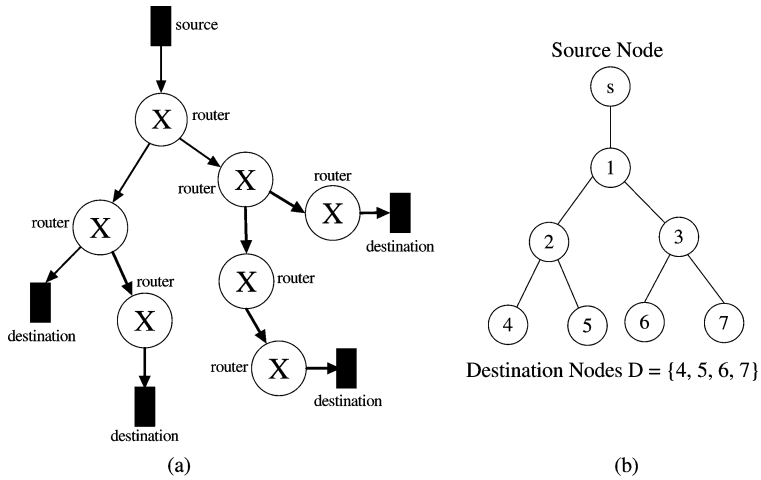

Fig. 1. Physical routing topology and the associated logical routing tree with a single source node and multiple destination nodes. (a) The physical routing topology. (b) The logical routing tree.

work, we have developed several distance-based inference algorithms to address the above mentioned challenges in [23]. In this paper, we provide rigorous analysis of the algorithms. We show that the algorithms are *consistent* (return correct topology and link performance with an increasing sample size) and *robust* (can tolerate a certain level of measurement errors). In particular, we establish certain optimality properties of the algorithms, i.e., they achieve the optimal l_∞-radius 1/2 for binary trees and 1/4 for general trees when a threshold neighbor selection criterion is used.

The rest of the paper is organized as follows. In Section II we describe the network model. In Section III we introduce additive metrics on trees, and we discuss how to construct additive metrics and compute/estimate the distances between the terminal nodes from end-to-end measurements. In Section IV we introduce the neighbor-joining (NJ) algorithm for constructing binary trees from distances. In Section V we present a rooted version of the NJ algorithm and extend it to general trees. In Section VI we apply our framework to infer the routing topology and link performance from multiple source nodes to a single destination node. We discuss the related work in Section VII and conclude our paper in Section VIII.

II. Network Model

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 connect 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 destination nodes is a (directed) tree. From the physical routing topology, we can construct 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 [5], [12], [25]. Notice that 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.

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 logical routing tree from s to nodes in D, with node set V and link set E. Let $U = s \cup D$[1] be the set of terminal nodes, which are nodes of degree one (e.g., end hosts). Every node $k \in V$ has a *parent* $f(k) \in V$ such that $(f(k), k) \in E$, 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 use e_k to denote link $(f(k), k)$. Each link $e \in E$ is associated with a performance parameter θ_e (e.g., success rate, delay distribution, utilization).

Notice that when the root of a directed tree is known, it is sufficient to infer the undirected version of the tree (the direction of any link can be recovered by the closeness of the two end points of the link to the root). For simplicity in this paper we use *routing tree* to express undirected logical routing tree (as shown in Fig. 1(b)) unless otherwise noted. We use $\mathcal{P}(i, j)$ to denote the (unique) path connecting two nodes i and j on the (logical) routing tree.

The network inference problem involves using measurements observed by the terminal nodes to infer:

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

III. Additive Metrics on Trees

The mathematical tool that we will use to analyze and solve the network inference problem is called additive tree metric [4], [27], or *additive metric* for short. We consider trees with internal node degree at least three. Such trees are called canonical trees [4]. Note that all logical routing trees are canonical trees.

[1]With a little abuse of notation, in this paper when we write $a \cup B$ where a is an element and B is a set, we mean $\{a\} \cup B$.

Definition: $d: V \times V \to \mathbb{R}^+$ is an additive metric on $T = (V, E)$ if

$$(a) \quad 0 < d(e) < \infty, \qquad \forall e = (i,j) \in E;$$
$$(b) \quad d(i,j) = d(j,i) = \begin{cases} \sum_{e \in \mathcal{P}(i,j)} d(e), & i \neq j; \\ 0, & i = j. \end{cases}$$

Note that $d(e)$ can be viewed as the *length* of link e, and $d(i,j)$ can be viewed as the *distance* between nodes i and j. Basically, an additive metric associates each link on the tree with a finite positive link length, and the distance between two nodes on the tree is the summation of the link lengths along the path that connects the two nodes.

Suppose $T(s, D) = (V, E)$ is a routing tree with source node s and destination nodes D. Let

$$d(E) = \{d(e) : e \in E\}$$

denote the link lengths of $T(s, D)$ under additive metric d.

Remember $U = s \cup D$ is the set of terminal nodes on the tree. Let

$$d(U^2) = \{d(i,j) : i, j \in U\}$$

denote the distances between the terminal nodes.

Buneman [4] showed that the topology and link lengths of a tree are uniquely determined by the distances between the terminal nodes under an additive metric.

Theorem 1: (Buneman [4]) There is a one-to-one mapping between $(T(s, D), d(E))$ and $(U, d(U^2))$ under any additive metric d on $T(s, D)$.

From Theorem 1, we know that we can recover the topology and link lengths of a routing tree if we know $d(U^2)$. In addition, if there is a one-to-one mapping between the link performance parameters and link lengths (see Section III-A), then we can recover the link performance parameters from the link lengths. The challenges are:

(1) Constructing an additive metric for which we can derive/estimate $d(U^2)$ from measurements taken at the terminal nodes. We will address this issue in this section.

(2) Developing efficient and effective algorithms to recover the topology and link lengths from the (estimated) distances between the terminal nodes. We will address this issue in Section IV and Section V.

A. Construct Additive Metrics

A source node can employ different probing techniques, e.g., *multicast* probing and *unicast* probing, to send probes (packets) to a set of destination nodes. For multicast probing, when an internal node on the routing tree receives an packet from its parent, it will send a copy of the packet to all its children on the tree. For a multicast 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 state set $\mathcal{Z}$. The distribution of Z_e is parameterized by θ_e, e.g.,

$$\mathbb{P}(Z_e = z) = \theta_e(z), \quad \forall z \in \mathcal{Z}. \tag{1}$$

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 an outcome 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})

$$X_k = g(X_{f(k)}, Z_{e_k}). \tag{2}$$

Assumption 1: The link states are independent from link to link (spatial independence assumption) and are stationary during the measurement period (stationarity assumption).

Proposition 1: Under the spatial independence assumption

$$X_V \triangleq (X_k : k \in V) \tag{3}$$

is a Markov random field (MRF) on $T(s, D)$. 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 $T(s, D)$.

Proof: For notational simplification, we use $p(x_A)$ to represent $\mathbb{P}(X_k = x_k : k \in A)$ for any subset $A \subseteq V$. First we prove by induction that

$$p(x_V) = p(x_s) \prod_{k \in V \setminus s} p(x_k | x_{f(k)}). \tag{4}$$

Equation (4) is clearly true for any tree with $|V| = 1$ or $|V| = 2$. Assume (4) is true for any tree with $|V| \leq n$. Now consider a tree T with $|V| = n + 1$.

Let i be a leaf node of T, then by (2) and the spatial independence assumption we have

$$\begin{aligned} p(x_V) &= p(x_i | x_{V \setminus i}) p(x_{V \setminus i}) \\ &= p(g(x_{f(i)}, z_{e_i}) | x_{V \setminus i}) p(x_{V \setminus i}) \\ &= p(g(x_{f(i)}, z_{e_i}) | x_{f(i)}) p(x_{V \setminus i}) \\ &= p(x_i | x_{f(i)}) p(x_{V \setminus i}). \end{aligned} \tag{5}$$

$X_{V \setminus i}$ is defined on $T' = (V \setminus i, E \setminus e_k)$, a tree with n nodes. By induction assumption

$$p(x_{V \setminus i}) = p(x_s) \prod_{k \in V \setminus i \setminus s} p(x_k | x_{f(k)}).$$

Substituting it into (5) we have shown that (4) holds for T with $|V| = n + 1$. By induction argument, (4) is true for any tree.

Now for any $k \in V$, from (4) we have

$$p(x_V) = \left(p(x_k | x_{f(k)}) \prod_{j \in c(k)} p(x_j | x_k) \right) \cdot q(x_{V \setminus k})$$

$$p(x_{V \setminus k}) = \sum_{x_k} \left(p(x_k | x_{f(k)}) \prod_{j \in c(k)} p(x_j | x_k) \right) \cdot q(x_{V \setminus k})$$

where $q(x_{V\setminus k})$ is a factor that does not depend on x_k. Then

$$\begin{aligned}p(x_k|x_{V\setminus k}) &= \frac{p(x_V)}{p(x_{V\setminus k})}\\ &= \frac{p(x_k|x_{f(k)})\prod_{j\in c(k)} p(x_j|x_k)}{\sum_{x_k}\left(p(x_k|x_{f(k)})\prod_{j\in c(k)} p(x_j|x_k)\right)}\\ &= \frac{p(x_{f(k)})p(x_k|x_{f(k)})\prod_{j\in c(k)} p(x_j|x_k)}{\sum_{x_k}\left(p(x_{f(k)})p(x_k|x_{f(k)})\prod_{j\in c(k)} p(x_j|x_k)\right)}\\ &= p(x_k|x_{f(k)\cup c(k)}).\end{aligned}$$

Therefore, X_V is a Markov random field on $T(s,D)$. ∎

For an MRF $X_V = (X_k : k \in V)$ on $T(s,D)$, we can construct an additive metric as follows. Assume $|\mathcal{X}| = M$. For each link $(i,j) \in E$, we define an $M\times M$ forward link transition matrix P_{ij} and an $M \times M$ backward link transition matrix P_{ji} with entries

$$\begin{aligned}P_{ij}(x,y) &= \mathbb{P}(X_j = y|X_i = x)\\ P_{ji}(x,y) &= \mathbb{P}(X_i = y|X_j = x),\ x,y \in \mathcal{X}.\end{aligned}$$

Suppose that the link transition matrices are *invertible* (so that $|P_{ij}| := |\det(P_{ij})| > 0$) and are not equal to a *permutation matrix*[2](so that $|P_{ij}| < 1$). Further suppose that there exists a node $i \in V$ with positive marginal distribution. Then we can construct an additive metric d_0 with link length [2], [8]

$$d_0(e) = -\log|P_{ij}| - \log|P_{ji}|, \quad \forall e = (i,j) \in E. \tag{6}$$

For any pair of terminal nodes $i, j \in U$, the distance between i and j under additive metric d_0 can be computed by

$$d_0(i,j) = -\log|P_{ij}| - \log|P_{ji}|, \quad \forall i,j \in U. \tag{7}$$

We can construct other additive metrics using different types of measurements. Here we use loss measurements as an example. Additive metrics based on delay/utilization measurements can be found in [23].

For each link $e \in E$, the link state variable Z_e is a Bernoulli random variable which takes value 1 with probability α_e if link e is in *good state* and the probe can go through the link, and takes value 0 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 X_k is also a Bernoulli random variable, which takes value 1 if the probe successfully reaches node k. Since the probe is sent by the source node s, we have $X_s \equiv 1$. It is clear that for loss measurements

$$X_k = X_{f(k)} \cdot Z_{e_k} = \prod_{e\in\mathcal{P}(s,k)} Z_e. \tag{8}$$

[2]A permutation matrix is a matrix with exactly one entry in each row and in each column being 1 and other entries being 0.

If $0 < \alpha_e < 1$ for all links, then we can construct an additive metric d_l with link length

$$d_l(e) = -\log\alpha_e, \quad \forall e \in E. \tag{9}$$

Note that there is a one-to-one mapping between the link lengths and link success rates; hence, we can derive the link success rates from the link lengths, and vice versa.

Under the spatial independence assumption that the link states are independent from link to link, we have

$$\begin{aligned}\mathbb{P}(X_i = 1) &= \mathbb{P}\Big(\prod_{e\in\mathcal{P}(s,i)} Z_e = 1\Big) = \prod_{e\in\mathcal{P}(s,i)} \alpha_e\\ \mathbb{P}(X_j = 1) &= \mathbb{P}\Big(\prod_{e\in\mathcal{P}(s,j)} Z_e = 1\Big) = \prod_{e\in\mathcal{P}(s,j)} \alpha_e\\ \mathbb{P}(X_iX_j = 1) &= \mathbb{P}\Big(\prod_{e\in\mathcal{P}(s,\underline{ij})} Z_e \prod_{e\in\mathcal{P}(\underline{ij},i)} Z_e \prod_{e\in\mathcal{P}(\underline{ij},j)} Z_e = 1\Big)\\ &= \prod_{e\in\mathcal{P}(s,\underline{ij})} \alpha_e \prod_{e\in\mathcal{P}(\underline{ij},i)} \alpha_e \prod_{e\in\mathcal{P}(\underline{ij},j)} \alpha_e\end{aligned}$$

where $\underline{ij}$ denotes the nearest common ancestor of i and j on $T(s,D)$ (i.e., the common 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.

Therefore, the distances between the terminal nodes, $d_l(U^2)$, can be computed by

$$d_l(i,j) = \log\frac{\mathbb{P}(X_i = 1)\mathbb{P}(X_j = 1)}{\mathbb{P}^2(X_iX_j = 1)}, \quad \forall i,j \in U. \tag{10}$$

B. Estimation of Distances

From (7) 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 $d(U^2)$. In actual network inference problems, however, the joint distributions of the outcome variables are not given. We need to 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 $X_V^{(t)} = (X_k^{(t)} : k \in V)$, $t = 1,2,\ldots,n$, one for each probe. For the tth probe, only the outcome variables $X_U^{(t)} = (X_k^{(t)} : k \in U)$ at the terminal nodes can be measured. We can estimate the joint distributions of the outcome variables using the observed empirical distributions, which will converge to the actual distributions almost surely if the link state processes are stationary and ergodic during the measurement period.

Suppose s sends a sequence of n probes to (a subset of) destination nodes in D. For any probed node i, let $X_i^{(t)}$ be the measured loss outcome of the tth probe at node i, with $X_i^{(t)} = 1$ if node i successfully receives the probe and $X_i^{(t)} = 0$ otherwise. We use the empirical distributions of the outcome variables to estimate the distances. For a Bernoulli random variable X (as in

loss measurements), the empirical probability that X takes value 1 is just the sample mean $\bar{X}$ of the samples $X^{(1)},\ldots,X^{(n)}$

$$\hat{P}(X=1)=\bar{X}\triangleq\frac{1}{n}\sum_{t=1}^{n}X^{(t)}. \tag{11}$$

Note that $\bar{X}$ is the maximum likelihood estimator (MLE) of $\mathbb{P}(X=1)$ from the samples. One can also derive exponential error bound of $\bar{X}$ using Chernoff bounds.

Lemma 1: Let $X^{(1)},X^{(2)},\ldots,X^{(n)}$ be n independent samples of Bernoulli random variable X with $0<\mathbb{P}(X=1)=p<1$. Let $\bar{X}=\sum_{t=1}^{n}X^{(t)}/n$ be the sample mean of the samples. Then for $0<\delta\leq 1$

$$\mathbb{P}(\bar{X}<(1-\delta)p)<e^{-\frac{\delta^2 p}{2}n}$$
$$\mathbb{P}(\bar{X}>(1+\delta)p)<e^{-\frac{\delta^2 p}{3}n}.$$

We can construct explicit estimators for the distances in (10) as follows (we use ˆ over d to represent estimated distances)

$$\hat{d}_l(i,j)=\log\frac{\bar{X}_i\bar{X}_j}{\overline{X_iX_j}},\ \forall i,j\in U \tag{12}$$

where

$$\bar{X}_i=\frac{1}{n}\sum_{t=1}^{n}X_i^{(t)},\ \forall i\in U$$
$$\overline{X_iX_j}=\frac{1}{n}\sum_{t=1}^{n}X_i^{(t)}X_j^{(t)},\ \forall i,j\in U.$$

We can derive exponential error bounds for the distance estimators in (12) based on the previous lemma.

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

$$\mathbb{P}(|\hat{d}_l(i,j)-d_l(i,j)|>\epsilon)<e^{-c_{ij}(\epsilon)n} \tag{13}$$

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

Proof: First consider the case where $i=s$ (source node) and $j\in D$ (destination node). Since $X_s^{(t)}=1,\forall t$, we have

$$\hat{d}_l(s,j)=-\log\bar{X}_j,\ \forall j\in D. \tag{14}$$

Let $p_j=\mathbb{P}(X_j=1)$ be the probability that destination node j successfully receives a probe sent by s. Using Lemma 1, for any small δ $(0<\delta\leq 1)$, we have

$$\mathbb{P}(\bar{X}_j<(1-\delta)p_j)=\mathbb{P}\left(-\log\bar{X}_j>-\log p_j-\log(1-\delta)\right)$$
$$<e^{-\frac{\delta^2 p_j}{2}n}.$$

Let $\delta=1-e^{-\epsilon}$ we have

$$\mathbb{P}\left(-\log\bar{X}_j>-\log p_j+\epsilon\right)<e^{-\frac{(1-e^{-\epsilon})^2 p_j}{2}n}.$$

Similarly, for any $\epsilon>0$, we have

$$\mathbb{P}\left(-\log\bar{X}_j<-\log p_j-\epsilon\right)<e^{-\frac{(e^{\epsilon}-1)^2 p_j}{3}n}.$$

Therefore

$$\begin{aligned}&\mathbb{P}\left(|\hat{d}_l(s,j)-d_l(s,j)|>\epsilon\right)\\&=\mathbb{P}\left(|\log\bar{X}_j-\log p_j|>\epsilon\right)\\&<e^{-\frac{(1-e^{-\epsilon})^2}{2}p_j n}+e^{-\frac{(e^{\epsilon}-1)^2}{3}p_j n}\\&\triangleq e^{-c_{sj}(\epsilon)n}.\end{aligned} \tag{15}$$

Now consider the case where both $i,\ j\in D$. Then $\hat{d}_l(i,j)=\log\frac{\bar{X}_i\bar{X}_j}{\overline{X_iX_j}}$. Since $\bar{X}_i$, $\bar{X}_j$ and $\overline{X_iX_j}$ are sample means of Bernoulli random variables with parameter p_i, p_j and p_{ij} $(\mathbb{P}(X_iX_j=1)=p_{ij})$ respectively, applying (15) and by triangular inequality and union bound

$$\begin{aligned}&\mathbb{P}\left(|\hat{d}_l(i,j)-d_l(i,j)|>\epsilon\right)\\&=\mathbb{P}\left(|\log\frac{\bar{X}_i\bar{X}_j}{\overline{X_iX_j}}-\log\frac{p_ip_j}{p_{ij}^2}|>\epsilon\right)\\&<\mathbb{P}\Bigg(|\log\bar{X}_i-\log p_i|>\frac{\epsilon}{4}\cup|\log\bar{X}_j-\log p_j|>\frac{\epsilon}{4}\\&\quad\cup|\log\overline{X_iX_j}-\log p_{ij}|>\frac{\epsilon}{4}\Bigg)\\&<e^{-\frac{(1-e^{-\frac{\epsilon}{4}})^2}{2}p_i n}+e^{-\frac{(e^{\frac{\epsilon}{4}}-1)^2}{3}p_i n}+e^{-\frac{(1-e^{-\frac{\epsilon}{4}})^2}{2}p_j n}\\&\quad+e^{-\frac{(e^{\frac{\epsilon}{4}}-1)^2}{3}p_j n}+e^{-\frac{(1-e^{-\frac{\epsilon}{4}})^2}{2}p_{ij} n}+e^{-\frac{(e^{\frac{\epsilon}{4}}-1)^2}{3}p_{ij} n}\\&\triangleq e^{-c_{ij}(\epsilon)n}.\end{aligned} \tag{16}$$

■

Remark 1: Note that $c_{ij}(\epsilon)$'s are some constants that depend on the error tolerance ϵ. Suppose that we want to bound the relative error $\mathbb{P}(|\hat{d}_l(i,j)-d_l(i,j)|>\zeta d_l(i,j))$. When the link lengths are small (i.e., when the link loss rates are small), $d_l(i,j)$ is small, and $c_{ij}(\zeta d_l(i,j))$ tends to be small, so it requires more measurements to ensure a certain (relative) estimation accuracy.

C. Other Additive Metrics and Information Fusion

We can also construct additive metrics and compute/estimate the distances between the terminal nodes using (end-to-end) unicast packet pair probing or traceroute probing, as described in [23]. A nice property of additive metrics is that a linear combination of several additive metrics is still an additive metric. In order to fuse information collected from different measurements, we can construct a new additive metric using a linear (convex) combination of additive metrics $d_1,d_2,\ldots,d_k$

$$\begin{aligned}d&=a_1d_1+a_2d_2+\cdots+a_kd_k\\&\text{s.t. } a_1+a_2+\cdots+a_k=1.\end{aligned} \tag{17}$$

The estimated distance between terminal nodes $i,\ j\in U$ under the new additive metric can be easily computed

$$\hat{d}(i,j)=a_1\hat{d}_1(i,j)+a_2\hat{d}_2(i,j)+\cdots+a_k\hat{d}_k(i,j).$$

In practice we can select the coefficients empirically based on the current network state or to minimize the variance of the estimator $\hat{d}$.

IV. Neighbor-Joining Algorithm

We have described how to construct additive metrics and estimate the distances between the terminal nodes via end-to-end packet probing measurements. In this section we introduce the *neighbor-joining* (NJ) algorithm proposed by Saitou and Nei [26], which is considered one of the most widely used algorithms for building binary phylogenetic trees from distances [15], [29].

Definition 2: A distance-based tree inference algorithm takes the (estimated) distances between the terminal nodes of a tree as the input and returns a tree topology and the associated link lengths. The input distances $\hat{d}(U^2)$ satisfy:

(1) $\hat{d}(i,j) \geq 0$, with equality if and only if $i = j$;
(2) $\hat{d}(i,j) = \hat{d}(j,i)$.

Definition 3: A group of nodes on a tree are called neighbors (siblings), if they are connected via one internal node (if they have the same parent) on the tree.

The NJ algorithm is an *agglomerative algorithm*. The algorithm begins with a leaf set which includes all the destination nodes. In each step it selects two leaf nodes that are likely to be neighbors, deletes them from the leaf set, creates a new node as their parent and adds the new 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. To avoid trivial cases, we assume $|D| \geq 2$.

Algorithm 1: Neighbor-Joining (NJ) Algorithm for Binary Trees

Input: Estimated distances between the nodes in U: $\hat{d}(U^2)$.

1. $V = \{s\}$, $E = \emptyset$.

2.1. For any pair of nodes $i, j \in D$, compute

$$\hat{Q}(i,j) = \sum_{k \in U} \hat{d}(i,k) + \sum_{k \in U} \hat{d}(j,k) - (|U|-2)\hat{d}(i,j). \quad (18)$$

2.2. Find $i^*, j^* \in D$ with the largest $\hat{Q}(i,j)$ (break the tie arbitrarily).
Create a node f as the parent of i^* and j^*.
$D = D \setminus \{i^*, j^*\}$, $U = U \setminus \{i^*, j^*\}$,
$V = V \cup \{i^*, j^*\}$, $E = E \cup \{(f, i^*), (f, j^*)\}$.

2.3. Compute the link lengths from the distances

$$\hat{d}(f, i^*) = \frac{1}{|U|} \sum_{k \in U} [\hat{d}(k, i^*) + \hat{d}(i^*, j^*) - \hat{d}(k, j^*)]/2$$

$$\hat{d}(f, j^*) = \frac{1}{|U|} \sum_{k \in U} [\hat{d}(k, j^*) + \hat{d}(i^*, j^*) - \hat{d}(k, i^*)]/2.$$

2.4. For each $k \in U$, compute the distance between k and f

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

$D = D \cup f$, $U = U \cup f$.

3. If $|D| = 1$, for the $i \in D$: $V = V \cup i$, $E = E \cup (s, i)$.
Otherwise, repeat Step 2.

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

The NJ algorithm has several nice properties:

- it has a polynomial-time complexity $O(N^3)$ for (binary) trees with N terminal nodes;
- it returns the correct tree topology and link lengths if the input distances are indeed *additive* (i.e., if the input distances are derived from an additive metric without estimation errors);
- it is robust: it achieves the optimal l_∞-radius among all distance-based algorithms for binary trees.

The l_∞-radius notation was introduced by Atteson [1].

Definition 4: For a distance-based algorithm, we say it has l_∞-radius r, if for any tree T associated with any additive metric d, whenever the input distances between the terminal nodes $\hat{d}(U^2)$ satisfy

$$\|\hat{d}(U^2) - d(U^2)\|_\infty \triangleq \max_{i,j \in U} |\hat{d}(i,j) - d(i,j)| < r \min_{e \in E} d(e) \quad (19)$$

the algorithm will return the correct topology of T.

An algorithm which achieves the optimal (maximum) l_∞-radius is the most robust in the sense that it can tolerate the maximum estimation error. [1] showed that no distance-based algorithm has l_∞-radius larger than $\frac{1}{2}$ via an example, and proved that the NJ algorithm in fact achieves the optimal l_∞-radius for binary trees.

Theorem 2: (Atteson [1]) The NJ algorithm achieves the optimal l_∞-radius $\frac{1}{2}$ for binary trees.

The major differences between our network inference problem and the phylogenetic inference problem include: (1) for the network inference problem we can control and observe the source node, while for the phylogenetic inference problem the information of the source node (the common ancestor of all the species) is lost; (2) for a network routing tree the degrees of the internal nodes are arbitrary (i.e., general trees), while for a phylogenetic tree the degrees of the internal nodes are three (i.e., binary trees). In Section VI we will extend the NJ algorithm to handle these differences.

V. Rooted Neighbor-Joining Algorithm

A. Binary Trees

We first present an algorithm which can be viewed as a *rooted* version of the NJ algorithm for binary trees. To avoid trivial cases, we assume $|D| \geq 2$.

Algorithm 2: Rooted Neighbor-Joining (RNJ) Algorithm for Binary Trees

Input: Estimated distances between the nodes in U: $\hat{d}(U^2)$.

1. $V = \{s\}$, $E = \emptyset$.
For any pair of nodes $i, j \in D$, compute

$$\hat{q}(i,j) = \frac{\hat{d}(s,i) + \hat{d}(s,j) - \hat{d}(i,j)}{2}. \quad (20)$$

2.1. Find $i^*, j^* \in D$ with the largest $\hat{q}(i,j)$ (break the tie arbitrarily).
Create a node f as the parent of i^* and j^*.

$D = D \setminus \{i^*, j^*\}$,
$V = V \cup \{i^*, j^*\}$, $E = E \cup \{(f, i^*), (f, j^*)\}$.

2.2. Compute

$$\hat{d}(s,f) = \hat{q}(i^*, j^*)$$
$$\hat{d}(f,i^*) = \hat{d}(s,i^*) - \hat{q}(i^*, j^*)$$
$$\hat{d}(f,j^*) = \hat{d}(s,j^*) - \hat{q}(i^*, j^*).$$

2.3. For each $k \in D$, compute

$$\hat{d}(k,f) = \frac{1}{2}[\hat{d}(k,i^*) - \hat{d}(f,i^*)] + \frac{1}{2}[\hat{d}(k,j^*) - \hat{d}(f,j^*)]$$
$$\hat{q}(k,f) = \frac{1}{2}[\hat{d}(s,k) + \hat{d}(s,f) - \hat{d}(k,f)]$$
$$= \frac{1}{2}[\hat{q}(k,i^*) + \hat{q}(k,j^*)].$$

$D = D \cup f$.

3. If $|D| = 1$, for the $i \in D : V = V \cup i$, $E = E \cup (s,i)$. Otherwise, repeat Step 2.

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

Note that the major difference between the NJ algorithm and the RNJ algorithm is the selection of the *score function*: the NJ algorithm uses the $\hat{Q}$ function defined in (18); while the RNJ algorithm uses the $\hat{q}$ function in (20), which has a simple interpretation that we will explain next.

For any pair of nodes $i, j \in D$, remember $\underline{ij}$ is their nearest common ancestor on $T(s, D)$. Under additive metric d, we can see that

$$q(i,j) = \frac{d(s,i) + d(s,j) - d(i,j)}{2} = d(s, \underline{ij}) \tag{21}$$

which is the distance from the root (source node s) to $\underline{ij}$. It is not hard to see that a pair of nodes i^*, j^* with the largest $q(i,j)$ must be neighbors (siblings) on the tree. $\hat{q}(i,j)$ in (20) is the estimated distance from the root to $\underline{ij}$ computed from the input distances. If the input distances are close to the true additive distances, then we would expect that the two nodes selected in Step 2.1 of Algorithm 2 are indeed neighbors.

We provide a sufficient condition for Algorithm 2 to return the correct tree topology. From this condition we can establish several nice properties of the algorithm.

Lemma 2: For binary trees, a sufficient condition for Algorithm 2 to return the correct tree topology is

$$\forall i,j,k \in D \text{ s.t. } \underline{ij} \prec \underline{ik}$$
$$\Rightarrow \quad \hat{q}(i,j) > \hat{q}(i,k) \tag{22}$$

where $\underline{ij} \prec \underline{ik}$ means that $\underline{ij}$ is descended from $\underline{ik}$.

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

(1) If $|D| = 2$, then clearly Algorithm 2 will return the correct tree topology.

(2) Assume Algorithm 2 returns the correct tree topology under condition (22) for $|D| \le N$. Now consider $|D| = N + 1$.

Claim 1: i^*, j^* found in Step 2.1 which maximize $\hat{q}(i,j)$ are siblings (neighbors).

If i^* and j^* are not siblings, then there exists $k \in D$ such that either $\underline{i^*k}$ or $\underline{j^*k}$ is descended from $\underline{i^*j^*}$. Under condition (22), this implies either $\hat{q}(i^*,k) > \hat{q}(i^*, j^*)$ or $\hat{q}(j^*,k) > \hat{q}(i^*, j^*)$, a contradiction to the maximality of $\hat{q}(i^*, j^*)$.

Claim 2: Condition (22) is maintained over the nodes in D after Step 2.

After Step 2, i^*, j^* are deleted from D and f is added to D as a new leaf node. Since i^*, j^* are siblings and f is their parent, we know that for any $i \in D$, $\underline{if} = \underline{ii^*} = \underline{ij^*}$. Therefore, $\forall i$, $j \in D$ s.t. $\underline{ij} \prec \underline{if}$, we have $\underline{ij} \prec \underline{ii^*}$ and $\underline{ij} \prec \underline{ij^*}$, which implies $\hat{q}(i,j) > \hat{q}(i,i^*)$ and $\hat{q}(i,j) > \hat{q}(i,j^*)$, hence

$$\hat{q}(i,j) > \hat{q}(i,f) = \frac{1}{2}[\hat{q}(i,i^*) + \hat{q}(i,j^*)].$$

Similarly, $\forall i,k \in D$ s.t. $\underline{if} \prec \underline{ik}$, we can show $\hat{q}(i,f) > \hat{q}(i,k)$.

From claims 1 and 2, we know that after one iteration of Step 2, Algorithm 2 will correctly find out a pair of siblings, and condition (22) is maintained for the new set of leaf nodes in D. Then $|D|$ is decreased by 1. By induction assumption, the algorithm will return the correct topology of the remaining part of the tree. This completes our proof of the lemma. ■

Proposition 3: For binary trees, Algorithm 2 will return the correct tree topology and link lengths if the input distances $\hat{d}(U^2)$ are additive.

Proof: If the input distances are additive, then $\hat{q}(i,j)$ and $\hat{q}(i,k)$ are the actual distances from s to $\underline{ij}$ and $\underline{ik}$ under an additive metric. In this case, if $\underline{ij}$ is descended from $\underline{ik}$, since link lengths are positive, we have $\hat{q}(i,j) > \hat{q}(i,k)$; hence, condition (22) holds. Then by Lemma 2, Algorithm 2 will return the correct tree topology. In addition, under additive distances, the link lengths computed in Step 2.2 of Algorithm 2 are correct. ■

In network inference problems, the distances between the terminal nodes are estimated from measurements taken by the end hosts. The estimated distances may deviate from the true additive distances due to measurement errors. Nevertheless, we will show that if the estimated distances are close enough to the true distances, then Algorithm 2 will return the correct tree topology. In fact, Algorithm 2 achieves the optimal l_∞-radius among all distance-based algorithms.

Proposition 4: The RNJ algorithm (Algorithm 2) achieves the optimal l_∞-radius $\frac{1}{2}$ for binary trees, i.e., for any binary tree associated with any additive metric d, whenever the input distances $\hat{d}(U^2)$ satisfy

$$\max_{i,j \in U} |\hat{d}(i,j) - d(i,j)| < \frac{1}{2} \min_{e \in E} d(e). \tag{23}$$

Algorithm 2 will return the correct tree topology.

Proof: Using Lemma 2 we only need to show that condition (23) implies condition (22). Let

$$\Delta = \min_{e \in E} d(e)$$

be the minimum link length on the tree. If $\underline{ij} \prec \underline{ik}$, i.e., if $\underline{ij}$ is descended from $\underline{ik}$, since link lengths $\geq\Delta$, we have $q(i,j) - q(i,k) \geq \Delta$. Then from (20), (21), (23), we have

$$\begin{aligned}
&\hat{q}(i,j) - \hat{q}(i,k) \\
&\geq (\hat{q}(i,j) - \hat{q}(i,k)) - (q(i,j) - q(i,k) - \Delta) \\
&\geq \Delta - \frac{1}{2}|(\hat{d}(s,j) - d(s,j)| - \frac{1}{2}|\hat{d}(i,j) - d(i,j)| \\
&\quad - \frac{1}{2}|\hat{d}(s,k) - d(s,k)| - \frac{1}{2}|\hat{d}(i,k) - d(i,k)| \\
&> \Delta - \frac{1}{4}\Delta - \frac{1}{4}\Delta - \frac{1}{4}\Delta - \frac{1}{4}\Delta = 0.
\end{aligned}$$

Hence, condition (23) indeed implies condition (22). Since (22) is a sufficient condition for Algorithm 2 to return the correct tree topology, (23) is also a sufficient condition for Algorithm 2 to return the correct tree topology. ∎

B. General Trees

Recall that $q(i,j)$ is the distance from the root to the nearest common ancestor of nodes i and j. For a general routing tree with positive link lengths, we have the following observations of the q function (where we consider leaf nodes i, j):

- If nodes i and j are neighbors on the tree, then for any other node k on the tree we have

$$q(i,j) \geq q(i,k). \tag{24}$$

- If nodes i and j are neighbors on the tree, then for any other node k that is also a neighbor of i and j we have

$$q(i,j) = q(i,k) \tag{25}$$

  because $\underline{ij} = \underline{ik}$.
- If nodes i and j are neighbors on the tree, then for any other node k that is not a neighbor of i and j we have

$$q(i,j) \geq q(i,k) + \Delta \tag{26}$$

  (where Δ is the minimum link length) because $\underline{ij}$ is descended from $\underline{ik}$ and they are separated by at least one link.

Therefore, we can determine whether a group of nodes are neighbors or not on the tree from knowledge of the q function under an additive metric. To extend the RNJ algorithm (Algorithm 2) for general trees, after we find out two nodes i^* and j^* that are likely to be neighbors in Step 2.1, we need to find out other nodes that are likely to be neighbors of i^* and j^* based on $\hat{q}$ computed from the input distances. We use the following threshold neighbor criterion:

Threshold Neighbor Selection Criterion: Suppose i^* and j^* are neighbors on the tree. Node k will be chosen as a neighbor of i^* and j^* if and only if

$$\hat{q}(i^*,j^*) - \hat{q}(i^*,k) \leq t \tag{27}$$

for some threshold $t > 0$.

Based on observations (25) and (26), and since $\hat{q}$ is an estimation of q with possible estimation errors, we use the middle point $\frac{\Delta}{2}$ as the threshold. Later we will show that such a choice enables the algorithm to achieve the optimal l_∞-radius $\frac{1}{4}$ for general trees if the threshold criterion is used in the algorithm (see the proof of Proposition 7).

Algorithm 3: Rooted Neighbor-Joining (RNJ) Algorithm for General Trees

---

Input: Estimated distances between the nodes in U : $\hat{d}(U^2)$; estimated minimum link length $\Delta > 0$.

1. $V = \{s\}$, $E = \emptyset$.
   For any pair of nodes $i, j \in D$, compute

$$\hat{q}(i,j) = \frac{\hat{d}(s,i) + \hat{d}(s,j) - \hat{d}(i,j)}{2}. \tag{28}$$

2.1. Find $i^*, j^* \in D$ with the largest $\hat{q}(i,j)$ (break the tie arbitrarily).
   Create a node f as the parent of i^* and j^*.
   $D = D \setminus \{i^*, j^*\}$,
   $V = V \cup \{i^*, j^*\}$, $E = E \cup \{(f,i^*), (f,j^*)\}$.

2.2. Compute

$$\begin{aligned}
\hat{d}(s,f) &= \hat{q}(i^*,j^*) \\
\hat{d}(f,i^*) &= \hat{d}(s,i^*) - \hat{q}(i^*,j^*) \\
\hat{d}(f,j^*) &= \hat{d}(s,j^*) - \hat{q}(i^*,j^*).
\end{aligned}$$

2.3. For every $k \in D$ such that $\hat{q}(i^*,j^*) - \hat{q}(i^*,k) \leq \frac{\Delta}{2}$:
   $D = D \setminus k$, $V = V \cup k$, $E = E \cup (f,k)$.
   Compute: $\hat{d}(f,k) = \hat{d}(s,k) - \hat{q}(i^*,j^*)$.

2.4. For each $k \in D$, compute

$$\begin{aligned}
\hat{d}(k,f) &= \frac{1}{2}[\hat{d}(k,i^*) - \hat{d}(f,i^*)] + \frac{1}{2}[\hat{d}(k,j^*) - \hat{d}(f,j^*)] \\
\hat{q}(k,f) &= \frac{1}{2}[\hat{d}(s,k) + \hat{d}(s,f) - \hat{d}(k,f)] \\
&= \frac{1}{2}[\hat{q}(k,i^*) + \hat{q}(k,j^*)].
\end{aligned}$$

   $D = D \cup f$.

3. If $|D| = 1$, for the $i \in D$: $V = V \cup i$, $E = E \cup (s,i)$.
   Otherwise, repeat Step 2.

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

Lemma 3: If $\Delta \leq \min_{e\in E} d(e)$, a sufficient condition for Algorithm 3 to return the correct tree topology is

$$\begin{aligned}
&\forall i,j,k \in D \text{ s.t. } \underline{ij} \prec \underline{ik} \Rightarrow \hat{q}(i,j) - \hat{q}(i,k) > \frac{\Delta}{2} \\
&\forall i,j,k \in D \text{ s.t. } \underline{ij} = \underline{ik} \Rightarrow |\hat{q}(i,j) - \hat{q}(i,k)| \leq \frac{\Delta}{2}.
\end{aligned} \tag{29}$$

Proof: We outline the proof, which is similar to the proof of Lemma 2. There are three key observations:

(1) Under condition (29), i^*, j^* found in Step 2.1 of Algorithm 3 are siblings.

(2) Under condition (29), k will be selected in Step 2.3 if and only if it is a sibling of i^* and j^*.
(3) Condition (29) is maintained over the nodes in D after Step 2.

The lemma then follows by induction on the cardinality of D. ∎

Proposition 5: For general trees, Algorithm 3 will return the correct tree topology and link lengths if the input distances $\hat{d}(U^2)$ are additive.

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

In practice the input distances may deviate from the true additive distances due to measurement errors. We can show that if the input distances are close enough to the true additive distances, then Algorithm 3 will return the correct tree topology.

Proposition 6: For a general tree with additive metric d, if the input parameter $\Delta \leq \min_{e\in E} d(e)$ and the input distances $\hat{d}(U^2)$ satisfy

$$\max_{i,j\in U} |\hat{d}(i,j) - d(i,j)| < \frac{\Delta}{4} \tag{30}$$

then Algorithm 3 will return the correct tree topology.

Proof: The proof is similar to the proof of Proposition 4. We can show that condition (30) implies condition (29), then the proposition follows by Lemma 3. ∎

If the input parameter $\Delta = \min_{e\in E} d(e)$, then Proposition 6 says that the RNJ algorithm has l_∞-radius $\frac{1}{4}$ for general trees.

Corollary 1: The RNJ algorithm (Algorithm 3) has l_∞-radius $\frac{1}{4}$ for general trees when $\Delta = \min_{e\in E} d(e)$.

We now show that no distance-based algorithm has l_∞-radius greater than $\frac{1}{4}$ if the threshold neighbor selection criterion (27) is used in the algorithm.

Proposition 7: If the threshold neighbor selection criterion (27) is used, then no distance-based algorithm has l_∞-radius greater than $\frac{1}{4}$ for general trees.

Proof: Suppose $\mathcal{A}$ is a distance-based algorithm with l_∞-radius r in which the threshold criterion (27) is used. Therefore, for any tree T associated with any additive metric d, if the input distances $\hat{d}(U^2)$ satisfy

$$\max_{i,j\in U} |\hat{d}(i,j) - d(i,j)| < r\Delta \tag{31}$$

where $\Delta = \min_{e\in E} d(e)$, then $\mathcal{A}$ will return the correct topology of T.

Suppose i^* and j^* are neighbors on T, and k is a neighbor of them, then we have $q(i^*, j^*) = q(i^*, k)$. Under condition (31) we know

$$\hat{q}(i^*, j^*) - \hat{q}(i^*, k) < 2r\Delta.$$

Since the threshold criterion (27) is used, we need to have

$$\hat{q}(i^*, j^*) - \hat{q}(i^*, k) < 2r\Delta \leq t \Rightarrow r \leq \frac{t}{2\Delta} \tag{32}$$

to correctly add k as a neighbor of i^* and j^*.

Now suppose k' is not a neighbor of i^* and j^*. Then we have $q(i^*, j^*) - q(i^*, k') \geq \Delta$. Under condition (31) we know

$$\hat{q}(i^*, j^*) - \hat{q}(i^*, k') > \Delta - 2r\Delta.$$

Since the threshold criterion (27) is used, we need to have

$$\hat{q}(i^*, j^*) - \hat{q}(i^*, k') > \Delta - 2r\Delta \geq t \Rightarrow r \leq \frac{1}{2} - \frac{t}{2\Delta} \tag{33}$$

to correctly not add k' as a neighbor of i^* and j^*.

Combining (32) and (33) we have

$$r \leq \min\left(\frac{t}{2\Delta}, \frac{1}{2} - \frac{t}{2\Delta}\right) \Rightarrow r \leq \frac{1}{4} \tag{34}$$

where the upper bound $\frac{1}{4}$ of r is achieved with the threshold $t = \frac{\Delta}{2}$. ∎

Combining the previous two results, we know that the RNJ algorithm (Algorithm 3) achieves the optimal l_∞-radius $\frac{1}{4}$ for general trees if the threshold neighbor selection criterion is used. We conjecture that a stronger result holds: the RNJ algorithm achieves the optimal l_∞-radius $\frac{1}{4}$ for general trees no matter what neighbor selection criterion is used.

C. Complexity and Consistency

The computational complexity of the RNJ algorithm is $O(N^2 \log N)$ for a routing tree with N destination nodes. We now show the *consistency* of the RNJ algorithm for general trees (Algorithm 3), and a similar result holds for binary trees.

Let $\hat{T}(n)$ be the inferred tree topology returned by the RNJ algorithm with a sample size n (number of probes to estimate the distances between the terminal nodes). Let

$$P_n = \mathbb{P}\{\hat{T}(n) = T\}$$

denote the probability of correct topology inference of the RNJ algorithm.

Proposition 8: Let $\Delta \leq \min_{e\in E} d(e)$ be the input parameter of the RNJ algorithm. If

$$\mathbb{P}\left\{|\hat{d}(i,j) - d(i,j)| \geq \frac{\Delta}{4}\right\} \leq e^{-c_{ij}(\Delta)n}, \forall i,j \in U \tag{35}$$

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

$$P_n \geq 1 - N^2 e^{-c(\Delta)n} \tag{36}$$

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

Proof: By Proposition 6 we have

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

∎

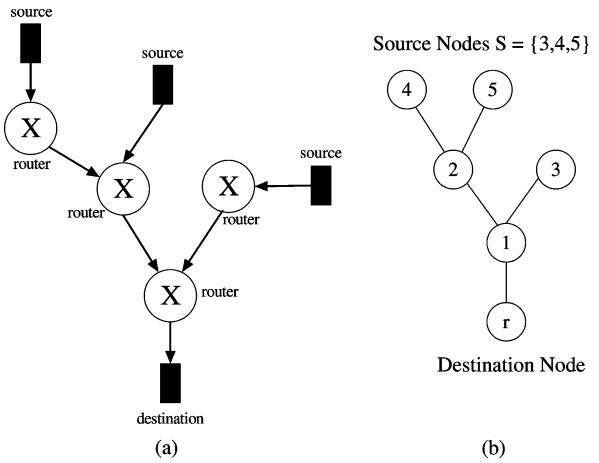

Fig. 2. Physical routing topology and the associated logical routing tree with multiple source nodes and a single destination node. (a) The physical routing topology. (b) The logical routing tree.

Proposition 9: If the input distances $\hat{d}(U^2)$ are consistent (i.e., they converge to the true distances in probability in the sample size) and the RNJ algorithm returns the correct tree topology, then the link lengths returned by the RNJ algorithm are consistent.

If we use the distance estimators in (12), since they satisfy condition (35) (see Proposition 2) and are consistent, by Proposition 8, the probability of correct topology inference of the RNJ algorithm goes to 1 exponentially fast in the sample size. If the inferred topology is correct, then by Proposition 9, the returned link lengths are also consistent. For network inference problems where there is a one-to-one mapping between the link performance parameters and the link lengths [e.g., (9)], the link lengths returned by the RNJ algorithm provide consistent estimators for the link performance parameters (e.g., success rates).

VI. Multiple-Source Single-Destination Network Inference

In this section we study the network inference problem of estimating the routing topology and link performance from multiple source nodes to a single destination node, in contrast to the single-source multiple-destination network inference problem we have addressed in the previous sections.

As before 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. Therefore, the physical routing topology from a set of source nodes to a destination node forms a reversed directed tree. From the physical routing topology, we can derive a logical routing tree which consists of the source nodes, the destination node, and the *joining* nodes (internal nodes with at least two incoming links) of the physical routing tree. Each internal node on the logical routing tree has degree at least three, and a logical link may comprise more than one physical links. An example is shown in Fig. 2.

Let r be a destination node (receiver) in the network, and S be a set of source nodes that will communicate with r. Let $T(S,r) = (V,E)$ denote the (logical) routing tree from nodes in S to r, with node set V and link set E. Let $U = S \cup r$ be the set of terminal nodes, which are nodes with degree one.

Each node $k \in V$ has a *child* $c(k) \in V$ such that $(k, c(k)) \in E$, and a set of parents $f(k) = \{j \in V : c(j) = k\}$, except that the destination node has no child and the source nodes have no parents. For notational simplification, we use e_k to denote link $(k, c(k))$. Each link e_k is associated with a performance parameter θ_k (e.g., success rate, delay distribution, utilization) that we want to estimate. The network inference problem involves using measurements taken at the terminal nodes to infer:

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

A. Reverse Multicast Probing

Similar to multicast probing from a source node to a set of destination nodes, we can have reverse multicast probing from a set of source nodes to a single destination node. We illustrate the idea of reverse multicast using Fig. 2(b) as the example. Under a reverse multicast probing, source nodes 4 and 5 will send a packet (probe) to their child node 2. Node 2 may receive both packets, or one of them, or none of them (because of packet loss). If node 2 receives at least one packet from its parents, it will combine (e.g., concatenate) the packets and sends the combined packet (as a probe) to its child node 1. Otherwise, node 2 will send nothing. Similarly, source node 3 will send a packet to its child node 1. Node 1 combines the packets received from its parents (if any) and sends the combined packet to the destination node r. The whole process is like the reverse process of multicasting a probe from node r to the other nodes on the routing tree.

For a probe sent from the source nodes in S to the destination node r, we define a set of link state variables Z_e for all links on the routing tree $T(S,r)$. Using loss measurements as example, Z_e is a Bernoulli random variable which takes value 1 with probability α_e if the probe can go through link e, and takes value 0 with probability $1 - \alpha_e \triangleq \bar{\alpha}_e$ if the probe is lost on the link.

For each node k on the routing tree, we use X_k to denote the (random) outcome of the probe sent from node k observed by the destination node r. For loss measurements, X_k takes value 1 if r successfully receives the probe sent from node k, and takes value 0 otherwise. It is clear that for any source node i

$$X_i = Z_{e_i} \cdot X_{c(i)} = \prod_{e \in \mathcal{P}(i,r)} Z_e. \tag{37}$$

If $0 < \alpha_e < 1$ for all links, then we can construct an additive metric d_l with link length

$$d_l(e) = -\log \alpha_e, \quad \forall e \in E. \tag{38}$$

For any pair of source nodes $i, j \in S$, let $\underline{ij}$ denote their nearest common descendant on $T(S,r)$ (i.e., the descendant of both nodes i and j that is closest to i and j on the routing tree). For example, in Fig. 2(b), the nearest common descendant of source nodes 4 and 5 is node 2, and the nearest common descendant of source nodes 3 and 4 is node 1.

Under the spatial independence assumption that the link states are independent from link to link, for any pair of source nodes i and j, we have

$$\mathbb{P}(X_i = 1) = \prod_{e \in \mathcal{P}(i,r)} \alpha_e$$
$$\mathbb{P}(X_j = 1) = \prod_{e \in \mathcal{P}(j,r)} \alpha_e$$
$$\mathbb{P}(X_i X_j = 1) = \prod_{e \in \mathcal{P}(i,\underline{ij})} \alpha_e \prod_{e \in \mathcal{P}(j,\underline{ij})} \alpha_e \prod_{e \in \mathcal{P}(\underline{ij},r)} \alpha_e.$$

Therefore, the distances between the terminal nodes, $d_l(U^2)$, can be computed by

$$d_l(i,j) = \log \frac{\mathbb{P}(X_i = 1)\mathbb{P}(X_j = 1)}{\mathbb{P}^2(X_i X_j = 1)}, \quad i,j \in U \tag{39}$$

where $\mathbb{P}(X_r = 1) = 1$.

We can see that the mathematical model of a reverse multicast probing on a routing tree (with multiple source nodes and a single destination node) is similar to the mathematical model of a multicast probing on a routing tree (with a single source node and multiple destination nodes). Therefore, the additive-metric framework can be directly applied to analyze and solve the multiple-source single-destination network inference problem. Specifically, we can construct additive metrics, estimate the distances between the terminal nodes from end-to-end measurements, and apply the distance-based algorithms to infer the routing tree topology and the link performance metrics.

B. Passive Network Monitoring in Wireless Sensor Networks

Although the current Internet does not support reverse multicast probing because internal nodes (routers) do not combine packets sent from different source nodes to a destination node, reverse multicast can be deployed in wireless networks (e.g., [17], [18]) for efficient data collecting.

A typical scenario in wireless sensor networks for data collecting is as follows. A base station (a receiver) will first propagate an *interest message* into the network via flooding or constrained/directional flooding. An interest message could be a query message which specifies what the base station wants (e.g., temperature and humidity statistics). A node, when first receives the interest message from another node, will set that node as its *child* and forward the interest message to its own neighbors excluding its child. Hence, the interest propagation procedure serves both to disseminate the interest message, and to set up a *reverse path* from each node to the base station.

When a sensor node which has the data of interest (a source node) receives the interest message, it can send the data back to the base station using the reverse path. Assume each source node has a unique ID (e.g., the geographical location of the node). The data sent by a source node to the base station also include the source node's ID so the base station knows from where it receives the data. If each node selects only one node as its child, i.e., if there is a unique path from a node to the base station, then we know that the routing topology (undirected version) from the source nodes to the base station is a tree. We call it a data collecting tree. Each internal node on the tree only needs to maintain the information of a set of parents that it will receive data from, and a child that it will send data to.

If directed diffusion [17] is applied on the data collecting tree, under which an internal node will aggregate (e.g., combine, compress, or code) the data received from its parents and then send the aggregated data to its child. Then this process is like a reverse multicast probing process as we described in Section VI-A. Using the inference algorithms we have introduced in this paper, the base station can infer: (1) the topology of the data collecting tree; (2) the link performance (e.g., packet delivery rate) of every link on the data collecting tree.

There are several advantages for the base station to do network inference using the collected data from the sensor nodes. First, the (internal) sensor nodes do not need to measure and infer the link performance which can save their resources, while normally the base station has sufficient resources so it is competent for the network inference task. Second, this is a passive network monitoring framework so no extra probing traffic is generated. Finally, since the inference is based on real data transmission, the inferred link performance metrics are more accurate and meaningful.

VII. Related Work

In [25], [12], [3] grouping algorithms were proposed to infer the routing tree topology based on measurements observed at the destination nodes. The authors in [7] and [28] formulated the topology inference problem as a hierarchical clustering problem and developed several hierarchical clustering algorithms to recover the tree topology. The RNJ algorithm studied in this paper is also a grouping type algorithm which recovers the tree topology by recursively joining the neighbors on the tree. The agglomerative joining/grouping/clustering idea has long been used in clustering for building cluster trees [16] and in evolutionary biology for building phylogenetic trees [26]. [11] proposed a Markov Chain Monte Carlo (MCMC) approach to search the most likely tree topologies. Although MCMC reduces the complexity of maximum-likelihood search [12], it may converge slowly, and consistency of such an algorithm is not guaranteed.

The main difference between the algorithms and analytic framework in this paper and those in the previous work is as follows. We have made an important connection between the network inference problem and the phylogenetic inference problem. Unlike previous work, the algorithms in this paper are based on additive metrics, and we have provided quantitative explanation and theoretical justification of the minimum link length and the optimal threshold parameter in connection to consistency and robustness.

VIII. Conclusion

In this paper we addressed the network inference problem of estimating the routing tree topology and link performance in a communication network. We introduced a general framework for designing and analyzing network inference algorithms based on additive metrics using ideas and tools from phylogenetic inference. The framework is applicable to a variety of measurement techniques. Based on the framework we developed and analyzed several distance-based inference algorithms. We

showed that the algorithms are computationally efficient (polynomial-time), consistent (return correct topology and link performance with an increasing sample size), and robust (can tolerate a maximum level of measurement errors). The framework provides powerful tools to infer the structure and dynamics of large-scale communication networks.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their helpful comments.

REFERENCES

[1] K. Atteson, "The performance of neighbor-joining methods of phylogenetic reconstruction," *Algorithmica*, vol. 25, pp. 251–278, 1999.
[2] D. Barry and J. A. Hartigan, "Asynchronous distance between homogenous DNA squences," *Biometrics*, vol. 43, pp. 261–276, Jun. 1987.
[3] A. Bestavros, J. W. Byers, and K. A. Harfoush, "Inference and labeling of metric-induced network topologies," *IEEE Trans. Parallel Distrib. Syst.*, vol. 16, no. 11, pp. 1053–1065, Nov. 2005.
[4] P. Buneman, "The recovery of trees from measures of dissimilarity," *Mathematics in the Archaeological and Historical Sciences, Edinburgh Univ. Press*, pp. 387–395, 1971.
[5] 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.
[6] 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.
[7] R. Castro, M. Coates, and R. Nowak, "Likelihood based hierarchical clustering," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2308–2321, Aug. 2004.
[8] J. T. Chang, "Full reconstruction of Markov models on evolutionary trees: Identifiability and consistency," *Math. Biosci.*, vol. 137, pp. 51–73, 1996.
[9] M. Coates and R. Nowak, "Network loss inference using unicast end-to-end measurement," presented at the ITC Conf. IP Traffic, Modelling and Management, Monterey, CA, Sep. 2000.
[10] 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.
[11] M. Coates, R. Castro, M. Gadhiok, R. King, Y. Tsang, and R. Nowak, "Maximum likelihood network topology identification from edge-based unicast measurements," presented at the ACM Sigmetrics, 2002.
[12] 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.
[13] N. G. Duffiled, F. L. 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.
[14] J. Felsenstein, *Inferring Phylogenies*. New York: Sinauer, 2004.
[15] O. Gascuel and M. Steel, "Neighbor-joining revealed," *Mol. Biol. Evol.*, vol. 23, no. 11, pp. 1997–2000, 2006.
[16] J. Hartigan, *Clustering Algorihtms*. Hoboken, NJ: Wiley, 1975.
[17] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed diffusion for wireless sensor networking," *IEEE/ACM Trans. Netw.*, vol. 11, no. 1, pp. 2–16, Feb. 2003.
[18] Y. Mao, F. R. Kschischang, B. Li, and S. Pasupathy, "A factor graph approach to link loss monitoring in wireless sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 4, pp. 820–829, Apr. 2005.
[19] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and C. Diot, "Traffic matrix estimation: Existing techniques and new directions," presented at the ACM SIGCOMM 2002, 2002.
[20] J. Ni and S. Tatikonda, "A Markov random field approach to multicast-based network inference problems," presented at the IEEE Int. Symp. Information Theory (ISIT), Seattle, WA, Jul. 2006.
[21] J. Ni, S. Tatikonda, and E. Yeh, "A large-scale distributed traffic matrix estimation algorithm," presented at the 49th IEEE Global Telecommunications Conf. (GLOBECOM), San Francisco, CA, Nov. 2006.
[22] J. Ni and S. Tatikonda, "Explicit link parameter estimators based on end-to-end measurements," presented at the 45th Allerton Conf. Communication, Control, and Computing, Monticello, IL, Sep. 2007.
[23] J. Ni, H. Xie, S. Tatikonda, and Y. R. Yang, "Network routing topology inference from end-to-end measurements," presented at the IEEE Conf. Computer Communications (INFOCOM), Phoenix, AZ, Apr. 2008.
[24] 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.
[25] S. Ratnasamy and S. McCanne, "Inference of multicast routing trees and bottleneck bandwidths using end-to-end measurements," presented at the IEEE INFOCOM, Mar. 1999.
[26] 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.
[27] C. Semple and M. Steel, *Phylogenetics*, ser. Volume 22 of Mathematics and Its Applications Series. Oxford, U.K.: Oxford Univ. Press, 2003.
[28] 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.
[29] K. Tamura, M. Nei, and S. Kumar, "Prospects for inferring very large phylogenies by using neigbhor-joining mehtod," *Proc. Nat. Acad. Sci.*, vol. 101, no. 30, pp. 11030–11035, Jul. 2004.
[30] Y. Tsang, M. Coates, and R. Nowak, "Network delay tomography," *IEEE Trans. Signal Process.*, vol. 51, no. 8, pp. 2125–2136, Aug. 2003, Special Issue on Signal Processing in Networking.
[31] Y. Vardi, "Network tomography: Estimating source-destination traffic intensities from link data," *J. Amer. Statist. Assoc.*, vol. 91, no. 433, pp. 365–377, 1996.
[32] Y. Zhang, M. Roughan, C. Lund, and D. Donoho, "An information-theoretic approach to traffic matrix estimation," presented at the ACM SIGCOMM, Aug. 2003.
[33] Q. Zhao, Z. Ge, J. Wang, and J. Xun, "Robust traffic matrix estimation with imperfect information: Making use of multiple data sources," presented at the ACM Sigmetrics, 2006.

Jian Ni (S'02–M'09) received the B.Eng. degree in automation from Tsinghua University, Beijing, in 2001; the M.Phil. degree in electrical and electronic engineering from the Hong Kong University of Science and Technology (HKUST), Hong Kong, in 2003; and the Ph.D. degree in electrical engineering from Yale University, New Haven, in 2008. From 2008 to 2010 he was a Postdoctoral Researcher with the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign. He is currently a Research Staff Member at the IBM T. J. Watson Research Center, Yorktown Heights, NY. His current research interests include performance analysis, algorithm design, communication networks, statistical inference, machine learning, and information extraction.

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.