Network Tomography: recent developments

Rui Castro Mark Coates Gang Liang Robert Nowak Bin Yu*

March 7, 2003

ABSTRACT

Today's Internet is a massive, distributed network which continues to explode in size as e-commerce and related activities grow. The heterogeneous and largely unregulated structure of the Internet renders tasks such as dynamic routing, optimized service provision, service level verification, and detection of anomalous/malicious behavior extremely challenging. The problem is compounded by the fact that one cannot rely on the cooperation of individual servers and routers to aid in the collection of network traffic measurements vital for these tasks. In many ways, network monitoring and inference problems bear a strong resemblance to other "inverse problems" in which key aspects of a system are not directly observable. Familiar signal processing or statistical problems such as tomographic image reconstruction and phylogenetic tree identification have interesting connections to those arising in networking. This article introduces network tomography, a new field which we believe will benefit greatly from the wealth of statistical theory and algorithms. It focuses especially on recent developments in the field including the application of pseudolikelihood methods and tree estimation formulations.

Keywords: Network tomography, pseudo-likelihood, topology identification, tree estimation.

1 Introduction

No network is an island, entire of itself; every network is a piece of an internetwork, a part of the main[1]. Although administrators of small-scale networks can monitor local traffic conditions and identify congestion points and performance bottlenecks, very few networks are completely

*Rui Castro and Robert Nowak are with the Department of Electrical and Computer Engineering, Rice University, Houston, TX; Mark Coates is with the Department of Electrical and Computer Engineering, McGill University, Montreal, Quebec, Canada; Gang Liang and Bin Yu are with the Department of Statistics, University of California, Berkeley, CA.

[1]With apologies to John Donne - Meditation XVII.

isolated. The user-perceived performance of a network thus depends heavily on the performance of an internetwork, and monitoring this internetwork is extremely challenging. Diverse subnetwork ownership and the decentralized, heterogeneous and unregulated nature of the extended internetwork combine to render a coordinated measurement framework infeasible. There is no real incentive for individual servers and routers to collect and freely distribute vital network statistics such as traffic rates, link delays, and dropped packet rates. Collecting all pertinent network statistics imposes an impracticable overhead expense in terms of added computational, communication, hardware and maintenance requirements. Even when data collection is possible, network owners generally regard the statistics as highly confidential. Finally, the task of relaying measurements to the locations where decisions are made consumes exorbitant bandwidth and presents scheduling and coordination nightmares.

Despite this state of affairs, accurate, timely and localized estimates of network performance characteristics are vital ingredients in efficient network operation. With performance estimates in hand, more sophisticated and ambitious traffic control protocols and dynamic routing algorithms can be designed. Quality-of-service guarantees can be provided if available bandwidth can be gauged; the resulting service-level agreements can be verified. Detecting anomalous or malicious behaviour becomes a more achievable task.

Usually we cannot directly measure the aspects of the system that we need in order to make informed decisions. However, we can frequently make useful measurements that do not require special cooperation from internal network devices and do not inordinately impact network load. Sophisticated methods of active network probing or passive traffic monitoring can generate network statistics that indirectly relate to the performance measures we require. Subsequently, we can apply inference techniques, derived in the context of other statistical inverse problems, to extract the hidden information of interest.

This article surveys the field of inferential network monitoring or *network tomography*, highlighting challenges and open problems, and identifying key issues that must be addressed. It builds upon the signal processing survey paper [1] and focuses on recent developments in the

field. The task of inferential network monitoring demands the estimation of a potentially very large number of spatially distributed parameters. To successfully address such large-scale estimation tasks, researchers adopt models that are as simple as possible but do not introduce significant estimation error. Such models are not suitable for intricate analysis of network queuing dynamics and fine time-scale traffic behaviour, but they are often sufficient for inference of performance characteristics. The approach shifts the focus from detailed queuing analysis and traffic modeling [2, 3] to careful design of measurement techniques and large-scale inference strategies.

Mesurement may be passive (monitoring traffic flows and sampling extant traffic) or active (generating probe traffic). In either case, statistical models should be developed for the measurement process. One must assess the temporal and spatial independence of measurements. If existing traffic is being used to sample the state of the network, care must be taken that the temporal and spatial structure of the traffic process does not bias the sample. If probes are used, then the act of measurement must not significantly distort the network state. Design of the measurement methodology must take into account the limitations of the network. As an example, the clock synchronization required for measurement of one-way packet delay is extremely difficult.

Once measurement has been accomplished, statistical inference techniques can be applied to determine performance attributes that cannot be directly observed. When attempting to infer a network performance measure, measurement methodology and statistical inference strategy must be considered *jointly*. In work thus far in this area, a broad array of statistical techniques have been employed: complexity reducing hierarchical statistical models; moment and likelihood based estimation; expectation-maximization and Markov Chain Monte Carlo algorithms. However, the field is still in the embryonic phase, and we believe that it can benefit greatly from the wealth of extant statistical theory and algorithms.

In this article, we focus exclusively on inferential network monitoring techniques that require minimal cooperation from network elements that cannot be directly controlled. Numerous tools

exist for active and passive measurement of networks (see [4] for a survey). The tools measure and report internetwork attributes such as bandwidth, connectivity and delay, but they do not attempt to use the recorded information to infer any performance attributes that have not been directly measured. The majority of the tools depend on accurate reporting by all network elements traversed during measurement.

The article commences by reviewing the area of internetwork inference and tomography, and provide a simple, generalized formulation of the network tomography problem. We then describe a pseudo-likelihood approach to network tomography that addresses some of the scalability limitations of existing techniques. We consider the problem of determining the connectivity structure or topology of a network, relate this task to the problem of hierarchical clustering. We introduce new likelihood-based hierarchical clustering methods and results for identifying network topology. Finally, we identify open problems and provide our vision of future challenges.

2 Network Tomography

2.1 Network Tomography Basics

Large scale network inference problems can be classified according to the type of data acquisition and the performance parameters of interest. To discuss these distinctions, we require some basic definitions. Consider the network depicted in Figure 1. Each node represents a computer terminal, router or subnetwork (consisting of multiple computers/routers). A connection between two nodes is called a *path*. Each path consists of one or more *links* — direct connections with no intermediate nodes. The links may be unidirectional or bidirectional, depending on the level of abstraction and the problem context. Each link can represent a chain of *physical* links connected by intermediate routers. Messages are transmitted by sending *packets* of bits from a *source* node to a *destination* node along a path which generally passes through several other nodes.

Broadly speaking, large scale network inference involves estimating network performance parameters based on traffic measurements at a limited subset of the nodes. Y. Vardi was one of the first to rigorously study this sort of problem and coined the term *network tomography* [5] due to the similarity between network inference and medical tomography. Two forms of network tomography have been addressed in the recent literature: i) link-level parameter estimation based on end-to-end, path-level traffic measurements [6, 7, 8, 9, 10, 11, 12, 13, 14, 15] and ii) sender-receiver path-level traffic intensity estimation based on link-level traffic measurements [5, 16, 17, 18, 19, 20].

In link-level parameter estimation, the traffic measurements typically consist of counts of packets transmitted and/or received between nodes or time delays between packet transmissions and receptions. The time delays are due to both propagation delays and router processing delays along the path. The measured path delay is the sum of the delays on the links comprising the path; the link delay comprises both the propagation delay on that link and the queuing delay at the routers lying along that link. A packet is dropped if it does not successfully reach the input buffer of the destination node. Link delays and occurrences of dropped packets are inherently random. Random link delays can be caused by router output buffer delays, router packet servicing delays, and propagation delay variability. Dropped packets on a link are usually due to overload of the finite output buffer of one of the routers encountered when traversing the link, but may also be caused by equipment down-time due to maintenance or power failures. Random link delays and packet losses become particularly substantial when there is a large amount of cross-traffic competing for service by routers along a path.

In path-level traffic intensity estimation, the measurements consist of counts of packets that pass through nodes in the network. In privately owned networks, the collection of such measurements is relatively straightforward. Based on these measurements, the goal is to estimate how much traffic originated from a specified node and was destined for a specified receiver. The combination of the traffic intensities of all these origin-destination pairs forms the origin-destination *traffic matrix*. In this problem not only are the node-level measurements inherently

random, but the parameter to be estimated (the origin-destination traffic matrix) must itself be treated not as a fixed parameter but as a random vector. Randomness arises from the traffic generation itself, rather than perturbations or measurement noise.

The inherent randomness in both link-level and path-level measurements motivates the adoption of statistical methodologies for large scale network inference and tomography. Many network tomography problems can be roughly approximated by the (not necessarily Gaussian) linear model

$$\mathbf{Y_t} \;=\; \mathbf{A}\mathbf{X_t} + \boldsymbol{\epsilon}, \tag{1}$$

where: $\mathbf{Y_t}$ is a vector of measurements, e.g., packet counts or end-to-end delays, recorded at a given time t at a number of different measurement sites, $\mathbf{A}$ is a *routing matrix*, $\boldsymbol{\epsilon}$ is a noise vector, and $\mathbf{X_t}$ is a vector of time-dependent packet parameters, e.g. mean delays, logarithms of packet transmission probabilities over a link, or the random origin-destination traffic vector. Typically, but not always, $\mathbf{A}$ is a binary matrix (the i, j-th element is equal to '1' or '0') that captures the topology of the network. In this paper, we consider the problems of estimating $\mathbf{X_t}$ or $\mathbf{A}$ based on the observations $\mathbf{Y_t}$.

What sets the large scale network inference problem (1) apart from other network inference problems is the potentially very large dimension of $\mathbf{A}$ which can range from a half a dozen rows and columns for a few packet parameters and a few measurement sites in a small local area network, to thousands or tens of thousands of rows and columns for a moderate number of parameters and measurements sites in the Internet. The associated high dimensional problems of estimating $\mathbf{X_t}$ are specific examples of *inverse problems*. Inverse problems have a very extensive literature [21]. Solution methods for such inverse problems depend on the nature of the noise $\boldsymbol{\epsilon}$ and the $\mathbf{A}$ matrix and typically require iterative algorithms since they cannot be solved directly. In general, $\mathbf{A}$ is not full-rank, so that identifiability concerns arise. Either one must be content to resolve only linear combinations of the parameters or one must employ statistical means to introduce regularization and induce identifiability. Both tactics are utilized in the examples in later sections of the article. In most of the large scale Internet inference and

tomography problems studied to date, the components of the noise vector $\boldsymbol{\epsilon}$ are assumed to be approximately independent Gaussian, Poisson, binomial or multinomial distributed. When the noise is Gaussian distributed with covariance independent of $\mathbf{AX_t}$, methods such as recursive linear least squares can be implemented using conjugate gradient, Gauss-Seidel, and other iterative equation solvers. When the noise is modeled as Poisson, binomial, or multinomial distributed more sophisticated statistical methods such as reweighted non-linear least squares, maximum likelihood via expectation-maximization (EM), and maximum a posteriori (MAP) via Monte Carlo Markov Chain (MCMC) algorithms are applicable.

3 Pseudo-likelihood Approaches

In developing methods to perform network tomography, there is a trade-off between statistical efficiency (accuracy) and computational overhead. In the past, researchers have addressed the extreme computational burden posed by some of the tomographic problems, developing suboptimal but lightweight algorithms, including a fast recursive algorithm for link delay distribution inference in a multicast framework [12] and a method-of-moments approach for OD matrix inference [22]. More accurate but computationally burdensome approaches have also been explored, including maximum likelihood methods [23, 8, 9], but in general, they are too intensive computationally for any network of reasonable scale.

More recently, we have proposed a unified pseudo likelihood (PL) approach [19, 24] that eases the computational burden but maintains good statistical efficiency. The idea of modifying likelihood is not new, and many modified likelihood models have been proposed. For example, pseudo-likelihood [25, 26] for Markov Random Field (MRF) by Besag (1974), partial likelihood [27] for hazards regression by Cox (1973), and quasi-maximum likelihood [28] for finance models by White (1994). In this section, we decribe the pseudo-likelihood approach. We explore two concrete examples : i) internal link delay distribution inference through multicast end-to-end measurements; ii) origin-destination (OD) matrix inference through link traffic counts (the OD

matrix specifies the volume of traffic between a source and a destination).

The network tomography model we consider in this section is a special case of (1), in which the error term $\boldsymbol{\epsilon}$ is omitted for further simplification. Hence the model can be rewritten as

$$\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{X}, \tag{2}$$

where $\mathbf{X} = (X_1, ..., X_J)'$ is a J-dimensional vector of network dynamic parameters, e.g., link delay, traffic flow counts at a particular time interval, $\mathbf{Y} = (Y_1, ..., Y_I)'$ is an I-dimensional vector of measurements, and $\boldsymbol{A}$ is an $I \times J$ routing matrix.

As mentioned before, $\mathbf{A}$ is not full rank in a general network tomography scenario, where typically $I \ll J$; hence, constraints have to be introduced to ensure the identifiability of the model. A key assumption is that all components of $\mathbf{X}$ are independent of each other. Such an assumption does not hold strictly in a real network due to the temporal and spatial correlations between network traffic, but it is a good first-step approximation. Furthermore, we assume that

$$X_j \sim f_j(\theta_j),\ j = 1, ..., J, \tag{3}$$

where f_j is a density function with θ_j being its parameter. Then the parameter of the whole model is $\theta = (\theta_1, ..., \theta_J)$.

The main idea of the pseudo-likelihood approach is to decompose the original model into a series of simpler subproblems by selecting pairs of rows from the routing matrix $\mathbf{A}$ and to form the pseudo-likelihood function by multiplying the marginal likelihoods of such subproblems. Let S denote the set of subproblems by selecting all possible pairs of rows from the routing matrix $\boldsymbol{A}$: $S = \{s = (i_1, i_2) : 1 \leq i_1 < i_2 \leq I\}$. Then for each subproblem $s \in S$, we have

$$\boldsymbol{Y}^s = \boldsymbol{A}^s \boldsymbol{X}^s, \tag{4}$$

where $\boldsymbol{X}^s$ is the vector of network dynamic components involved in the given subproblem s,

$\boldsymbol{A}^s$ is the corresponding sub-routing matrix, and $\boldsymbol{Y}^s = (Y_{i_1}, Y_{i_2})'$ is the observed measurement vector of s. Let θ^s be the parameter of s, and $p^s(\boldsymbol{Y}^s; \theta^s)$ be its marginal likelihood function. Usually subproblems are dependent, but ignoring such dependencies, the pseudo-likelihood function can be written as the product of marginal likelihood functions of all subproblems, i.e., given observation $y_1, ..., y_T$, the pseudo log-likelihood function is defined as

$$L^p(y_1, ..., y_T; \theta) = \sum_{t=1}^{T} \sum_{s \in S} l^s(y_t^s; \theta^s), \tag{5}$$

where $l^s(\boldsymbol{Y}^s; \theta^s) = \log p^s(\boldsymbol{Y}^s; \theta^s)$ is the log-likelihood function of subproblem s. Maximizing the pseudo log-likelihood function L^p gives the maximum pseudo likelihood estimate (MPLE) of parameter θ. Maximizing the pseudo likelihood is not an easy task because $L^p(y_1, ..., y_T; \theta)$ is a summation of many functions. Since the maximization of pseudo-likelihood function is a typical missing value problem, a pseudo-EM algorithm [19, 24] (a variant of the EM algorithm) is employed to maximize the function $L^p(y_1, ..., y_T; \theta)$. Let $l^s(\boldsymbol{X}^s; \theta^s)$ be the log-likelihood function of a subproblem s given the complete data $\boldsymbol{X}^s$ and $\theta^{(k)}$ be the estimate of θ obtained in the kth step. The objective function $Q(\theta, \theta^{(k)})$ to be maximized in the $(k+1)$th step of the pseudo-EM algorithm is defined as

$$Q(\theta, \theta^{(k)}) = \sum_{s \in S} \sum_{t=1}^{T} \mathrm{E}_{\theta^{s(k)}} \left(l^s(x_t^s; \theta^s) | y_t^s \right), \tag{6}$$

which is obtained by assuming the independence of subproblems in the expectation step. The starting point of the pseudo-EM algorithm can be arbitrary, but just as in the EM algorithm, care needs to be taken to ensure that the algorithm does not converge to a local maxima.

There are several points worth noting in constructing the pseudo-likelihood function: i) selecting three or more rows each time may also be reasonable to construct a pseudo-likelihood function, but there is a trade-off between the computational complexity incurred and the estimation efficiency achieved by taking more dependence structures into account. The experience with the two examples we will discuss later shows that selecting two rows each time gives sat-

isfactory estimation results while keeping the computational cost within a reasonable range; ii) currently all possible pairs are selected to construct the pseudo-likelihood function, but a subset can be judiciously chosen to reduce the computation.

In summary, the pseudo-likelihood approach keeps a good balance between the computational complexity and the statistical efficiency of the parameter estimation. Even though the basic idea of divide-and-conquer is not new, it is very powerful when combined with pseudo-likelihood for large network problems.

3.1 Example: Multicast Delay Distribution Inference

The Multicast-based Inference of Network-internal Characteristics (MINC) Project [6] pioneered the use of multicast probing for network delay distribution estimation, and a similar approach through unicast end-to-end measurements can be found in [9]. Consider a general multicast tree depicted in Figure 1. Each node is labeled with a number, and we adopt the convention that link i connects node i to its parental node. Each probing packet with a time stamp sent from root node 0 will be received by all end receiver computers 4–7. For any pair of receivers, each packet experiences the same amount of delay over their common path. For instance, copies of the same packet received at receiver 4 and 5 experience the same amount of delay on their common links 1 and 2. Measurements are made at end receivers, so only the aggregated delays over the paths from root to end receivers are observed.

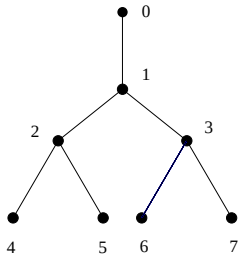


Figure 1: An arbitrary virtual multicast tree with four receivers

Due to the aggregation of the measured delays, model (2) can be naturally applied to the

problem of the multicast internal link delay distribution inference. For each probing packet, $\mathbf{X}$ is the vector of unobserved delays over each link, and $\mathbf{Y}$ is the vector of observed path-level delays at each end receiver. $\mathbf{A}$ is an $I \times J$ routing matrix determined by the multicast spanning tree, where I is the number of end receivers and J the number of internal links. For the multicast tree depicted in Figure 1, (2) can be written as:

$$\begin{pmatrix} Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_7 \end{pmatrix},$$

where $Y_1, ..., Y_4$ are the measured delays at end receivers $4, ..., 7$ and $X_1, ..., X_7$ are the delays over internal links ending at nodes 1, ..., 7.

Each link has a certain amount of minimal delay (the propagation delay on the link), which is assumed to be known beforehand. After compensating for the minimal delay of each link, a discretization scheme is imposed on link-level delay by Lo Presti *et al.* (1999) such that X_j takes finite possible values $\{0, q, 2q, ..., mq, \infty\}$, where q is the bin width and m is a constant. Therefore, each X_j is an independent multinomial variable with $\theta_j = (\theta_{j0}, \theta_{j1}, ..., \theta_{jm}, \theta_{j\infty})$. When the delay is infinite, it implies the packet is lost during the transmission. The choice of m enables us to decide how fine we want to approximate the true delay distributions. In order to ensure identifiability, we only consider canonical multicast trees [12] defined as those satisfying

$$\theta_{j0} = P(X_j = 0) > 0, \ j = 1, ..., J,$$

i.e., each individual packet has a positive probability to have zero delay over any internal link.

For the problem of multicast internal delay inference, the maximum likelihood method is usually infeasible for networks of realistic size, because the likelihood function involves finding all possible internal delay vectors $\boldsymbol{X}$, which can account for each observed delay vector $\mathbf{Y}$. We can show that the computational complexity grows at a non-polynomial rate. Lo Presti

et al.'s recursive algorithm [12] is a computationally efficient method for estimating internal delay distributions by solving a set of convolution equations. Our pseudo-likelihood approach is motivated by the decomposition of multicast spanning trees depicted in Figure 2. A virtual two-leaf subtree is formed by only considering two receivers R_1, R_3 in the original multicast tree. The marginal likelihood function of the virtual two-leaf subtree is tractable because of its simple structure. For a multicast tree with I end receivers, there are total $I(I-1)/2$ subtrees: different subtrees contain delay distribution information on different virtual links. Combining all subproblems by ignoring their dependencies enables us to recover link delay distributions. Since forming the subtree is equivalent to selecting two rows from the routing matrix $\mathbf{A}$, the pseudo-likelihood method is applicable to the general network tomography model (2).

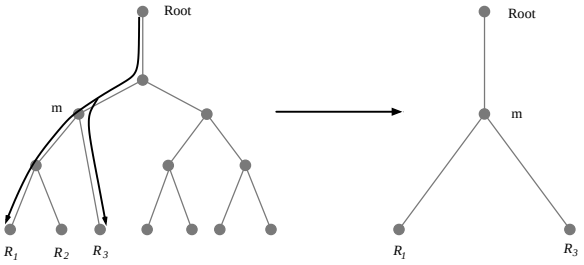


Figure 2: Pseudo-Likelihood: subtree decomposition

Given the observed end-to-end multicast measurements $\{y_1, ..., y_T\}$, the pseudo log-likelihood function can be written as

$$L^p(y_1, ..., y_T; \theta) = \sum_{s \in S} \sum_{t=1}^{T} \log p(\boldsymbol{Y}^s = y_t^s | \theta^s),$$

where $p(\boldsymbol{Y}^s = y_t^s | \theta^s)$ is the probability of the delay measurement $\boldsymbol{Y}^s$ of subtree s being y_t^s when its link delay distributions are θ^s. The pseudo log-likelihood function is maximized in an EM fashion with small variations.

We evaluate the performance of the pseudo-likelihood methodology by model simulations carried out on a 4-leave multicast tree depicted in Figure 1. Due to the small size of the multicast tree, the MLE method is possible to implement, so we can compare the performance

of MPLE with that of MLE and also with that of the recursive algorithm of [12]. For each link, the bin size $q = 1$ and the number of bins m is set to be 14. During each simulation, 2000 i.i.d. multicast delay measurements are generated, with each internal link having an independent discrete delay distribution. Figure 3 shows the delay distribution estimates of three arbitrarily selected links along with their true delay distributions in one such experiment. The plot shows that both MPLE and MLE capture most of the link delay distributions and their performance is comparable, whereas the recursive algorithm sometimes gives estimate far from the truth.

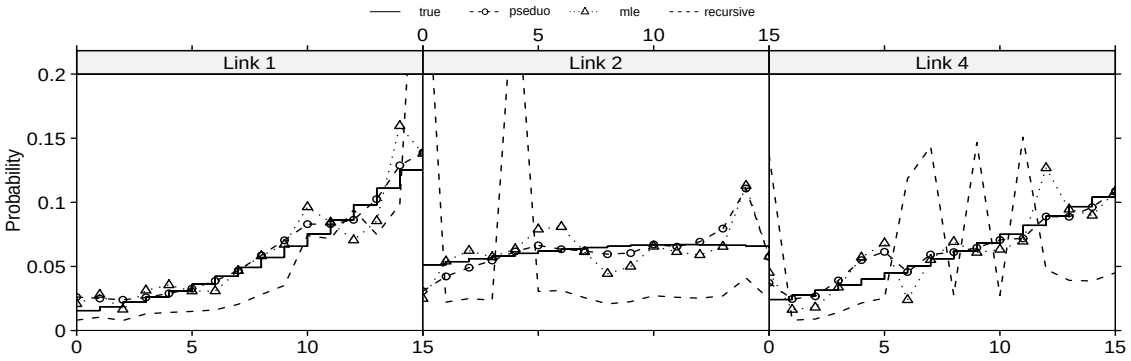


Figure 3: Delay distribution estimates of 3 arbitrarily selected internal links: Link 1, Link 2 and Link 4. Solid step function is the true distribution, dash line with circle is MPLE, dotted line with triangle is MLE, and dash line only is recursive estimate.

A further comparison is illustrated in Figure 4, which shows the L_1 error norm of MLE and MPLE for each link, as averaged over 30 independent simulations. For each link, the L_1 error norm is simply the sum of the absolute difference between probability estimates and the true probabilities. As a common measure of the performance of density estimates, the L_1 error norm enjoys several theoretical advantages as discussed in [29]. The plot shows that MLE and MPLE have comparable estimation performance for tracking link delay distributions, while the recursive algorithm has much larger L_1 errors on all links. Meanwhile, we can see that MPLE has smaller SD on L_1 error norm than MLE on all links, implying that MPLE is more robust than MLE. This is because the pseudo likelihood function, which is a product of less complex likelihood functions on subproblems, has a nicer surface than the full likelihood function [30].

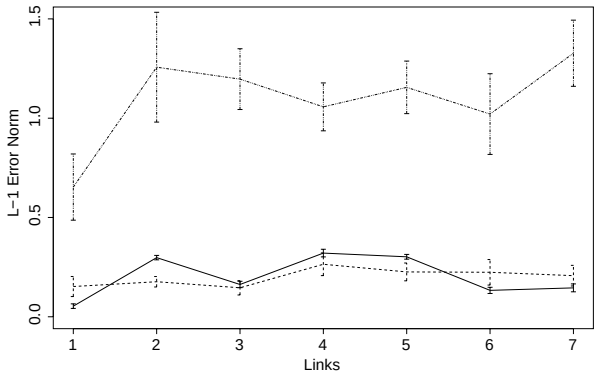


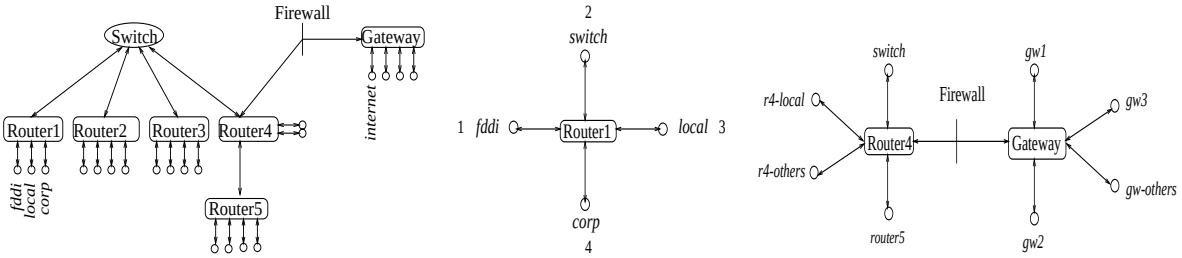
Figure 4: Link L_1 error norm averaged over 30 simulations: solid line is MPLE, dashed line is MLE, and dotted line is recursive algorithm. For each link, the vertical bar shows the SD of the L_1 error norm for the given link.

3.2 Example: Origin-Destination Traffic Matrix Inference

Vardi [22] was the first to study the origin-destination (OD) traffic matrix inference problem through link traffic counts at router interfaces: the observed are the link counts at router interfaces and the OD traffic variables to be estimated are linear aggregations of these link counts (his work was originated in 1993, but appeared in 1996, see [22]). Assuming i.i.d. Poisson distributions for the OD traffic byte counts on a general network topology, he specifies the identifiability of the Poisson model and develops an expectation-maximization (EM) algorithm to estimate Poisson parameters in both deterministic and Markov routing schemes. In order to reduce the computational complexity of the EM algorithm, he proposes a moment estimation method and briefly discusses the normal model as an approximation to the Poisson model. Follow-up works treat the special case involving a single set of link counts: Vanderbei and Iannone apply the EM algorithm [31], and Tebaldi and West have a Bayesian perspective and a Markov Chain Monte Carlo implementation [16].

Cao *et al.* [23] use real data to revise the Poisson model and to address the non-stationary aspect of the problem. They represent link count measurements as summations of various OD counts that are modeled as independent random variables. Even though the Transport Control Protocol (TCP) feedback creates dependence, direct measurements of OD traffic indicate that the dependence between traffic in opposite directions is weak. This renders the independence

assumption a reasonable approximation. Time-varying traffic matrices estimated from a sequence of link counts are validated by comparing the estimates with actual OD counts that were collected by running Cisco's NetFlow software on a small network depicted in Figure 5(b). Such direct point-to-point measurements are often not available because they require additional router CPU resources, can reduce packet forwarding efficiency, and involve a significant administrative burden when used on a large scale.



(a) (b) (c)

Figure 5: (a) A router network at Lucent Technologies, (b) Network topology around *Router1*, (c) A two-router network around *Router4* and *Gateway*.

The network tomography model specified by (2) is applicable to the OD matrix inference through link traffic counts since the observed link traffic counts are linear aggregations of the unobserved OD variables to be estimated. Here, $\mathbf{Y} = (Y_1, Y_2, ..., Y_I)'$ is the vector of observed traffic byte counts measured on each link interface during a given time interval and $\mathbf{X} = (X_1, X_2, ..., X_J)'$ is the corresponding vector of unobserved true OD traffic byte counts at the same time period. $\mathbf{X}$ is called OD traffic matrix, even though it is arranged as a column vector for notational convenience. Under a fixed routing scheme, $\mathbf{Y}$ is determined uniquely by $\mathbf{X}$ through the $I \times J$ routing matrix $\mathbf{A}$, in which I is the number of measured incoming/outgoing unidirectional links and J is the number of possible OD pairs.

Each component of $\mathbf{X}$ is assumed to be independent normally distributed, satisfying the following mean-variance relationship: $X_j \sim N(\lambda_j, \phi\lambda_j^c)$ independently, where ϕ is a positive scalar applicable to all OD pairs and c is a power constant. It implies that

$$\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{X} \sim N(\boldsymbol{A}\lambda, \boldsymbol{A}\Sigma\boldsymbol{A}'), \tag{7}$$

where $\lambda = (\lambda_1, \cdots, \lambda_J)$ and $\Sigma = \phi\,\mathrm{diag}(\lambda_1^c, \cdots, \lambda_J^c)$. So the parameter of the full model is $\theta = (\phi, \lambda)$. The mean-variance relationship is a key assumption to ensure the identifiability of the normal model. It implies that an OD pair with large traffic byte counts tends to have large variance with the same scale factor ϕ. For the power constant c, both $c = 1$ and 2 work well with the Lucent network data as shown in [18, 23]. Because $c = 1$ or $c = 2$ give similar results, in this paper, we use $c = 1$ as in [18]. But note that the pseudo-likelihood method can deal with $c = 2$ without any additional technical difficulties. Then given observed link traffic count vectors $\{y_1, ..., y_T\}$, the pseudo log-likelihood function can be written as

$$L^p(\lambda, \Sigma) \propto -\frac{1}{2} \sum_{s \in S} \sum_{t=1}^{T} \left\{ -\log |\boldsymbol{A}^s \Sigma_s \boldsymbol{A}^{s\prime}| + (y_t^s - \boldsymbol{A}^s \lambda^s)' (\boldsymbol{A}^s \Sigma_s \boldsymbol{A}^{s\prime})^{-1} (y_t^s - \boldsymbol{A}^s \lambda^s) \right\},$$

where for a subproblem s, λ^s is its mean traffic vector, Σ_s is its covariance matrix, and $\boldsymbol{A}^s$ is the sub routing matrix. The maximization of the pseudo log-likelihood function is realized by the pseudo-EM algorithm as well.

Cao *et al.* [23] address the non-stationarity of the data using a local likelihood model; that is, for any given time interval t, analysis is based on a likelihood function derived from the observations within a symmetric window of size w around t (e.g., in the experiments described below, $w = 11$ corresponds to observations within about an hour in real time). Within this window, an i.i.d. assumption is imposed (as a simplified and yet practical way to treat the approximately stationary observations within the window). Maximum-likelihood estimation is carried out for the parameter estimation via a combination of the EM algorithm and a second-order global optimization routine. The component-wise conditional expectations of the OD traffic, given the link traffic, estimated parameters, and the positivity constraints on the OD traffic, are used as the initial estimates of the OD traffic. The linear equation $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ is enforced via the iterative proportional fitting algorithm [23, 32] to obtain the final estimates of the OD traffic. The positivity and the linear constraints are very important final steps to get reliable estimates of the OD traffic, in addition to the implicit regularization introduced by the i.i.d. statistical model. To smooth the parameter estimates, a random walk model is also

applied in [23] to the logarithm of the parameters λ's and ϕ over the time windows.

Even though the full likelihood method described in [23] uses all available information to estimate parameter values and the OD traffic vector $\boldsymbol{X}$, it does not scale to networks with many nodes. In general, if there are N_e edge nodes, the number of floating point operations needed to compute the MLE is at least proportional to N_e^5 after exploiting sparse matrix calculation in each iteration. Assuming that the average number of links between an OD pair is $O(\sqrt{N_e})$, it can be shown that the overall computational complexity of each iteration of the pseudo-EM algorithm is $O(N_e^{3.5})$. Compared with the complexity of the full likelihood $O(N_e^5)$, the pseudo-likelihood approach reduces the computational complexity considerably. Moreover, the pseudo-likelihood approach fits into the framework of the distributed computing, which is beneficial to realistic applications.

First, to compare with works in [23] we choose the same raw network OD traffic data collected on Feb. 22, 1999 for the *Router1* network depicted in Figure 5(b). Figures 6 and 7 show the estimated OD traffic from MPLE and MLE based on the link traffic for the subnetwork along with the validation OD traffic via NetFlow. Figure 6 gives the full scale plot and Figure 7 is the zoomed-in scale (20x). From the plot, we can see that both estimated OD traffic from MPLE and MLE agrees well with the NetFlow measured OD traffic for large measurements, but not so well for small measurements where the Gaussian model is a poor approximation. From the point of view of traffic engineering, it is adequate that the large traffic flows are inferred accurately. Hence for tasks such as planning and provisioning activities OD traffic estimates could be used as inexpensive substitutes for direct measurements. The performances of MPLE and MLE are comparable in this case, but the computation of the MPLE is faster than MLE. For this example, the computations are carried out using R 1.5.0 [33] on a 1G Hz laptop: it takes about 12 seconds for computing the MPLE, and about 49 seconds for the MLE in producing Figure 6.

Second, in order to assess the performance of MPLE more thoroughly, simulations are carried out on some larger networks through network simulator (NS) [34]: i) a two-router network

depicted in Figure 5(c); ii) the Lucent network illustrated in Figure 5(a), which comprises 21 end nodes and 27 links. From the simulation results (plots not shown), we see that both pseudo- and full-likelihood methods capture the dynamics of the simulated OD traffic under the zoomed-in scale. Table 1 summaries the execution time for both pseudo- and full-likelihood approaches under the three different settings. From the table, we can see that the pseudo-likelihood approach speeds up the computation without losing much estimation performance, so it is more scalable to larger networks.

Network Topology	Number of Edge Nodes	MPLE Time (sec)	MLE Time (sec)
Figure 5(b)	4	12	49
Figure 5(c)	8	18	88
Figure 5(a)	21	151	2395

Table 1: Execution times of MPLE and MLE on router networks of different sizes.

4 Topology Identification

In the previous section it was assumed that the network topology was known; this knowledge is essential for successful application of the techniques described. In most situations the topology is obtained using tools such as `traceroute` that rely on close cooperation from the network internal devices such as routers. When these tools are used, the topology can only be determined if the network is functioning properly and network elements are prepared to cooperate and reveal themselves. These conditions are often not met and are becoming more uncommon as the Internet grows in size and speed. There is little motivation for extremely high-speed or heavily-loaded switches to spend time processing extraneous signalling packets.

It is therefore desirable to develop a method for estimating topology that uses only measurements taken at the network edge, obtained without cooperation from internal devices. We consider a single source, communicating with multiple receivers. There is thus a finite set of traffic flows, each corresponding to a specific source-destination pair. The routing is assumed to be constant over a measurement period, so the total network traffic traverses a fixed set of

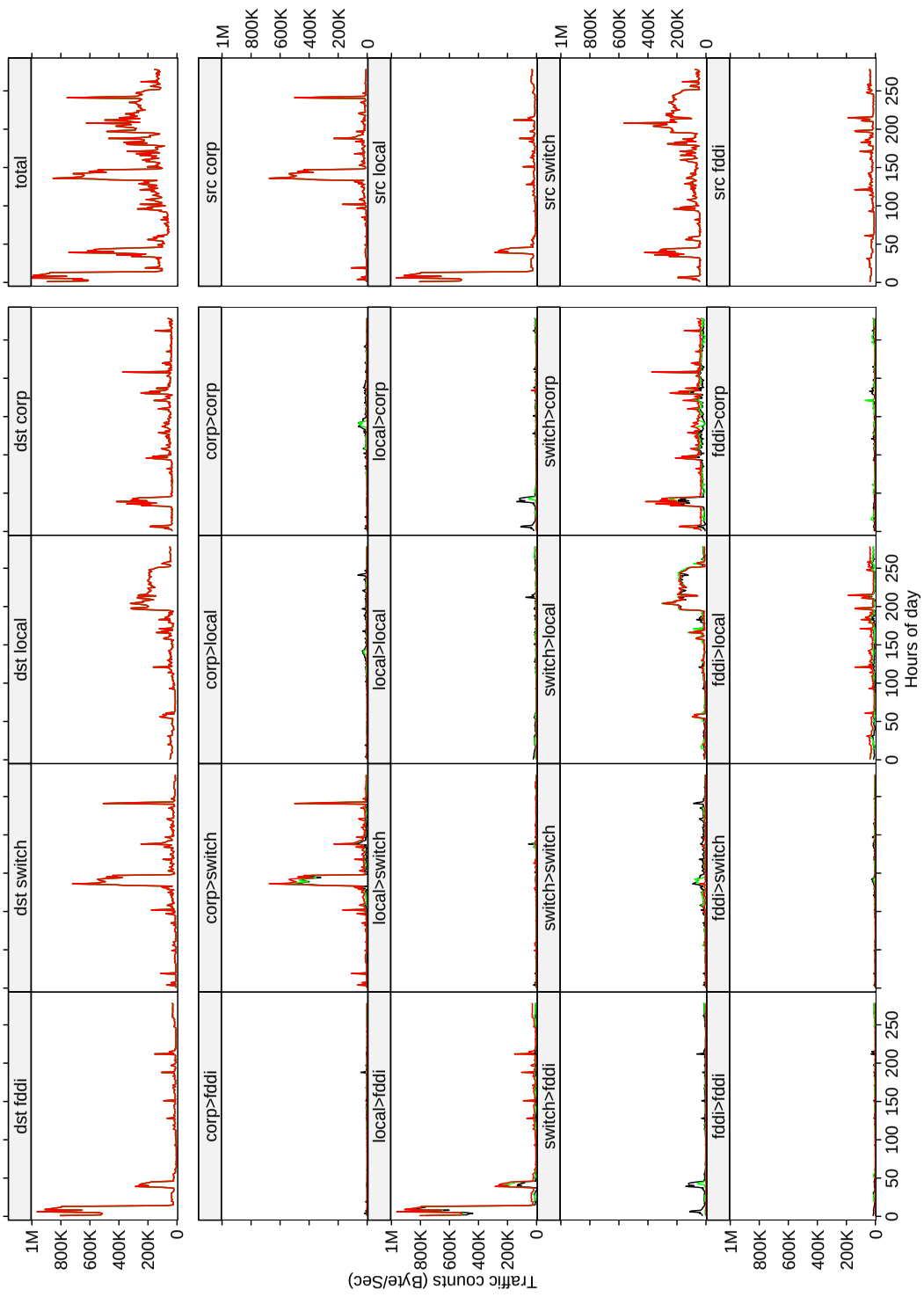


Figure 6: Full scaled OD traffic counts estimates $\hat{x}_t$ obtained from pseudo and full likelihood methods against the true OD traffic counts for 4 nodes network around *Router1*.

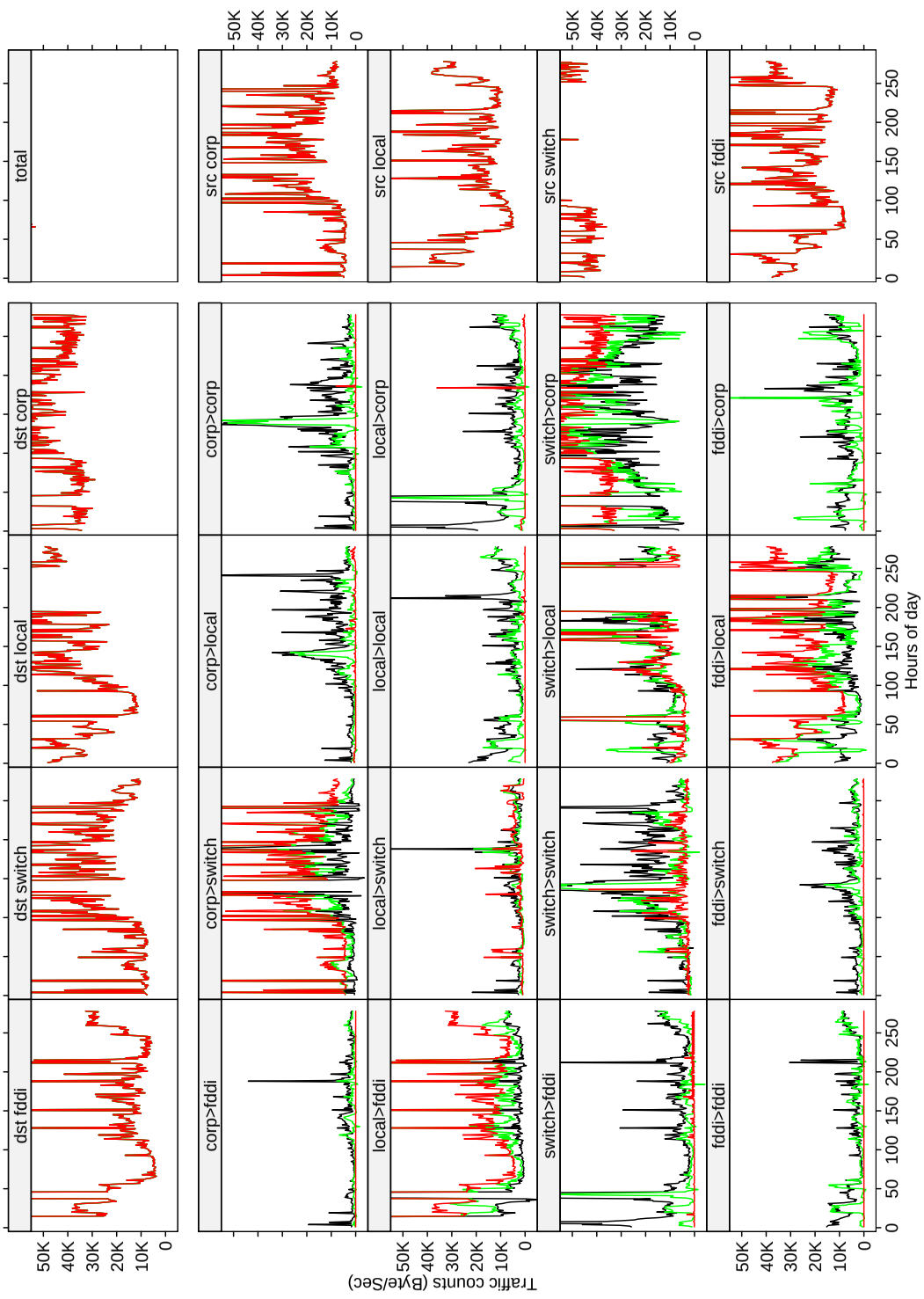


Figure 7: OD traffic counts estimates $\hat{x}_t$ obtained from pseudo and full likelihood methods against the true OD traffic counts for 4 nodes network around *Router1*. The plot has been zoomed-in 20 time to illustrate the detailed features.

network devices. The logical network topology can be represented as a graph. Each vertex represents a physical network device where traffic branching occurs, that is where two or more source-destination traffic flows diverge. The set of vertices thus corresponds to a subset of the traversed physical devices. An edge is included between two vertices if traffic travels between the corresponding network devices and does not pass through any other devices in the included subset. Each edge corresponds to a connection between two physical devices, but the connection may include several network devices where no traffic branching occurs. We assume that the routes from the sender to the receivers are fixed during the measurement period, in which case the topology is a tree-structured graph, as in Figure 8.

Recall equation (1): in the topology identification problem the quantity of interest is $\boldsymbol{A}$, the routing matrix. Note that the entries of this matrix are only 0 or 1. The measurements $\mathbf{Y_t}$ are obtained through special measurement techniques described below, and the *partial ordering* of $\mathbf{Y_t}$ can be used to determine $\boldsymbol{A}$. The matrix estimation formulation above is not well-suited to the topology identification problem, so below we formulate it as a tree estimation exercise.

One can regard the topology discovery problem as a hierarchical clustering exercise: within such a framework one wants to identify clusters of receivers that share certain properties. These clusters can be represented by a dendritic tree. Hierarchical clustering has been used in a variety of areas and are particularly popular for document clustering [35, 36, 37, 38].

4.1 Problem Statement

We formulate the topology identification problem as a tree estimation exercise. Let $\mathcal{T} = (V, L)$ denote a rooted tree with nodes V and directed links L (we consider a strongly acyclic tree, that is, if we disregard the direction of the edges the graph is still acyclic). Denote the root node (corresponding to the sender host) by 0. Denote by R the leaf nodes (corresponding to the receiver hosts). Each leaf node corresponds to one receiver in the receiver set R, and the root node corresponds to the sender host. For example, in Figure 8, $R = \{5, 8, 9, 12, 14, 15, 16, 17, 18, 19\}$.

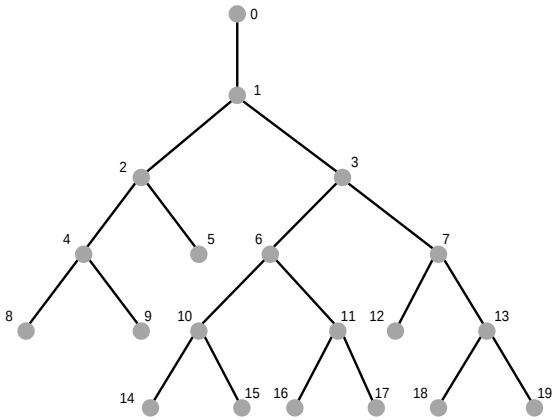


Figure 8: A binary logical tree topology

Every node has at least two descendants, apart from the root node (which has one, denoted by 1), and the leaf nodes (which have none). If all internal nodes have exactly two descendants then the tree is called binary. For each internal node let $f(i),\ i \in V \setminus \{0\}$, denote the parent of i, e.g. $f(10) = 6$. Denote by $\mathcal{P}_i$ the (unique) path from the source node to i, e.g., $\mathcal{P}_{12} = \{0, 1, 3, 7, 12\}$. Let $a(i, j),\ i, j \in V$, denote the nearest ancestor shared by the pair of nodes (i, j), e.g., $a(15, 17) = 6$.

Consider an arbitrary tree $\mathcal{T} = (V, L)$. For each node in the tree we can associate a metric value $\gamma_k \in \mathbb{R},\ k \in V$. This metric value is related to the extent (number of links) of the (unique) path from the root to node k. Another interpretation is that the metric value reflects the similarity of the subset of receivers that share node k as an ancestor. For each pair of input elements $i, j \in R$ we can consider the metric value of the nearest ancestor, $\gamma_{a(i,j)}$. Define $\gamma_{ij} \equiv \gamma_{a(i,j)}$. We regard this metric value as a measure of similarity of nodes i and j. The value of γ_{ij} can be regarded also as a measure of the extent of the shared portion of the paths to i and j. Notice that the tree structure imposes restrictions on γ_{ij}; for example, if two different pairs of nodes have the same shared paths then their similarity values must be the same, e.g. consider the tree in Figure 8; The pairs of nodes $(14, 16)$ and $(15, 17)$ have the same shared paths, thus $\gamma_{14\ 15} = \gamma_{15\ 17}$.

Define the matrix $\boldsymbol{\gamma} = (\gamma_{ij} : i, j \in R)$. According to the discussion above, in order for $\boldsymbol{\gamma}$ to be compatible with the tree structure it must belong to the set

$$\boldsymbol{\Gamma}(\mathcal{T}) = \{\boldsymbol{\gamma} : \gamma_{ij} = \gamma_{kl} \text{ if } a(i,j) = a(k,l), \forall i, j, k, l \in R\} \ . \tag{8}$$

To ensure identifiability of the tree $\mathcal{T}$ we require the metrics $\boldsymbol{\gamma}$ to satisfy the

Monotonicity Property: Let $i, j \in V \setminus \{0, R\}$ be any two nodes and let $\mathcal{P}_i$, $\mathcal{P}_j$ denote the paths from the root to i and j respectively . If $\mathcal{P}_i$ is a proper subpath of $\mathcal{P}_j$ then $\gamma_i < \gamma_j$.

Knowledge of the metric values for each pair of elements of R (i.e. knowledge of matrix $\boldsymbol{\gamma}$) and the requirement of the monotonicity property are sufficient for identification of the underlying topology. For example, referring to Figure 8, the metric $\gamma_{18\ 19}$ will be greater than $\gamma_{i\ 19}$ for all $i \in R \setminus \{18, 19\}$, revealing that nodes 18 and 19 have a common parent in the logical tree. This property can be exploited in this manner to devise simple and effective bottom-up merging algorithm that identifies the complete, logical topology [13, 39, 40, 41]. These same techniques are used in agglomerative hierarchical clustering methods [42, 43, 44].

For a given tree $\mathcal{T}$, the set of all metrics satisfying the monotonicity property is defined as

$$\mathcal{G}(\mathcal{T}) = \left\{\boldsymbol{\gamma} \in \boldsymbol{\Gamma}(\mathcal{T}) : \gamma_{f(k)} < \gamma_k, \ \forall k \in W(\mathcal{T})\right\}, \ W(\mathcal{T}) = V \setminus \{0, 1, R\} \ . \tag{9}$$

The set $W(\mathcal{T}) \in V$ is the set of all the internal nodes, that is, all nodes except the root and the leaf nodes.

4.2 Likelihood Formulation

In general, we do not have access to the exact pairwise metric values, but can only measure a noisy and distorted version, usually obtained by actively probing the network. In this section,

we describe a technique for generating a topology estimate based on the noisy observations. For a given unknown tree $\mathcal{T}$ let $\mathbf{X} \equiv \{X_{ij} : i,j \in R,\ i \neq j\}$, where each X_{ij} is a random variable parameterized by $\boldsymbol{\gamma} \equiv (\gamma_{ij}) \in \boldsymbol{\Gamma}(\mathcal{T})$. Let $p(\mathbf{x}|\boldsymbol{\gamma})$ denote the probability density function of $\mathbf{X}$, parameterized by $\boldsymbol{\gamma} \in \boldsymbol{\Gamma}(\mathcal{T})$, with respect to some dominating measure. A sample $\mathbf{x} \equiv \{x_{ij} : i,j \in R,\ i \neq j\}$ of $\mathbf{X}$ is observed. When $p(\mathbf{x}|\boldsymbol{\gamma})$ is viewed as a function of $\mathcal{T}$ and $\boldsymbol{\gamma} \in \boldsymbol{\Gamma}(\mathcal{T})$ it is called the likelihood of $\mathcal{T}$ and $\boldsymbol{\gamma} \in \boldsymbol{\Gamma}(\mathcal{T})$. The maximum likelihood tree estimate is given by

$$\mathcal{T}^* = \arg\max_{\mathcal{T}\in\mathcal{F}} \log \sup_{\boldsymbol{\gamma}\in\mathcal{G}(\mathcal{T})} p(\mathbf{x}|\boldsymbol{\gamma}) \,, \tag{10}$$

where $\mathcal{F}$ denotes the *forest* of all possible trees with leafs R. If the maximizer of the above expression is not unique define $\mathcal{T}^*$ as one of the possible maximizers. In many situations we are not interested in $\widehat{\boldsymbol{\gamma}}(\mathbf{x})$, an estimate of $\boldsymbol{\gamma}$ from the measurements. Hence we can regard $\boldsymbol{\gamma}$ as nuisance parameters. In that case (10) can be interpreted as a maximization of the profile likelihood [45]

$$\mathcal{L}(\mathbf{x}|\mathcal{T}) \equiv \sup_{\boldsymbol{\gamma}\in\mathcal{G}(\mathcal{T})} p(\mathbf{x}|\boldsymbol{\gamma}) \tag{11}$$

The solution of (10) is referred to as the Maximum Likelihood Tree (MLT).

Consider now some more structure in the log-likelihood $\log p(\mathbf{x}|\boldsymbol{\gamma})$: Assume the random variables X_{ij} are independent and have densities $p(x_{ij}|\gamma_{ij})$, $i,j \in R$, $i \neq j$, with respect to a common dominating measure. Assume that $\log p(x_{ij}|\gamma_{ij})$ is a strictly concave functional of γ_{ij} having a maximizer in $\mathbb{R}$ (note that the maximizer is unique since the function is strictly concave). The log-likelihood is hence

$$\log p(\mathbf{x}|\boldsymbol{\gamma}) = \sum_{i\in R} \sum_{j\in R\setminus\{i\}} \log p(x_{ij}|\gamma_{ij}) \,. \tag{12}$$

4.3 Probing Techniques and Modeling

To illustrate our approach we will focus on one type of metric. In earlier work we proposed a topology identification method based on delay differences [46]. The method relies on a measure-

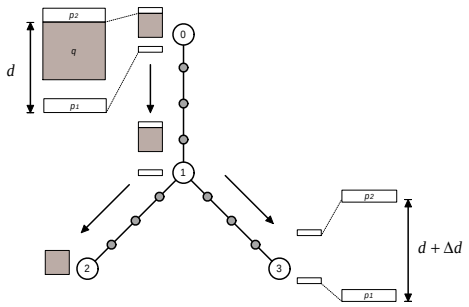


Figure 9: An example of sandwich probe measurement. The large packet is destined for node 2, the small packets for node 3. The black circles on the links represent physical queues where no branching occurs. In the absence of cross-traffic, the initial spacing between the small probes d is increased along the shared path from nodes 0 to 1 because the second small probe p_2 queues behind the large packet. The measurement $x_{2,3}$ for this receiver pair is equal to $d + \Delta d$. A larger initial spacing d reduces the chance of p_2 catching p_1 because of a bottleneck or cross-traffic on the path from node 1 to 3.

ment scheme called sandwich probing. Here we briefly describe the measurement technique, details of which can be found in [46]. Each sandwich probe consists of three packets, and gives information about the shared path among two receivers. Figure 9 shows the details of the probing scheme.

The metrics used are the mean delay differences. Since we only need to measure local delay differences (at the sender and the receiver), there is no need for clock synchronization between the source host and the receivers. In the case where there is no cross traffic the measurement Δd is directly related to the bandwidth of the shared queues. We assume the cross-traffic is stationary, and that the initial spacing of the two small packets d is large enough so that neither the large packet nor the second small packet queue behind the first small packet at any time. We send each probe far apart in time, so that we can assume that the outcomes of different measurements are independent.

The delay differences provide (noisy) measurements of a metric related to the number of shared queues in the paths to two receivers. Let x_{ij} be the sample mean of repeated delay difference measurements for pair $i, j \in R$. Under reasonable assumptions the measurements are statistically independent and have finite variance, hence, according to the Central Limit Theorem, the distribution of each empirical mean tends to a Gaussian. This motivates the

following (approximate) model:

$$x_{ij} \ \sim \ \mathcal{N}(\gamma_{ij}, \sigma_{ij}^2), \tag{13}$$

where σ_{ij}^2 is the sample variance of the measurements divided by the number of measurements, ($\boldsymbol{\sigma}^2 \equiv \{\sigma_{ij}^2\}$), x_{ij} is the sample mean of the measurements, and $\mathcal{N}(\gamma, \sigma^2)$ denotes the Gaussian density with mean γ and variance σ^2. Notice that we are not assuming that the delay differences are normally distributed, but only their empirical means. Under the above assumptions, as the number of measurements increases, the model accuracy increases. Henceforth we will refer to $\mathbf{x}$ and $\boldsymbol{\sigma}^2$ as estimated metrics and estimated variances, respectively. We also assume that the measurements for the different receiver pairs are statistically independent.

Note that, although we have considered the particular case of sandwich probing here, this modeling technique can be used with a variety of other probing schemes, provided that for each pair of receivers we have a number of independent measurements with finite variance; the application of the central limit theorem provides the above approximate model. Using the models describe above, we are in the scenario described in Section 4.2:

$$\log p(x_{ij}, \sigma_{ij} | \gamma_{ij}) = -\frac{(x_{ij} - \gamma_{ij})^2}{2\sigma_{ij}^2} + C_{ij} \ , \tag{14}$$

where C_{ij} is a normalizing constant, and the densities are taken with respect to the Lebesgue measure.

4.4 Characterization of the Maximum Likelihood Structure

The optimization problem in (10) is quite formidable. We are not aware of any method for computation of the global maximum except by a brute force examination of each tree in the forest. Consider a tree with N leafs. A very loose lower bound on the size of the forest $\mathcal{F}$ is $N!/2$. For example, if $N = 10$ then there are more than 1.8×10^6 trees in the forest. This explosion of the search space precludes the brute force approach in all but very small forests. Moreover, the computation of the profile likelihood (11) is non-trivial because it involves a

constrained optimization over $\mathcal{G}(\mathcal{T})$. Using the model (12) we have that $p(\mathbf{x}|\boldsymbol{\gamma})$ is a continuous function of $\boldsymbol{\gamma}$ and hence we can rewrite the profile likelihood as

$$\mathcal{L}(\mathbf{x}|\mathcal{T}) \equiv \max_{\boldsymbol{\gamma}\in\overline{\mathcal{G}(\mathcal{T})}} p(\mathbf{x}|\boldsymbol{\gamma}) \;, \tag{15}$$

where $\overline{\mathcal{G}(\mathcal{T})}$ denotes the closure of the set $\mathcal{G}(\mathcal{T})$. Hence the profile likelihood can be computed by solving this constrained optimization problem.

The following results establish some important properties of this problem. Theorem 1 identifies required characteristics of the maximum-likelihood tree and allows us to use Lemma 1 to compute the profile likelihood. For a proof of the theorem, see [47].

Lemma 1 *Let $\mathcal{T}$ be an arbitrary tree. The solution of*

$$\widehat{\boldsymbol{\gamma}} = \arg \max_{\boldsymbol{\gamma}\in\boldsymbol{\Gamma}(\mathcal{T})} \log p(\mathbf{x}|\boldsymbol{\gamma})$$

is unique and given by

$$\widehat{\gamma}_{ij} = \arg\max_{\gamma\in I\!R} \sum_{k,l\in R: a(k,l)=a(i,j)} f_{kl}(x_{kl}|\gamma) \;. \tag{16}$$

This lemma, whose proof is elementary, characterizes a modified version of the profile likelihood (15) (this version of the profile likelihood is "unconstrained", that is, the optimization is over $\boldsymbol{\Gamma}(\mathcal{T})$ instead of $\overline{\mathcal{G}(\mathcal{T})}$). The evaluation of this version of the profile likelihood for a given tree is very simple.

Theorem 1 *Let $\log p(\mathbf{x}|\boldsymbol{\gamma})$ be given by (12) and $\widetilde{\mathcal{T}}$ be a tree such that*

$$\max_{\boldsymbol{\gamma}'\in\boldsymbol{\Gamma}(\widetilde{\mathcal{T}})} \log p(\mathbf{x}|\boldsymbol{\gamma}') \;>\; \max_{\boldsymbol{\gamma}'\in\overline{\mathcal{G}(\widetilde{\mathcal{T}})}} \log p(\mathbf{x}|\boldsymbol{\gamma}') \;. \tag{17}$$

Then there exists another tree $(\mathcal{T}, \boldsymbol{\gamma})$, $\boldsymbol{\gamma} \in \mathcal{G}(\mathcal{T})$, *satisfying the monotonicity property, such that*

$$\log p(\mathbf{x}|\boldsymbol{\gamma}) \; > \max_{\boldsymbol{\gamma}' \in \overline{\mathcal{G}(\widetilde{\mathcal{T}})}} \log p(\mathbf{x}|\boldsymbol{\gamma}') \tag{18}$$

In particular, if $\mathcal{T}^*$ *is the solution to (10), i.e., the MLT, we have*

$$\arg \max_{\boldsymbol{\gamma}' \in \boldsymbol{\Gamma}(\mathcal{T}^*)} \log p(\mathbf{x}|\boldsymbol{\gamma}') \; = \; \arg \max_{\boldsymbol{\gamma}' \in \overline{\mathcal{G}(\mathcal{T}^*)}} \log p(\mathbf{x}|\boldsymbol{\gamma}') \; . \tag{19}$$

Remark 1: Consider a arbitrary tree $\widetilde{\mathcal{T}}$. Suppose that the maximum of $p(\mathbf{x}|\boldsymbol{\gamma})$ is attained for $\boldsymbol{\gamma} \in \boldsymbol{\Gamma}(\widetilde{\mathcal{T}}) \setminus \overline{\mathcal{G}(\widetilde{\mathcal{T}})}$, that is, expression (17) holds. The theorem says that in that case we can construct another tree $(\mathcal{T}, \boldsymbol{\gamma})$ from $\widetilde{\mathcal{T}}$ such that it yields a higher likelihood than any tree $(\widetilde{\mathcal{T}}, \boldsymbol{\gamma})$, $\boldsymbol{\gamma} \in \overline{\mathcal{G}(\widetilde{\mathcal{T}})}$. Consequently $\widetilde{\mathcal{T}}$ cannot be the maximum likelihood tree.

Remark 2: The second part of the theorem, expressed by (19), means that it is unnecessary to perform the strongly constrained optimization over $\mathcal{G}(\mathcal{T})$. For each tree, we can compute the much simpler optimization over $\boldsymbol{\Gamma}(\mathcal{T})$, using Lemma 1, and subsequently check if the resulting maximizer lies in the set $\overline{\mathcal{G}(\mathcal{T})}$.

Define the set of trees

$$\mathcal{F}' = \left\{ \mathcal{T} \in \mathcal{F} : \arg \max_{\boldsymbol{\gamma} \in \boldsymbol{\Gamma}(\mathcal{T})} \log p\left(\mathbf{x}|\boldsymbol{\gamma}\right) \in \overline{\mathcal{G}(\mathcal{T})} \right\} \; .$$

The maximum likelihood tree must belong to this set, i.e., $\mathcal{T}^* \in \mathcal{F}'$.

Remark 3: From the proof technique (see [47]) we note that we need only to consider binary trees in $\mathcal{F}'$, because for any non-binary tree we can construct a corresponding binary tree yielding the same likelihood value.

4.5 Bottom-up Agglomerative Approach

In *Remark 2* above we observed that we can greatly simplify the task of evaluating the profile likelihood for trees belonging to $\mathcal{F}'$, and we know that the MLT, ultimately what we want to determine, belongs to that set. Nevertheless a brute force examination of each tree in $\mathcal{F}'$ is still infeasible. In a scenario where one can determine the true pairwise similarities $\boldsymbol{\gamma}$, it is possible to reconstruct the dendritic tree using a simple agglomerative bottom-up procedure, following the same conceptual framework as in many hierarchical clustering methods [43, 44, 42]. The following result ensures that, if a pairwise metrics matrix $\boldsymbol{\gamma}$ satisfies the monotonicity property, then it completely determines the tree.

Proposition 1 *Let $\mathcal{T}$ be a tree topology with receiver set R. Let $\{\gamma_{ij}\}$ be the set of pairwise similarities, corresponding to a monotonic metric on $\mathcal{T}$. Then $\mathcal{T}$ is the only tree with pairwise similarities $\{\gamma_{ij}\}$ satisfying the monotonicity property. That is*

$$\boldsymbol{\gamma} \in \mathcal{G}(\mathcal{T}), \quad \textit{and} \quad \boldsymbol{\gamma} \not\subseteq \mathcal{G}(\mathcal{T}'),\ \forall\ \mathcal{T}' \neq \mathcal{T}\ .$$

Recall that, in most practical scenarios, we only have access to the measurements $\mathbf{x}$, conveying information about $\boldsymbol{\gamma}$ (and hence about $\mathcal{T}$). In this case we can still develop a bottom-up agglomerative clustering algorithm to estimate the true topology.

We restrict ourselves to binary trees. Note that for any non-binary topology tree there exists an equivalent binary tree (in the sense that there are extra, unnecessary branches in the binary tree). The binary restriction leads to a particularly simple algorithm as follows. Consider the estimates of the pairwise similarities for each pair of leaf nodes, given by

$$\widehat{\gamma}_{ij} = \arg\max_{\gamma \in \mathbb{R}} \left(\log p(x_{ij}|\gamma) + \log p(x_{ji}|\gamma)\right),\ i,j \in R,\ i \neq j\ .$$

One expects that the above estimated pairwise similarities are reasonably close to the true similarities $\boldsymbol{\gamma}$, with the differences being due to measurement noise and limitations of the measurement procedure. Consider the pair of leaf nodes such that $\widehat{\gamma}_{ij}$ is greatest, that is

$$\widehat{\gamma}_{ij} \geq \widehat{\gamma}_{kl}, \ \forall k, l \in R' \ ,$$

where $R' = R \setminus \{i, j\}$. We infer that i and j are the most similar leaf nodes, and so they must have a common parent in the dendrogram. Denote their parent node by k. In other words, we infer that the receivers descending from k are $\{i, j\}$.

Assuming that our decision is correct then the tree structure imposes that $a(i, l) = a(j, l)$ for all $l \notin \{i, j\}$. Hence we can update our pairwise similarity estimates for pairs involving i and j, using Lemma 1. Furthermore, since $\widehat{\gamma}_{il} = \widehat{\gamma}_{jl}$ for any $l \notin \{i, j\}$, we can just add node k as a new leaf node, and remove i and j from the leafs set. Define the new leaf set $R' = R \bigcup \{k\} \setminus \{i, j\}$. We just need to define pairwise similarity estimates for pairs involving the new node k:

$$\widehat{\gamma}_{kl} = \widehat{\gamma}_{lk} \equiv \arg\max_{\gamma \in \mathbb{R}} \sum_{r \in S_k} \log p(x_{rl}|\gamma) + \log p(x_{lr}|\gamma) \ , \quad \text{where } l \in R' \setminus \{k\} \ . \tag{20}$$

This procedure is iterated until there is only one element left in R'. This is called the Likelihood based Binary Tree (LBT) algorithm. Notice that the number of elements of R' is decreased by one at each step of the algorithm, guaranteeing the algorithm to stop. The algorithm structure ensures that the obtained tree and similarity estimates $\widehat{\boldsymbol{\gamma}}$ satisfy the monotonicity property.

As pointed out before, one expects the estimated pairwise similarity values $\widehat{\boldsymbol{\gamma}}$ to be close to the true similarities $\boldsymbol{\gamma}$. We now explore this intuitive notion in more detail. Define

$$\gamma_{ij}(\mathbf{X}) = \arg\max_{\gamma \in \mathbb{R}} \log p(x_{ij}|\gamma) + \log p(x_{ji}|\gamma) \ .$$

Given the measurements $\mathbf{x}$, $\gamma_{ij}(\mathbf{X})$ is the maximum likelihood estimate of the pairwise similarities γ_{ij}. In the most common scenarios, the measurements $\mathbf{x}$ for a particular pair of input

nodes correspond to several individual measurements for that pair (or a composition of such measurements). It is desirable that, the accuracy of the estimates increases as the number of pairwise measurements increases.

Let $n \in \mathbb{N}$ represent the number of measurements for each input pair and assume that

$$\gamma_{ij}(\mathbf{X}) \xrightarrow{P} \gamma_{ij} \quad \text{as} \quad n \to \infty \,, \tag{21}$$

where $\xrightarrow{P}$ denotes convergence in probability. It can be shown that, for a binary tree $\mathcal{T}$, the LBT algorithm is consistent, that is, if $\widehat{\mathcal{T}}(\mathbf{X})$ is the tree obtained by the LBT algorithm then

$$\lim_{n\to\infty} \Pr(\mathcal{T} = \widehat{\mathcal{T}}(\mathbf{X})) = 1 \,.$$

Hence, the LBT algorithm perfectly reconstructs the original binary tree, provided that one can estimate the similarity values with enough accuracy. We can also show that, as n grows, this is the only tree in $\mathcal{F}'$, that is

$$\lim_{n\to\infty} \Pr(\mathcal{F}' = \{\mathcal{T}\}) = 1 \,.$$

(see [47]) Based on these observations, we conclude the following, important result:

Proposition 2 *The MLT is a consistent estimator of the true topology tree, for binary trees. That is, if $\mathcal{T}^*$ denotes the MLT then*

$$\lim_{n\to\infty} \Pr(\mathcal{T} = \mathcal{T}^*) = 1 \,.$$

4.5.1 Normally Distributed Measurements:

In the case of the particular measurement model in Section 4.3 the above algorithm is greatly simplified. Note that the model (14) has some desirable properties, namely, it is closed under

summation:

$$f(x_{ij},\sigma_{ij}^2|\gamma)+f_{kl}(x_{kl},\sigma_{kl}^2|\gamma)=-\frac{\left(\left(\frac{x_{ij}}{\sigma_{ij}^2}+\frac{x_{kl}}{\sigma_{kl}^2}\right)/\left(\frac{1}{\sigma_{ij}^2}+\frac{1}{\sigma_{kl}^2}\right)-\gamma\right)^2}{2\left(\frac{1}{\sigma_{ij}^2}+\frac{1}{\sigma_{kl}^2}\right)}+C$$

This makes the computations arising from Lemma 1 very simple, namely the aggregation step (equation 20) in the LBT algorithm, since throughout the algorithm we only need to propagate the matrix of empirical means and variances of the aggregated receiver set R'.

4.6 Markov Chain Monte Carlo Method

Although the LBT algorithm is a consistent estimator of the true dendrogram, it is essentially greedy, based on local decisions over the estimated pairwise similarities. Unlike the LBT, the MLT estimator takes a global approach, seeking for the best (in a maximum likelihood sense) tree. The price to pay is that we now must search over the entire forest $\mathcal{F}$ (or at least over $\mathcal{F}'$). In this section we propose a random search technique to efficiently search the forest of trees.

Consider the profile likelihood $\mathcal{L}(\mathbf{x}|\mathcal{T})$, as defined in (15). Note that the maximum likelihood tree is the tree that maximizes $\mathcal{L}(\mathbf{x}|\mathcal{T})$. For a fixed measurement $\mathbf{x}$ we can regard the profile likelihood $\mathcal{L}(\mathbf{x}|\mathcal{T})$ as a discrete distribution over $\mathcal{F}$ (up to a normalizing factor). Then one way of searching the set $\mathcal{F}$ is to sample it according to this distribution. The more likely trees are going to be sampled more often than the less likely trees, making the search more efficient.

The evaluation of the profile likelihood (15) is still complicated, since it involves a constrained optimization over $\mathcal{G}(\mathcal{T})$. As we observed in Section 4.4 the MLT belongs to the set of feasible trees $\mathcal{F}'$. For trees in $\mathcal{F}'$ one can compute the profile likelihood very easily using Lemma 1. Also, given a tree, one can easily verify if that tree belongs to $\mathcal{F}'$ or not: For a given tree compute the less constrained optimization (over $\mathbf{\Gamma}(\mathcal{T})$), using Lemma 1, and check if the

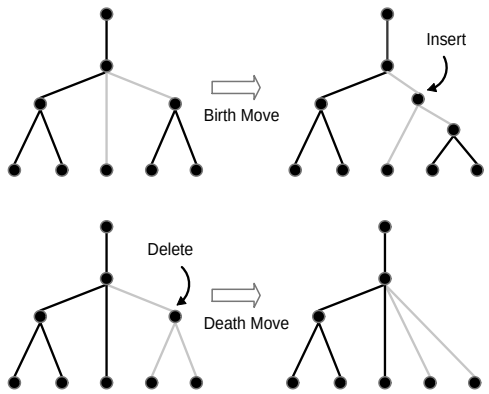


Figure 10: The birth-step and death-step moves illustrated: The birth-step selects a node with more than two children, chooses two of these children, and inserts an extra node as the new parent of these children. The death step chooses a node with two children, and deletes that node

resulting maximizer lies in the set $\overline{\mathcal{G}(\mathcal{T})}$. If so the tree lies in $\mathcal{F}'$. Define

$$\mathcal{L}'(\mathbf{x}|\mathcal{T}) = \begin{cases} \mathcal{L}(\mathbf{x}|\mathcal{T}) & \text{if } \mathcal{T} \in \mathcal{F}' \\ 0 & \text{otherwise} \end{cases} . \tag{22}$$

The above expression can be evaluated much faster than (15). Again, for a fixed measurement $\mathbf{x}$ we can regard $\mathcal{L}'(\mathbf{x}|\mathcal{T})$ as a discrete distribution over $\mathcal{F}'$ (up to a normalizing factor). We can search the set $\mathcal{F}'$ by sampling it according to this distribution. One way of performing this task is to use the Metropolis-Hastings algorithm. For this we need to construct a irreducible Markov chain with state space $\mathcal{F}$ (note that in this case the state space is finite), so each state corresponds to a tree. We allow only certain transitions (equivalently, certain transitions have probability 0). For a given state (a tree) $s_i \in \mathcal{F}$ we can move to another state (tree) using "birth moves" and "death moves" as illustrated in Figure 10.

For a given state $s_i \in \mathcal{F}$ there are n_{s_i} allowed transitions (both deaths and births). This number, denoted n_{s_i}, can be easily determined by simple enumeration of the possibilities. We build the chain such that we choose the next state with probability $1/n_{s_i}$ if it is within one move from s_i. All the other states have probability zero. Denote by $\{s_i : i \in \mathbb{N}\}$ the Markov chain constructed. It can be easily shown that the chain $\{s_i\}$ is irreducible.

Using a generalization of the Metropolis algorithm due to Hastings [48] we construct another Markov Chain in order to obtain a chain whose unique limit distribution (and also unique stationary distribution) is precisely $\mathcal{L}'(\mathbf{x}|\mathcal{T})$. Thus if we sample from this chain for a sufficiently large period of time, the observed states will be samples of the distribution $\mathcal{L}'(\mathbf{x}|\mathcal{T})$, regardless of the initial state.

The new chain can be constructed in a two-stage fashion: For a given state $\mathcal{T} \in \mathcal{F}'$ we randomly choose the possible next state $\mathcal{T}'$ according to the previously constructed chain. We accept this transition with probability

$$\min\left\{\frac{\mathcal{L}'(\mathbf{x}|\mathcal{T}')}{\mathcal{L}'(\mathbf{x}|\mathcal{T})}, 1\right\} \text{ if } \mathcal{L}'(\mathbf{x}|\mathcal{T}) > 0 \ ,$$

and zero otherwise. It can be easily shown that the resulting chain (over $\mathcal{F}'$) is still irreducible, and thus it has unique limit distribution (22). To get our (approximate) solution of (10) we simulate the above chain and recall the state with largest likelihood visited, the longer the chain is simulated the higher is the chance of visiting the MLT at least one time.

Although theoretically the starting point (initial state) of the chain is not important, provided that the chain is simulated for long enough, starting at a reasonable point can help the chain to visit the MLT more rapidly. Starting the chain simulation from the tree obtained using the LBT algorithm is a reasonable approach, since this is a consistent estimator, and so one expects the resulting tree to be "close" (in terms of the number of moves) to the actual MLT tree.

One drawback to the likelihood criterion is that it places no penalty on the number of links in the tree. As a consequence, trees with more links can have higher likelihood values (since the extra degrees of freedom they possess can allow them to fit the data more closely). In general, the true tree $\mathcal{T}$ will have a smaller likelihood value than another tree $\mathcal{T}'$ that is identical to $\mathcal{T}$ except that one or more of the nodes in $\mathcal{T}$ are replaced with extra branching nodes that allow $\mathcal{T}'$ to fit the data more closely. This is an instance of the classic "overfitting" problem

associated with model estimation; the more degrees of freedom in a model, the more closely the model can fit the data. Of course, we are not interested in simply fitting the data, but rather in determining a reasonable estimate of the underlying topology.

The overfitting problem can be remedied by applying regularization, replacing the simple likelihood criterion with a *penalized* likelihood criterion.

$$\widehat{\mathcal{T}}_\lambda \ = \arg\max_{\mathcal{T}\in\mathcal{F}} \ \mathcal{L}(\mathbf{x}|\mathcal{T}) - \ \lambda\, n(\mathcal{T}) \ , \tag{23}$$

where $n(\mathcal{T})$ is the number of links in the tree $\mathcal{T}$ and $\lambda \geq 0$ is a parameter, chosen by the user, to balance the trade-off between fitting to the data and controlling the number of links in the tree. Under this approach there are still some simplifications we can consider. A modified version of Theorem 1 still applies

Proposition 3 *Let* $\log p(\mathbf{x}|\boldsymbol{\gamma})$ *be given by (12). If* $\widehat{\mathcal{T}}$ *is the solution to (23), i.e., the MPLT, we have*

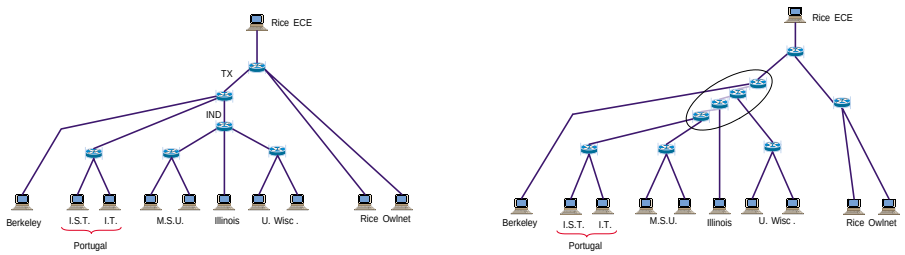
$$\arg \max_{\boldsymbol{\gamma}'\in\boldsymbol{\Gamma}(\widehat{\mathcal{T}})} \log p(\mathbf{x}|\boldsymbol{\gamma}') - \ \lambda\, n(\widehat{\mathcal{T}}) \ = \ \arg \max_{\boldsymbol{\gamma}'\in\overline{\mathcal{G}(\widehat{\mathcal{T}})}} \log p(\mathbf{x}|\boldsymbol{\gamma}') - \ \lambda\, n(\widehat{\mathcal{T}}) \ .$$

Remark: As before, this result shows that it is unnecessary to perform the strongly constrained optimization over $\mathcal{G}(\mathcal{T})$. For each tree, we can compute the less constrained optimization (over $\boldsymbol{\Gamma}(\mathcal{T})$), using Lemma 1, and check if the resulting maximizer lies in the set $\overline{\mathcal{G}(\mathcal{T})}$.

We can construct, in a similar fashion as before, a Markov Chain, that samples the forest $\mathcal{F}$, according to a distribution proportional to

$$\mathcal{L}'(\mathbf{x}|\mathcal{T}) \exp(-\ \lambda\, n(\mathcal{T})).$$

This method finds a reasonably simple tree that accurately fits the measured data. Setting $\lambda = 0$ will produce the MLT. The larger the value of λ the more the penalized likelihood criterion



(a) (b)

Figure 11: (a) The topology of the network used for Internet experiments, obtained using *traceroute*. (b) Estimated topology using the LBT algorithm. The signaled links have link-parameter values $\gamma_k - \gamma_{f(k)}$ one order of magnitude smaller than all the other links. Those links can be collapsed, that is, the three devices inside the circle can be identified as one unique device.

favors simpler trees with fewer links. The choice of the penalty λ must also be addressed. Minimum description length principles [49] motivate a penalty that is dependent on the size of the network (in terms of the number of receivers). However, other model selection techniques lead to the selection of different penalties [50].

5 Experimental Results

We have implemented a software tool called `nettomo` that performs sandwich probing measurements and estimates the topology of a tree-structured network. We conducted Internet experiments using several hosts in the United States and abroad. The topology inferred from `traceroute` is depicted in Figure 11(a). The source for the experiments was located at Rice University. There are ten receiver clients, two located on different networks at Rice, two at separate hosts in Portugal, and six located at four other US universities.

The experiment was conducted for a period of eight minutes, during which a sandwich probe was sent to a randomly chosen receiver-pair once every 50 ms. Without any loss, the maximum number of probes available is 8600. This corresponds to less than 200 probes per pair, hence the traffic overhead on any link is very low.

We applied the LBT algorithm to the measurements collected and the result is depicted in Figure 11(b). Since the procedure is suited only for binary trees, it adds some extra links with

small link-level metric value (i.e. $\gamma_k - \gamma_{f(k)} \approx 0$), indicating possibly high bandwidth links). Typically those links have link-level parameters one order of magnitude smaller than all remaining ones, hence, if we collapse those links, identifying their end-nodes with each other, we get a non-binary tree that provides a more truthful estimate of the network (avoiding overfitting the data) . Using this pruning procedure we get a very good estimate of the true network topology, although it fails to detect the backbone connection between Texas and Indianapolis. We expect that the latter connection is very high speed and the queuing effects on the constituent links are too minor to influence measurements. The estimated topology also places an extra element shared between the Rice computers. Although that element is not a router, hence it is not shown in the topology estimate using `traceroute`, it corresponds to a real physical device. To the best of our knowledge the detected element is a bandwidth limitation device.

6 Conclusion and Future Directions

This article has provided an overview of the area of large scale inference and tomography in communications networks. As is evident from the limited scale of the simulations and experiments discussed in this article, the field is only just emerging. Deploying measurement/probing schemes and inference algorithms in larger networks is the next key step. Statistics will continue to play an important role in this area and here we attempt to stimulate the reader with an outline of some of the many open issues. These issues can be divided into extensions of the theory and potential networking applications areas.

The spatio-temporally stationary and independent traffic and network transport models have limitations, especially in tomographic applications involving heavily loaded networks. Since one of the principal applications of network tomography is to detect heavily loaded links and subnets, relaxation of these assumptions continues to be of great interest. Some recent work on relaxing spatial dependence and temporal independence has appeared in unicast [14] and multicast [7] settings. However, we are far from the point of being able to implement

flexible yet tractable models which simultaneously account for long time traffic dependence, latency, dynamic random routing, and spatial dependence. As wireless links and ad hoc networks become more prevalent spatial dependence and routing dynamics will become dominant.

Recently, there have been some preliminary attempts to deal with the time-varying, nonstationary nature of network behavior. In addition to the estimation of time-varying OD traffic matrices discussed in Section 3.2, others have adopted a dynamical systems approach to handle nonstationary link-level tomography problems [51]. Sequential Monte Carlo inference techniques are employed in [51] to track time-varying link delay distributions in nonstationary networks. One common source of temporal variability in link-level performance is the nonstationary characteristics of cross-traffic.

There is also an accelerating trend toward network security that will create a highly uncooperative environment for active probing — firewalls designed to protect information may not honor requests for routing information, special packet handling (multicast, TTL, etc.), and other network transport protocols required by many current probing techniques. This has prompted investigations into more passive traffic monitoring techniques, for example based on sampling TCP traffic streams [52]. Furthermore, the ultimate goal of carrying out network tomography on a massive scale poses a significant computational challenge. Decentralized processing and data fusion will probably play an important role in reducing both the computational burden and the high communications overhead of centralized data collection from edge-nodes.

The majority of work reported to date has focused on reconstruction of network parameters which may only be indirectly related to the decision-making objectives of the end-user regarding the existence of anomalous network conditions. An example of this is bottleneck detection which has been considered in [53, 15] as an application of reconstructed delay or loss estimation. Other important decision-oriented applications may be detection of coordinated attacks on network resources, network fault detection, and verification of services.

Finally the impact of network monitoring, which is the subject of this article, on network control and provisioning could become the application area of most practical importance. Ad-

mission control, flow control, service level verification, service discovery, and efficient routing could all benefit from up-to-date and reliable information about link and router level performances. The big question is: can statistical methods be developed which ensure accurate, robust and tractable monitoring for the development and administration of the Internet and future networks?

Acknowledgments

This work was supported by the National Science Foundation, grant nos. MIP–9701692, ANI-0099148, FD01-12731, and ANI-9734025, the Office of Naval Research, grant no. N00014-00-1-0390, the Army Research Office, grant nos. DAAD19-99-1-0290, DAAD19-01-1-0643, and DAAH04-96-1-0337, and the Department of Energy, grant no. DE-FC02-01ER25462. The authors would also like to acknowledge the invaluable contributions of J. Cao, D. Davis, M. Gadhiok, R. King, E. Rombokas, Y. Tsang, and S. Vander Wiel to the work described in this article.

References

[1] M.J. Coates, A.O. Hero, R. Nowak, and B. Yu. Internet tomography. *IEEE Signal Processing Magazine*, 19(3):47–65, May 2002.

[2] F. P. Kelly, S. Zachary, and I. Ziedins. *Stochastic networks: theory and applications*. Royal Statistical Society Lecture Note Series. Oxford Science Publications, Oxford, 1996.

[3] X. Chao, M. Miyazawa, and M. Pinedo. *Queueing networks: customers, signals and product form solutions*. Systems and Optimization. Wiley, New York, NY, 1999.

[4] CAIDA: Cooperative Association for Internet Data Analysis. http://www.caida.org/Tools/.

[5] Y. Vardi. Network tomography: estimating source-destination traffic intensities from link data. *J. Amer. Stat. Assoc.*, pages 365–377, 1996.

[6] Multicast-based inference of network-internal characteristics (MINC). http://gaia.cs.umass.edu/minc.

[7] R. Cáceres, N. Duffield, J. Horowitz, and D. Towsley. Multicast-based inference of network-internal loss characteristics. *IEEE Trans. Info. Theory*, 45(7):2462–2480, November 1999.

[8] M. Coates and R. Nowak. Network loss inference using unicast end-to-end measurement. In *ITC Seminar on IP Traffic, Measurement and Modelling*, Monterey, CA, Sep. 2000.

[9] M. Coates and R. Nowak. Network delay distribution inference from end-to-end unicast measurement. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc.*, May 2001.

[10] N.G. Duffield, F. Lo Presti, V. Paxson, and D. Towsley. Inferring link loss using striped unicast probes. In *Proceedings of IEEE INFOCOM 2001*, Anchorage, Alaska, April 2001.

[11] K. Harfoush, A. Bestavros, and J. Byers. Robust identification of shared losses using end-to-end unicast probes. In *Proc. IEEE Int. Conf. Network Protocols*, Osaka, Japan, Nov. 2000. *Errata* available as Boston University CS Technical Report 2001-001.

[12] F. Lo Presti, N.G. Duffield, J. Horowitz, and D. Towsley. Multicast-based inference of network-internal delay distributions. Technical Report CMPSCI 99-55, University of Massachusetts, 1999.

[13] S. Ratnasamy and S. McCanne. Inference of multicast routing trees and bottleneck bandwidths using end-to-end measurements. In *Proceedings of IEEE INFOCOM 1999*, New York, NY, March 1999.

[14] M.F. Shih and A.O. Hero. Unicast inference of network link delay distributions from edge measurements. Technical report, Comm. and Sig. Proc. Lab. (CSPL), Dept. EECS, University of Michigan, Ann Arbor, May 2001.

[15] A.-G. Ziotopolous, A.O. Hero, and K. Wasserman. Estimation of network link loss rates via chaining in multicast trees. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc.*, May 2001.

[16] C. Tebaldi and M. West. Bayesian inference on network traffic using link count data (with discussion). *J. Amer. Stat. Assoc.*, pages 557–576, June 1998.

[17] J. Cao, D. Davis, S. Vander Wiel, and B. Yu. Time-varying network tomography: router link data. *J. Amer. Statist. Assoc*, 95:1063–1075, 2000.

[18] J. Cao, S. Vander Wiel, B. Yu, and Z. Zhu. A scalable method for estimating network traffic matrices. Technical report, Bell Labs, 2000.

[19] G. Liang and B. Yu. Maximum pseudo likelihood estimation in network tomography. In *Proc. IEEE Infocom 2003*, San Francisco, California, April 2003.

[20] R.J. Vanderbei and J. Iannone. An EM approach to OD matrix estimation. Technical Report SOR 94-04, Princeton University, 1994.

[21] Finbarr O'Sullivan. A statistical perspective on ill-posed inverse problems. *Statistical Science.*, 1(4):502–527, 1986.

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

[23] J. Cao, D. Davis, S. Vander Wiel, and B. Yu. Time-varying network tomography: router link data. *Journal of American Statistics Association*, 95(452):1063–1075, 2000.

[24] G. Liang and B. Yu. Maximum pseudo likelihood estimation in network tomography. To appear, IEEE Trans. Signal Processing, August 2003.

[25] J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society, Series B*, 36(2):192–236, 1974.

[26] J. Besag. Statistical analysis of non-lattice data. *Statistican*, 24(3):179–195, 1975.

[27] D. R. Cox. Partial likelihood. *Biometrika*, 62:269–276, 1975.

[28] H. White. *Estimation, Inference and Specification Analysis*. New York: Cambridge University Press, 1994.

[29] D.W. Scott. *Multivariate Density Estimation: Theory, Practice and Visulization*. Wiley, Ney York, 1992.

[30] David Blackwell. Approximate normality of large products. Technical report, University of California at Berkeley, 1973.

[31] R. J. Vanderbei and J. Iannone. An EM approach to OD matrix estimation. Technical report, Princeton University, 1994. SOR 94-04.

[32] I. Csiszár. I-divergence geometry of probability distributions and minimization problems. *The Annals of Probability*, 3(1):146–158, 1975.

[33] Ross Ihaka and Robert Gentleman. R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5(3):299–314, 1996.

[34] UCB/LBNL/VINT network simulator ns (version 2). www.isi.edu/nsnam/ns/.

[35] P. Willet. Recent trends in hierarchical document clustering: a critical review. *Information Processing and Management*, 24:577–597, 1988.

[36] E. M. Voorhees. Implementing agglomerative hierarchical clustering algorithms for use in document retrieval. *Information Processing and Management*, 22:465–476, 1986.

[37] A. El-Hamdouchi and P. Willet. Hierarchic document clustering using ward's method. In *proceedings of the Ninth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1986.

[38] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. BIRCH: an efficient data clustering method for very large databases. In *ACM SIGMOD International Conference on Management of Data*, pages 103–114, Montreal, Canada, June 1996.

[39] N.G. Duffield, J. Horowitz, F. Lo Presti, and D. Towsley. Multicast topology inference from end-to-end measurements. In *ITC Seminar on IP Traffic, Measurement and Modelling*, Monterey, CA, Sep. 2000.

[40] N.G. Duffield, J. Horowitz, F. Lo Presti, and D. Towsley. Multicast topology inference from measured end-to-end loss. *IEEE Trans. Info. Theory*, 48(1):26–45, January 2002.

[41] A. Bestavros, J. Byers, and K. Harfoush. Inference and labeling of metric-induced network topologies. Technical Report BUCS-2001-010, Computer Science Department, Boston University, June 2001.

[42] D. Fasulo. An analysis of recent work on clustering algorithms, 1999.

[43] J. H. Ward. Hierarchical grouping to optimize an objective function. *J. American Stat. Assoc.*, 58:236–245, 1963.

[44] F. Murtagh. A survey of recent advances in hierarchical clustering algorithms. *The Computer Journal*, 26(4), 1983.

[45] J. O. Berger, Liseo B., and Wolpert R. L. Integrated likelihood methods for eliminating nuisance parameters. *Statistical Science*, 14:1–28, 1999.

[46] M.J. Coates, R. Castro, M. Gadhiok, R. King, Y. Tsang, and R.D. Nowak. Maximum likelihood network topology identification from edge-based unicast measurements. In *Proc. ACM Sigmetrics*, Marina Del Rey, CA, Jun. 2002.

[47] R. Castro, M.J. Coates, M. Gadhiok, R. King, R. Nowak, E. Rombokas, and Y. Tsang. Maximum likelihood network topology identification from edge-based unicast measurements. Technical Report TREE0107, Department of Electrical and Computer Engineering, Rice University, Oct. 2001.

[48] W.K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109, 1970.

[49] J. Rissanen. *Stochastic Complexity in Statistical Inquiry*. World Scientific, Singapore, 1989.

[50] C.P. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer Verlag Series in Statistics, 1998.

[51] M.J. Coates and R. Nowak. Sequential Monte Carlo inference of internal delays in nonstationary communication networks. *IEEE Trans. Signal Processing, Special Issue on Monte Carlo Methods for Statistical Signal Processing*, pages 366–376, Mar. 2002.

[52] Y. Tsang, M. Coates, and R. Nowak. Passive network tomography using EM algorithms. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc.*, May 2001.

[53] M.F. Shih and A.O. Hero. Unicast inference of network link delay distributions from edge measurements. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc.*, May 2001.