NETWORK TOMOGRAPHY FOR INTERNAL DELAY ESTIMATION

Mark J. Coates and Robert D. Nowak

Department of Electrical and Computer Engineering, Rice University
6100 South Main Street, Houston, TX 77005–1892
Email: $\{$*mcoates, nowak*$\}$*@ece.rice.edu, Web: www.dsp.rice.edu*

ABSTRACT

On-line, spatially localized information about internal network performance can greatly assist dynamic routing algorithms and traffic transmission protocols. However, it is impractical to measure network traffic at all points in the network. A promising alternative is to measure only at the edge of the network and infer internal behavior from these measurements. In this paper we concentrate on the estimation and localization of internal delays based on end-to-end delay measurements from sources to receivers. We develop an EM algorithm for computing MLEs of the internal delay distributions in cases where the network dynamics are stationary over the observation period. For time-varying cases, we propose a sequential Monte Carlo procedure capable of tracking non-stationary delay characteristics. Simulations are included to demonstrate the promise of these techniques.

1. INTRODUCTION

Optimizing communication network routing and service strategies requires knowledge of the congestion (delay) at different points in the network. However, it is impractical to directly measure packet delays at each and every router. Measuring end-to-end (from source to receivers) delays is relatively easy and inexpensive in comparison. Consequently, it is natural to consider the following inverse problem: from end-to-end measurements can we resolve the delay experienced at each router? This is somewhat analogous to the medical tomography problem, and hence the name *network tomography* [1, 2, 3, 4].

The basic idea is quite straightforward. Consider a network consisting of a single source, sending packets to several receivers. Standard network routing protocols produce a tree-structured topology for the network in this case (*c.f.* Fig. 1), with the source at the root and the receivers at the leaves. Suppose two closely time-spaced (back-to-back) packets are sent from the source to two different receivers. The paths to these receivers traverse a common set of links (connections between routers), but at some point the two paths diverge (as the tree branches). The two packets should experience approximately the same delay on each shared link in their path. This facilitates the resolution of the delays on individual links (at least in a probabilistic sense, to be made clear shortly).

We collect measurements of the end-to-end delays from source to receivers, and we index the packet pair measurements by $k = 1, \dots, K$. For the k-th packet pair measurement, let $y_1(k)$ and $y_2(k)$ denote the two end-to-end delays measured. The ordering 1 and 2 is completely arbitrary. The delays are quantized such that the (quantized) delay on each link falls in the range $0, 1, \dots, L$ time units. Associated with each individual link/router in the network is a probability mass function (pmf) for the queuing delay. Let $p_i = \{p_{i,0} \dots, p_{i,L}\}$ denote probabilities of a delay on link i.

The goal of the network tomography problem considered in this paper is to estimate these probabilities, based on the end-to-end packet pair measurements. If the network is approximately stationary over the measurement period, then a natural approach is to use the maximum likelihood estimator, and in Section 2 we develop an EM algorithm for this purpose. More generally, the dynamics of the network may be changing over time, and the delay distributions themselves are no longer static. In this case, we must model the dynamics and track the network behavior. In Section 3 we propose a stochastic model of the network dynamics and develop a sequential Monte Carlo algorithm for tracking the time-varying delay distributions. Our approaches differ considerably from a previously proposed method for inferring internal delays [2] in three key respects: 1. The method in [2] employs a multicast probing technique, which is not supported by many networks. In contrast, our methods are based on unicast measurements, which can be made on any network[1]. 2. Our sequential methods are specifically designed for tracking time-varying behavior, whereas the method in [2] is only appropriate for stationary cases. 3. Both our approaches are based on a likelihood function analysis, in contrast to the sample-average approach employed in [2].

Before moving on, let us comment briefly on the assumption that back-to-back packets are delayed by roughly the same amount on each shared link in their path. If the delays are identical on shared links, then the difference between the two delay measurements can be attributed solely to the delays experienced on unshared links in the two paths. This is the key to resolving the delays on a link by link basis. However, in practice the two packets may experience slightly different delays on shared links due to the fact that one packet precedes the other in the queues and additional packets may intervene between the two. In effect, the discrepancies between the delays on shared links adds a zero mean error to the difference between the two end-to-end measurements. This "noise" produces a smoothing (or blurring) in the inferred delay pmfs. Nonetheless, because the errors are zero mean, we can still use the estimated delay pmfs to obtain reasonable estimates of the average delay on each link. Thus, our methodology is still very useful, even when the delays on shared links are not identical.

The paper is organized as follows. In Sections 2 and 3 we develop an EM algorithm for maximum likelihood estimation in

This work was supported by the National Science Foundation, grant no. MIP–9701692, the Army Research Office, grant no. DAAD19-99-1-0349, the Office of Naval Research, grant no. N00014-00-1-0390.

[1] As pointed out in [4], it should be possible to extend the method in [2] to the unicast case.

stationary scenarios and a sequential Monte Carlo procedure for estimation in both stationary and nonstationary cases. In Section 4, we evaluate the performance of these methods with simulated network experiments. We make closing remarks in Section 5.

2. EM ALGORITHM FOR STATIONARY NETWORKS

Given the packet pair measurements $\mathbf{y} \equiv \{y_1(k), y_2(k)\}$, we are interested in maximum likelihood estimates (MLEs) of $\mathbf{p} \equiv \{p_i\}$. The likelihood of each delay measurement is parameterized by a convolution of the pmfs in the path from the source to receiver. The coupling of the pmfs of each link results in a likelihood function that cannot be maximized analytically. The joint likelihood $l(\mathbf{y}|\mathbf{p})$ of all measurements is equal to a product of the individual likelihoods.

The maximization of the joint likelihood function requires numerical optimization, and the EM algorithm is an attractive strategy for this purpose. The first step in developing an EM algorithm is to propose a suitable *complete data* quantity that simplifies the likelihood function. Let $z_i(k)$ denote the delay on link i for the packets in the k-th pair. Let $\mathbf{z}_i = \{z_i(k)\}$ and $\mathbf{z} = \{\mathbf{z}_i\}$. The link delays $\mathbf{z}$ are not observed, and hence $\mathbf{z}$ is called the *unobserved data*. Define the *complete data* $\mathbf{x} \equiv \{\mathbf{y}, \mathbf{z}\}$. Note that the complete data likelihood may be factorized as follows:

$$l(\mathbf{x}|\mathbf{p}) = f(\mathbf{y}|\mathbf{z})g(\mathbf{z}|\mathbf{p}),$$

where f is the conditional pmf of $\mathbf{y}$ given $\mathbf{z}$ (which is a point mass function since $\mathbf{z}$ determines $\mathbf{y}$), and g is the likelihood of $\mathbf{z}$. The factorization shows that $l(\mathbf{x}|\mathbf{p}) \propto g(\mathbf{z}|\mathbf{p})$, since $f(\mathbf{y}|\mathbf{z})$ does not depend on the parameters $\mathbf{p}$. If we were able to measure the unobserved data, then the MLEs we seek would be trivially obtained from the complete data likelihood. Thus, the complete data likelihood is far simpler to work with than the original likelihood.

The complete data likelihood is $g(\mathbf{z}|\mathbf{p}) = \prod_{i,j} p_{i,j}^{m_{i,j}}$, where $m_{i,j} \equiv \sum_{k=1}^{N} \mathbf{1}_{z_i(k)=j}$ is the number of packets (out of all the packet pair measurements) that experience a delay of j on link i. Therefore, we have

$$l(\mathbf{x}|\mathbf{p}) \propto \prod_{i,j} p_{i,j}^{m_{i,j}}.$$

If the $m_{i,j}$ were available, then the MLE of $p_{i,j}$ is simply

$$\widehat{p}_{i,j} = \frac{m_{i,j}}{\sum_{l=1}^{L} m_{i,l}}. \tag{1}$$

The EM algorithm is an iterative method for finding the MLE of $\mathbf{p}$ that uses the complete data likelihood function. Specifically, the EM algorithm alternates between computing the conditional expectation of complete data log likelihood given the observations $\mathbf{y}$ (the E-Step) and maximizing this quantity over $\mathbf{p}$ (the M-Step). Notice that the complete data log likelihood is linear in $\mathbf{m}$:

$$\log l(\mathbf{x}|\mathbf{p}) \propto \sum_{i,j} m_{i,j} \log p_{i,j}.$$

Thus, in the E-Step we need only compute the expectation of $\mathbf{m} = \{m_{i,j}\}$.

E-Step: Let $\mathbf{p}^{(l)}$ denote the value of $\mathbf{p}$ after the l-th iteration. Then

$$\begin{aligned} \mathbf{E}_{\mathbf{p}^{(l)}}[m_{i,j}|\mathbf{y}] &= \mathbf{E}_{\mathbf{p}^{(l)}}\left[\sum_{k=1}^{N} \mathbf{1}_{\{z_i(k)=j\}}|\mathbf{y}\right], \\ &= \sum_{k=1}^{N} \mathbf{E}_{\mathbf{p}^{(l)}}\left[\mathbf{1}_{\{z_i(k)=j\}}|y_1(k), y_2(k)\right], \\ &= \sum_{k=1}^{N} p^{(l)}\left(z_i(k) = j|y_1(k), y_2(k)\right). \end{aligned}$$

We see that the conditional expectation of $\mathbf{m}$ can be computed by determining the conditional probabilities above for each packet pair measurement. This can be accomplished by a simple upward-downward probability propagation algorithm [5].

M-Step: Replace $m_{i,j}$ by

$$\widehat{m}_{i,j} = \mathbf{E}_{\mathbf{p}^{(l)}}[m_{i,j}|\mathbf{y}]$$

in (1) above to obtain the update $\mathbf{p}^{(l+1)}$.

The overall complexity of the EM algorithm is $O(KML^2)$, where M is the average number of links per path, K is the number of measurements, and L is the number of possible delay units per link.

3. SEQUENTIAL MONTE CARLO TRACKING OF TIME-VARIATION

We now consider the problem of estimating time-varying delay distributions. We first formulate a model describing the evolution of the network delay dynamics and then define a delay distribution in the time-varying context. Finally, we describe a sequential Monte Carlo procedure for dynamic estimation.

The queuing delay experienced by a measurement packet at each node in the network is due to other packets in the queue. We consider a network in which each node has a queue buffer size L with Markovian services at rate μ. The extension to inhomogeneous networks (differing service rates and queue sizes) is straightforward. We assume that we make measurements (send packet-pairs) at a rate of $C_1\mu L$ where $C_1 > 1$ is a constant. This ensures that there is sufficient time for the queues to relax between measurements, resulting in approximately statistically independent measurements. We model all other packet arrivals at a given queue using a time-varying Poisson arrival process and assume that the bandwidth B of this process is limited such that

$$B < \frac{1}{2C_1\mu L}. \tag{2}$$

This implies a *quasi*-stationarity; the dynamics of the system are evolving at a rate slow enough that we can discretize at the measurement rate (specifically where the measurements are made) and study the discretized system. We complete our model by imposing a random walk structure on the log-intensity of the traffic arrivals:

$$\log \lambda_k = \log \lambda_{k-1} + w_k, \tag{3}$$

where k denotes the k-th measurement, and w_k is zero-mean Gaussian noise of variance σ^2. The model described thus far induces instantaneous delay pmfs of the form $p_{i,j} \propto \rho_i^j$, where ρ_i is the ratio of the arrival rate and service rate on the i-th link. Such pmfs are exponentially increasing or decreasing, for $\rho > 1$

and $\rho < 1$, respectively. This implies that the mode is either at delay 0 or delay L. In real networks, however, the delay pmfs can display modes a other points due to the non-Poissonian nature of traffic. A straightforward extension of the model above can handle these situations. We introduce an additional dynamical (continuous) parameter κ_i for each link and define the delay pmf as $p_{i,j} \propto \rho_i^{|j-\kappa_i|}$, which places the mode of the pmf near κ_i. The parameter κ_i evolves according to a continuous random walk (with reflection at 0 and L).

We now present our sequential Monte Carlo estimation procedure. For ease of presentation, we describe the special case where $\kappa_i \equiv 0$. The more general case is a straightforward modification. Our goal is the estimation of a delay distribution at each node in the network. Under the model we have just outlined, the notion of a delay distribution is ill-defined. We now define the time-varying delay distribution of window size R at measurement k as:

$$p_{i,j}(R,k) = \frac{1}{R} \sum_{l=k-R+1}^{k} \mathbf{1}_{\{z_i(l)=j\}}, \tag{4}$$

with $z_i(l)$ being the delay experienced at queue i by measurement packets l. Due to the slowly evolving queue dynamics, this is a good approximation (for large R) to a discrete distribution formed by considering the queue length at every arrival and service event between measurements $k - R + 1$ and k.

We would like to track the delay distribution over time. The available observations are a highly non-linear function of the system. As a result, the extended Kalman filter is not suitable for the task, and we use a sequential Monte Carlo algorithm instead. We wish to calculate the following estimate of $p_{i,j}(R,k)$:

$$\begin{aligned}\widehat{p}_{i,j}(R,k) \quad &:= \quad \mathbf{E}_{p(\mathbf{z}_{k-R+1:k}|\mathbf{y}_{0:k})}\left[\sum_{l=k-R+1}^{k} \mathbf{1}_{\{z_i(l)=j\}}\right] \\ &= \frac{1}{R} \sum_{l=k-R+1}^{k} p(z_i(l) = j | y_{0:k}) \\ &= \frac{1}{R} \sum_{l=k-R+1}^{k} \int_0^\infty p(z_i(l) = j | y(l), \lambda_l) p(\lambda_l | y_{0:k})\, d\lambda_l, \end{aligned} \tag{5}$$

where $y(l) \equiv [y_1(l), y_2(l)]$.

Our sequential Monte Carlo algorithm is based on sequential importance sampling techniques [6]. The algorithm makes use of a set of N trajectories or *particles*, each of which represents an independent sample path of the network's dynamical evolution and thus independently explores part of the sample space. Our proposed estimator (5), requires an integration over the density $p(\lambda_l|y_{0:k})$, which cannot be analytically solved. Therefore, we approximate the estimator using Monte Carlo integration. To do this, we must sample from $p(\lambda_l|y_{0:k})$, which itself is not easily accomplished. An alternative approach is to perform *importance* sampling. The basic idea here is to generate draws from an *importance distribution* π_k, which can be sampled from more easily. We use these draws to compute the desired Monte Carlo integration as follows. We can re-write the integration as,

$$\int_0^\infty p(z_i(l) = j | y(l), \lambda_l) \left[\frac{p(\lambda_l|y_{0:k})}{\pi_k(\lambda_l|y_{0:k})}\right] \pi_k(\lambda_l|y_{0:k})\, d\lambda_l.$$

Then, the Monte Carlo estimate is

$$\frac{1}{N} \sum_{v=1}^{N} p(z_i(l) = j | y(l), \lambda_l^{(v)}) w_k^{(v)}, \tag{6}$$

where $w_k^{(v)} = p(\lambda_l^{(v)}|y_{0:k}) \,/\, \pi_k(\lambda_l^{(v)}|y_{0:k})$. We form our approximate estimator, denoted $\widetilde{p}_{i,j}(R,k)$, by replacing the true integral in (5) by its Monte Carlo approximation.

In this paper, we simply use the prior distribution as the importance sampling distribution (*i.e.,* the distribution governing the random walk (3)). In the sequential framework, we wish to obtain at time k an estimate of the distribution $p(\lambda_{0:k}|\mathbf{y}_{0:k})$ without redoing all the work involved in generating the estimate at time $k-1$. This is achieved by forming the trajectory $\lambda_{0:k}^{(v)}$ without modifying the previous trajectory $\lambda_{0:k-1}^{(v)}$, which is possible since importance sampling distribution has a Markovian structure (first-order random walk (3)). At time k, we sample from $\pi_k(\lambda_k|\lambda_{0:k-1}^{(v)}, \mathbf{y}_{0:k})$, and form the time-$k$ particle v by appending $\lambda_k^{(v)}$ to $\lambda_{0:k-1}^{(v)}$.

Degeneracy is a major issue in the application of sequential importance sampling. The variance of the weights increases over time, so that at some stage many importance weights may be very close to zero, and the number of particles contributing to the estimator is greatly reduced. This effect increases the variability of the estimator (compared to the variance one would have with the full N particles contributing). The procedure of *resampling* consists of eliminating particles that have small importance weights and branching the sample paths of particles with substantial weights to create new particles. This ensures that the number of significant weights remains close to N.

This culling and birthing process does introduce some additional computational overhead in the formation of our approximate estimator. Technically, it necessitates "backtracking" the new sample paths over the interval R of interest in our estimator, *i.e.*, performing fixed-interval smoothing [6]. This induces a substantial computational overhead (the complexity of the smoothing algorithm is $O(RN^3)$ per measurement). In simulations, we observe that if we use the approximation (replace $w_k^{(v)}$ by $w_l^{(v)}$)

$$\frac{1}{N} \sum_{v=1}^{N} p(z_i(l) = j | y(l), \lambda_l^{(v)}) w_l^{(v)},$$

for the summation in (6), then we achieve similar performance at a complexity of $O(ML^2N)$ per measurement, where M is the average number of links per path, and L is the number of possible delay units per link.

4. EXPERIMENTS

To assess the performance of our algorithms we simulate (in Matlab) the four-receiver network depicted in Figure 1 below.

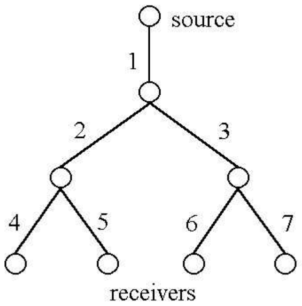

Fig. 1. Network topology in simulation experiments.

Experiment 1: We generate 1000 packet-pair measurements from stationary delay distributions on each link. Figure 2 depicts the true delay distributions on links $2, \dots, 7$ along with the MLEs computed by the EM algorithm. This same experiment is repeated in 50 independent trials. Figure 3 (a) shows the true average delay for each link and the average delay computed from the estimated pmfs. Similar results were obtained with the sequential Monte Carlo procedure in this case.

Experiment 2: We perform 50 independent trials of the scenario in Experiment 1, but this time introduce small, random discrepancies between the delays on shared links. Figure 3 (b) depicts the true average delay for each link and the average delay computed from the estimated pmfs (note the agreement with Figure 3 (a), indicative of the robustness of our methods to such errors).

Experiment 3: We generate 3000 packet-pair measurements from time-varying delay distributions. The temporal dynamics are governed by (3). Figure 4 depicts the true and estimated pmfs (generated by our sequential Monte Carlo algorithm) on links 1, 2, and 7 at two different times. Figure 5 plots the true and estimated average delay on links 2, 4, and 7 as a function of time.

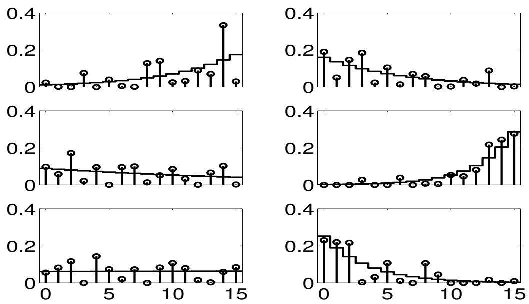

Fig. 2. True (solid) and estimated (stem) delay pmfs for links 2 and 3 (row 1), 4 and 5 (row 2), and 6 and 7 (row 3) using the EM algorithm.

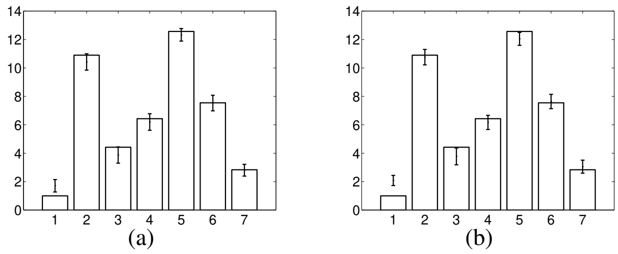

Fig. 3. Estimation of average delays on each link for (a) identical delays on shared links and (b) small delay discrepancies on shared links. Boxes indicate the true average delay on each link (1-7). Error bars denote the one-standard-deviation confidence interval of the estimated average delay (using the EM algorithm).

5. DISCUSSION

Our experiments demonstrate the potential of both the EM and sequential Monte Carlo algorithms for network delay tomography. We find that very good estimates of the delay pmfs can be obtained from a small number of measurements, and estimates of average delays are very robust, even in the presence of non-ideal delay discrepancies on shared links. The sequential Monte Carlo algorithm appears to track slowly varying network behavior reasonably well. Ongoing work is aimed at theoretical analyses of our methods.

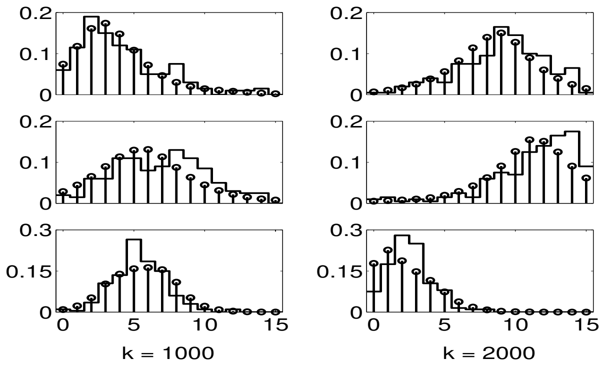

Fig. 4. Delay pmf estimates in nonstationary scenario using sequential Monte Carlo procedure. True (solid) and estimated (stem) pmfs on links 1, 2, and 7 at measurements $k = 1000$ and $k = 2000$ for a window size $R = 200$.

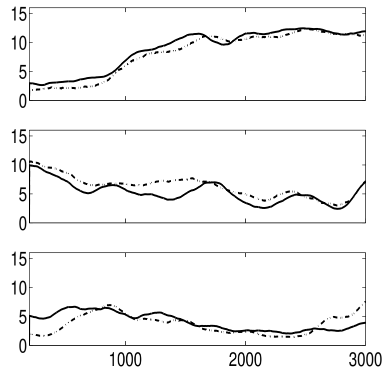

Fig. 5. Tracking of average delay on links 2, 4, and 7 over measurement period. True (solid) and estimated (dashed) average delay versus time.

6. REFERENCES

[1] Y. Vardi, "Network tomography: Estimating source-destination traffic intensities from link data," *J. Amer. Statist. Assoc.*, vol. 91, pp. 365–377, 1996.

[2] F. LoPresti, N.G. Duffield, J. Horowitz, and D. Towsley, "Multicast -based inference of network-internal delay distributions," Tech. Rep., UMass CMPSCI 99-55, 1999.

[3] M. Coates and R. Nowak, "Network loss inference using unicast end-to-end measurement," in *Proc. ITC Seminar on IP Traffic, Measurement and Modelling*, Monterey, CA, Sep. 2000, pp. 28–1–28–9.

[4] N.G. Duffield, F. Lo Presti, V. Paxson, and D. Towsley, "Inferring link loss using striped unicast probes," to appear, *Proc. IEEE Infocom'01*, April 2001. Available as http://www.research.att.com/projects/minc/dlpt00.ps.

[5] B. Frey, *Graphical Models for Machine Learning and Digital Communication*, MIT Press, Cambridge, Massachusetts, 1998.

[6] A. Doucet, N. de Freitas, and N.J. Gordon, Eds., *Sequential Monte Carlo Methods in Practice*, Series Statistics for Engineering and Information Science. Springer-Verlag, New York, 2001.