Online methods for network endpoint localization

Derek Justice and Alfred Hero,
EECS Department, University of Michigan, Ann Arbor, MI 48109-2122, USA
email: hero@umich.edu

Communications and Signal Processing Laboratory
Technical Report: cspl-390
Date: Dec. 30 2008

This TR was submitted to IT Trans on Information Theory, April 2007.

Online Methods for Network Endpoint Localization

Derek Justice and Alfred Hero

Abstract

Online techniques are presented for estimating the source and destination of a suspect transmission through a network based on the activation pattern of sensors placed on network components. A hierarchical Bayesian model is used to relate routing, tracking, and topological parameters. A controlled Markovian routing model is used in conjunction with a recursive EM algorithm to derive adaptive routing and tracking parameter estimates. Previously developed semidefinite programming methods are used to account for any prior topological information through Monte Carlo estimates of the topology parameters. Convergence of the routing and tracking parameter estimates is proven and it is shown that their asymptotic estimates are fixed points of an exact EM algorithm. Approximate methods based on permutation clustering are presented to reduce the complexity of sums that arise in the estimator formulas. A multiarmed bandit approach to the design problem of online probe scheduling is also presented. Finally, the effectiveness of the new methods is illustrated through a variety of tracking simulations inspired by real world scenarios and involving real Internet data. Speedy performance and good accuracy are observed.

Index Terms

Network measurement, recursive EM algorithm, combinatorial sum approximation, statistical inference, nonstochastic multiarmed bandit

Derek Justice (corresponding author): 3M Company, 3M Center, Bldg 518-01-01 St. Paul, MN 55144-1000, 651-736-1441 (Office), 651-736-3122 (Fax) , djustice@mmm.com (email)

Alfred Hero: 4229 EECS, University of Michigan, 1301 Beal Ave, Ann Arbor, MI 48109-2122, 734-763-0564 (Office), 734-763-8041 (Fax), hero@umich.edu (email)

I. Introduction

There are many situations in which the location of a sender or source of a message transmitted in a communication network is unknown. For example, IP spoofing involves the use of a forged source IP address and is often the first step in many denial of service attacks [1], [2]. Multiple suspect transmissions consisting of malicious data packets with forged source IP's are targeted at some destination computer. The only trustworthy information is the ordered activation pattern of routers traversed by the packets along their paths. As such, packet rejection mechanisms [3] and attacker source localization algorithms [4] utilize these patterns along with specific properties of the computer network. An analogous problem involving telephone networks is described in [5]. Here, sensors are placed on telephone lines and are able to indicate when a particular call passes a monitored line. The precise temporal order in which the call passed the sensors is not available because synchronization is difficult. The work in [5] discusses using the activation pattern of sensors associated with a series of probing calls to determine the topology of the network. However, one might also want to use the pattern of activated sensors associated with a suspect call to determine the endpoint nodes (source and destination) when certain parties are communicating.

This paper considers the general problem of online endpoint localization, which involves the use of efficient, recursive estimators for determining the source and destination nodes of a suspect transmission based upon the partially ordered pattern of links or nodes traversed by the transmission along its path through the network. We utilize an abstract mathematical representation of a graph throughout, thereby making our methods applicable to a wide variety of networks. Our results lead to significant reduction in computational complexity and permit online tracking of possibly changing endpoint locations of a suspect message over time. Estimation is based on a hierarchical Bayesian model relating routing, tracking, and topological parameters. Estimates are derived and analyzed using a recursive expectation-maximization (EM) algorithm and semidefinite programming (SDP) methods. Under a complete lack of ordering information, the recursive and exact EM algorithms require a number of operations at each iteration that grow exponentially with the number of sensors activated by a given path. To cope with this problem, we present approximate methods based on permutation clustering that reduce the complexity to only quadratic growth in the number of activated sensors. Some ideas are also given for the design problem; we apply a recursive algorithm to control a multiarmed bandit model for online probe scheduling. Finally, we illustrate the effectiveness of the new methods through experiments involving Internet data.

The measurement apparatus for the system is identical to that described in [6]. It consists of a number

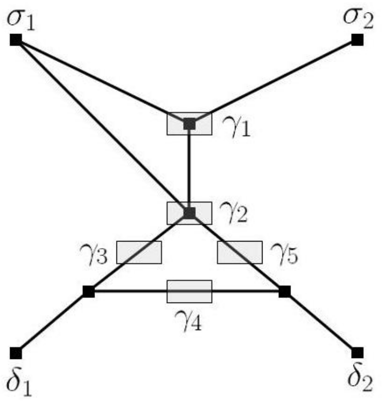


Fig. 1. Diagram of the measurement apparatus on a sample network. Probing sites are sources $\Sigma = \{\sigma_1, \sigma_2\}$ and destinations $\Delta = \{\delta_1, \delta_2\}$. Sensors are $\Gamma = \{\gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5\}$. A box on a link or node represents a sensor that indicates when a transmission path intercepts that link/node. We see γ_1 and γ_2 monitor nodes while γ_3, γ_4, and γ_5 monitor links.

of asynchronous sensors, denoted Γ, placed on some subset of elements (links or nodes) in a network. A sensor is activated, and its activation recorded, whenever the path of a data transmission is intercepted on the element where the sensor is situated. We suppose transmissions originate at some source nodes in a set Σ and terminate at some destination nodes in a set Δ, activating sensors in Γ along the way. The apparatus is illustrated on a sample network in Fig. 1. Individual transmissions produce measurements recorded at discrete time instances, and the subscript k is used throughout the paper to index time. If multiple sensors are activated by a single transmission, they may not be capable of providing the precise order in which they were activated. In general, a probability distribution $P_k(\rho)$ on the possible orders of activation is observed for each measurement; here the argument $\rho \in \{1, 2, \ldots,\}$ is simply a natural number used to indicate a specific ordering of the sensors activated at time k. For example, a transmission with endpoints (σ_1, δ_1) in Fig. 1 might activate sensors $y_1 = \{\gamma_2, \gamma_3\}$–suppose this is the first measurement so $k = 1$. The ordering (γ_2, γ_3), corresponding to $\rho = 1$, might have probability $P_1(1) = \frac{3}{4}$, while the ordering (γ_3, γ_2), where $\rho = 2$, has probability $P_1(2) = \frac{1}{4}$. We are able to probe the network by scheduling a message to be passed from some source $\sigma \in \Sigma$ to some destination $\delta \in \Delta$ and observing the activated sensor set y and ordering distribution $P(\rho)$. Based on the results of our probing observations, we wish to determine the unknown endpoints (source and destination in $\Sigma \times \Delta$) of suspect observations, each consisting of an activated sensor set and ordering distribution.

A Monte Carlo method for endpoint estimation was developed in [6]. This approach averages over feasible sample topologies given a batch of measurements in order to produce endpoint posteriors. It is not clear how one might recursively update posteriors produced in this fashion. We address the updating problem here. Fundamentally, the online model utilizes a generalization of the homogeneous Markovian routing assumption in [7]. We suppose that the next hop in a message's path depends only on its current position and its final destination. This model induces a set of routing parameters θ_{ij}^{d} that represent the probability of going from element i to element j given that the final destination is d. Since the ordering of activated sensors is uncertain, up to a probability distribution $P(\rho)$, the probability of any measured path under the Markovian assumption takes the form of a multinomial mixture distribution parameterized by θ. Endpoint posterior distributions then immediately follow from this model given plug-in estimates of the routing parameters $\hat{\theta}_{ij}^{d}$ and suspect endpoint priors $P(s, d)$. In order to avoid a growing memory problem, we use a recursive form of the EM algorithm [8] to update approximate MAP estimates of the routing parameters when new measurements are made. This is the first of three key approximations necessary to make our method practical for online implementation. The recursive EM method requires that we retain only an information state that summarizes all past measurements, rather than the measurements themselves. We are able to prove, however, that the asymptotic estimates produced by the recursive EM are fixed points of an exact EM algorithm that uses all measurements directly. We utilize an objective function surrogate related to the log-likelihood that permits recursive computation. Other recent works, including [9], [10], use different surrogates to accelerate the EM algorithm and interpret it in the context of a larger class of optimization methods.

Because of the multinomial form of the path likelihoods, it is analytically convenient to make use of Dirichlet priors on the routing parameters [11]. The Dirichlet priors are defined by the hyperparameters β_{ij} for all sources/destinations/sensors i, j. We can then track the endpoints of suspect transmissions by using the suspect observations to compute estimates of the hyperparameters in an empirical Bayes framework [12]. This scheme not only allows the use of suspect measurements to augment the probes in forming a more complete picture of routing in the network, but also localizes which elements of the network are being utilized by the suspects, and thereby tracks them. EM recursions, similar to those used for estimation of the routing parameters θ_{ij}^{d}, update approximate MAP estimates. Any prior information taking the form of linear equalities constraining the unknown network topology's adjacency matrix can be included through a Dirichlet hyperprior on the tracking parameters β_{ij}. Such a characterization is useful because common priors, including vertex degree information and necessary conditions for connectivity of certain network components, can be written as linear constraints on the logical topology's adjacency matrix. The hyperprior is parameterized by γ_{ij}, where these are estimated by averaging over approximately feasible topologies produced using a semidefinite programming (SDP) relaxation generated from the linear prior equalities. The computation of γ_{ij} is done only during an initialization step, so the burden of solving an SDP online is not an issue. This second key approximation allows us to include any topological information with a polynomial time algorithm. Analysis and performance guarantees for the SDP algorithm are presented in [6].

Our models lead to estimator update equations that involve sums over all activated sensor set orderings ρ. The number of such orderings grows exponentially with the number of activated sensors. We may have sufficient synchronization to rule out many of the possible permutations (i.e. $P_k(\rho) = 0$ for most values of ρ). Without such information, however, computing the necessary sums quickly becomes intractable when the number of activated sensors exceeds six or seven. This problem motivates our final crucial approximation, which is to form permutation clusters [13] and use these to compute the sums. The clusters are defined using a sort of augmented generating tree whose construction is driven by the particular estimate values appearing in the sum to be approximated. The tree is built to some depth and then truncated when either the approximation error is tolerable or the number of clusters is too large. In the case that the tree is not truncated, it represents the exact sum by enumerating every possible permutation. The idea of using generating trees for enumerating permutations was first proposed in [13]. It has more recently been applied to the enumeration of restricted permutations [14], and extended to allow for the enumeration of a wide variety of combinatorial objects [15]. Our technique of clustering the permutations is also similar to the separable operator approximations used in [16]. By grouping permutations into clusters, we are effectively decoupling many of the terms that appear in the full sum. This allows us to approximate the full sum with fewer terms that separate according to the clusters.

In addition to estimation and tracking, we can implement probe scheduling online using an algorithm developed for the nonstochastic multiarmed bandit [17]. The idea is to treat each source/destination pair as a different arm on a multiarmed bandit. The multiarmed bandit is a classic problem model used to capture the trade off between exploration and exploitation [18]. Imagine a slot machine with several arms, each giving some unknown reward. The objective is to decide a strategy for pulling the arms so as to maximize your reward over time. The slot machine has a cost per play, so exploration in the form of trying different arms is costly; however if a single arm is played always, one might miss out on exploiting an arm with higher payoff. In our scenario, the reward associated with scheduling a probe between a specific pair is determined by the reduction in the entropy of suspect endpoint posterior distributions resulting from the probe. This sort of information gain criterion has found successful application to sensor management [19], [20]. Under this framework, we can directly apply the Exp3 algorithm of [17]. Exp3 draws the probing pair to be scheduled from a mixture distribution containing two components: a uniform component and a shaped component determined by normalizing some weights. The weights, in turn, are recursively updated in response to observed rewards. The two components of the mixture distribution reflect the explore/exploit trade off; the uniform component promotes even exploration, while the shaped component exploits the high payoff of certain arms. Several performance guarantees are proven for the algorithm in [17]. Along with the performance guarantees, the computational simplicity and recursive nature of the algorithm makes it very suitable for online scheduling.

Endpoint localization is related to the general class of network tomography problems considered in the literature [21], [22], [23], [24]. These problems typically are interested in physical characteristics of a communication network, including link delay and loss. As in [6], we utilize an abstract network model that does not appeal to any physical specifications. These techniques are therefore applicable to endpoint localization in a wide variety of networks, such as those describing partially observed technological, social, or biological structures [25]. Also, the methods may be adapted to produce an online topology inference scheme to address the problem considered in [7]. The method for permutation clustering tackles the general problem of reduced complexity approximations, and may well find applications beyond this present work.

The paper is organized as follows. Section II presents the hierarchical Bayesian model used to explain observed measurements. Section III derives recursive, adaptive estimators for parameters appearing in the hierarchical model, while Section IV analyzes the convergence properties of these estimators. In Section V, we describe the permutation clustering method for approximating certain combinatorial sums that appear in the estimate update equations. Section VI discusses the application of an algorithm for control of the multiarmed bandit to the problem of online probe scheduling. A variety of tracking simulations that apply the new methods to real Internet data collected by Rabbat et. al. [7] are presented in Section VII. These simulations emulate the IP spoofing scenario discussed earlier. We conclude our discussion in Section VIII with a summary and propositions for future work.

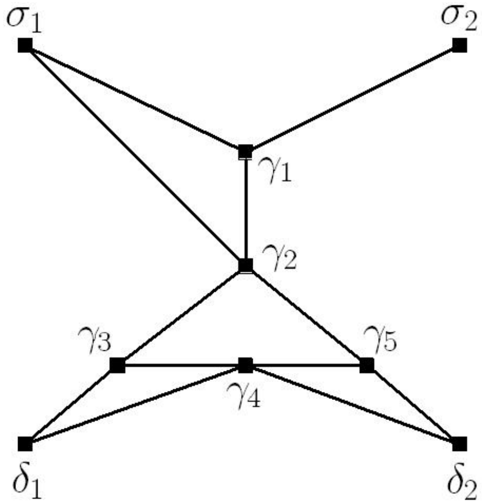


Fig. 2. Example logical topology for the monitored network in Figure 1. The vertex set of the logical network consists of sensors $\Gamma = \{\gamma_i\}_{i=1}^5$ and probing sites $\Sigma = \{\sigma_1, \sigma_2\}$, $\Delta = \{\delta_1, \delta_2\}$. The edges summarize logical adjacencies among sensors and probing sites with any intervening unmonitored elements short-circuited.

II. Hierarchical Bayesian Model

The basis for our online estimation scheme is a hierarchical Bayesian model. A diagram illustrating the relationships among variables in the model is shown in Figure 3A. At the highest level, we have parameters γ_{ij} associated with characteristics (such as vertex degree and connectivity) of the logical topology of the network. The logical topology considers adjacency relationships among only those elements (vertices and edges) that are either monitored with a sensor or used as a probing site. For example, we cannot hope to pinpoint the position of a link in the original network that is not monitored by a sensor. We assume unmonitored elements are essentially 'short-circuited' in the logical network. The idea here is to assure two elements are logically adjacent even if they are physically separated by an element (or subgraph of elements) that is not monitored. An example logical topology is given in Figure 2 for the monitored network in Figure 1.

The topology parameters serve as priors for the tracking parameters β_{ij}. The tracking parameters indicate the extent to which the suspects are utilizing specific parts of the network; they are therefore updated in response to new observed suspects. The routing parameters θ_{ij}^d appear next in the hierarchy. These are updated by the probing measurements, and serve as parameters in a controlled Markovian routing model for observed message paths. One might compare this model to the measurement model of [6], which is depicted in Figure 3B. This model explains observed message paths using only the logical topology A_{ij}, which is constrained by linear equalities in the same way that our γ_{ij} are constrained.

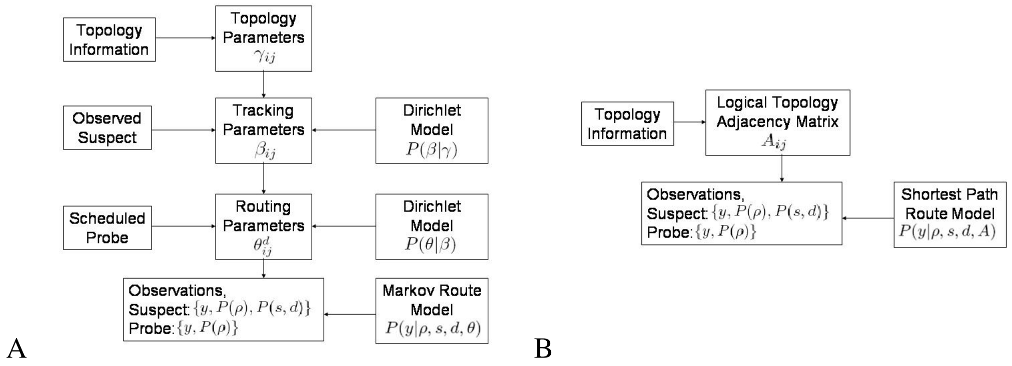

Topology Information
Topology Parameters γ_{ij}
Observed Suspect
Tracking Parameters β_{ij}
Dirichlet Model $P(\beta|\gamma)$
Scheduled Probe
Routing Parameters θ_{ij}^{d}
Dirichlet Model $P(\theta|\beta)$
Observations, Suspect: $\{y, P(\rho), P(s,d)\}$ Probe: $\{y, P(\rho)\}$
Markov Route Model $P(y|\rho, s, d, \theta)$
A
Topology Information
Logical Topology Adjacency Matrix A_{ij}
Observations, Suspect: $\{y, P(\rho), P(s,d)\}$ Probe: $\{y, P(\rho)\}$
Shortest Path Route Model $P(y|\rho, s, d, A)$
B

Fig. 3. Diagram of the hierarchical Bayesian models. The model for our present online system is given in A, while the model of [6] for offline estimation is in B. Vertical arrows represent prior dependencies, while right arrows indicate data used in updating parameter estimates, and left arrows indicate the associated probability models. The model introduces routing and tracking parameters into the hierarchy in order to adaptively account for changes in network routing or suspect location. The method in [6] processes a batch of data offline, so there is no need for adaptation.

The method in [6] processes a batch of data offline, so that there is no need for adaptation. Our model introduces routing and tracking parameters into the hierarchy in order to adaptively account for changes in network routing or suspect location. We will proceed to describe in detail each of the components in the model of Figure 3A from the bottom up.

A. Controlled Markov Routing Model

We use a generalization of the Markovian routing model supposed in [7]. The basic assumption is that the next hop in a message's path through the network depends only on its current position and its final destination. Note that this is a fair assumption for many modern routing algorithms [26]. In contrast, [7] assumes the next hop in a path depends only on the current position of a message, irrespective of the final destination. We let θ_{ij}^{d} denote the probability that a message currently at element (source/sensor) i will go next to element (sensor/destination) j given that its final destination is d. Under this assumption, we can interpret each θ^{d} (for all i, j) as the transition matrix of some Markov chain. Indeed, one may view this as a controlled Markov model where d serves as the control [27]. The model in [7] utilizes a single transition matrix since it does not treat d as a control; note that although our model might be more realistic, it does require more parameters.

The Markov chain assumption implies the following path likelihood model for an activated sensor set

y given the ordering ρ and endpoints s, d:

$$\begin{aligned} P(y|\rho, s, d, \theta) &= \theta^d_{sy^1_\rho} \prod_{n=1}^{|y|-1} \theta^d_{y^n_\rho y^{n+1}_\rho} \theta^d_{y^{|y|}_\rho d} \\ &= \prod_{(i,j)\in\chi_\rho} \theta^d_{ij}, \end{aligned} \tag{1}$$

where $y_\rho \equiv (y^1_\rho, y^2_\rho, \ldots, y^{|y|}_\rho)$ indicates the particular ordering ρ of the activated sensor set y and $\chi_\rho \equiv \{(s, y^1_\rho), (y^1_\rho, y^2_\rho), (y^2_\rho, y^3_\rho), \ldots (y^{|y|-1}_\rho, y^{|y|}_\rho), (y^{|y|}_\rho, d)\}$. From this model, we easily get the endpoint posterior distribution of a suspect measurement y as

$$P(s, d|y, \theta) = \frac{1}{\kappa} \sum_\rho P(y|\rho, s, d, \theta) P(\rho) P(s, d), \tag{2}$$

where $P(\rho)$ is the ordering distribution associated with the measurement, $P(s, d)$ is the endpoint prior, and κ is a normalization constant independent of ρ.

B. *Dirichlet Priors*

Because Eq. (1) is in the form of a multinomial distribution, and its parameters θ^d_i lie on the probability simplex for all d, i, the Dirichlet prior, which is conjugate to the multinomial distribution, provides an analytically tractable scheme for incorporating additional information about θ [11]. The prior is given by

$$P(\theta|\beta) = \frac{1}{\kappa'} \prod_{d=1}^{|\Delta|} \prod_{i=1}^{|\Gamma|+|\Sigma|} \prod_{j=1}^{|\Gamma|+1} (\theta^d_{ij})^{\beta_0 \beta_{ij}}, \tag{3}$$

where conditional independence is assumed across transition matrices indexed by d and rows indexed by i. Although it appears that we have also assumed independence over columns j, there is in fact coupling over columns since all rows θ^d_i must satisfy $\sum_j \theta^d_{ij} = 1$. The tracking parameters β_{ij} in the prior are nonnegative and satisfy $\sum_j \beta_{ij} = 1$ for all i. Also in Eq. (3), we have a normalization constant κ' and a positive precision parameter β_0 that allows one to scale the strength of the prior.

We utilize the law of total probability as follows to derive a likelihood model for an ordered path given the tracking parameters β.

$$P(y|\rho, s, d, \beta) = \mathbf{E}\left[P(y|\rho, s, d, \theta) \mid \beta\right], \tag{4}$$

where the expectation is taken over θ with respect to the prior $P(\theta|\beta)$. In order to evaluate this expectation in closed form, we require that all paths be loopless i.e. $y^m_\rho \neq y^n_\rho$ for all $m \neq n$. This ensures that distinct terms $\{\theta^d_{ij}\}$ appearing in Eq. (1) are conditionally independent (given β) since they all come from different rows of the transition matrix θ^d (recall in defining the prior, we assumed conditional independence over rows). Applying conditional independence and using Eq. (1) allows the expectation to be written as

$$\begin{aligned} P(y|\rho, s, d, \beta) &= \textstyle\prod_{(i,j)\in\chi_\rho} \mathbf{E}\left[\theta^d_{ij}|\beta\right] \\ &= \textstyle\prod_{(i,j)\in\chi_\rho} (1+\beta_0\beta_{ij})/(|\Gamma|+|\Delta|+\beta_0), \end{aligned} \tag{5}$$

where the second line follows from inserting the mean of the Dirichlet distribution in Eq. (3). This is the only result that requires a loopless path. One might still apply these techniques to paths with cycles, however it would then be necessary to compute higher order moments of the Dirichlet distribution and revise subsequent estimators.

As mentioned previously, each row of the tracking parameter matrix also lies in the probability simplex. Again given the multinomial-like product factorization of the likelihood in Eq. (5), it is convenient to assume a Dirichlet prior on these given by

$$P(\beta|\gamma) = \frac{1}{\kappa''} \prod_{i=1}^{|\Gamma|+|\Sigma|} \prod_{j=1}^{|\Gamma|+|\Delta|} (\beta_{ij})^{\gamma_0\gamma_{ij}}. \tag{6}$$

Conditional independence (given γ) over rows is assumed as before. The topology parameters γ_{ij} define this prior, along with a positive scale factor γ_0. We may set the scale factor based on our confidence in the topological information.

Although the exact logical topology of the monitored network is unknown to us, we have available some prior information of the form $Q(A) = v$. Here, $A \in \{0,1\}^{(|\Gamma|+|\Sigma|+|\Delta|)\times(|\Gamma|+|\Sigma|+|\Delta|)}$ is the adjacency matrix of the logical topology, Q is a linear operator, and v is a vector. Through appropriate choices of Q and v, it is possible to define various network priors including cliques, vertex degrees, or even some known portions of the topology. See [6] for some concrete examples of these. We define the topology parameter γ_{ij} as the probability that a logical connection exists between element i and element j given the prior information; i.e.

$$\gamma_{ij} = \mathbf{E}[A_{ij} \mid Q(A) = v]. \tag{7}$$

In this way, one might also use the topology parameters, along with associated precision parameters β_0 and γ_0, to account for knowledge of stable network routing components.

In contrast to independence assumptions for the tracking and routing parameters, it is clear that the topology parameters might share complicated dependencies due to coupling of adjacency elements A_{ij} by the prior equalities. This observation not only strengthens our conditional independence assumptions made earlier, but also illustrates an advantage of the hierarchical Bayesian model. It is usually topological aspects of a network that induce dependencies among routes. We might assume independence of parameters related to routing, provided we condition on topology. Since the topology parameters are placed at the highest level of the Bayesian hierarchy, we are able to exploit conditional independence to simplify computations at lower levels.

III. Parameter Estimation

We now proceed to derive estimators for the parameters introduced in the model of the previous section. It is useful, however, to first give a high level view of the flow of computations necessary for the online system. The system is first initialized by formulating and solving a semidefinite program using prior information about the network that is linear in the logical topology's adjacency matrix (such as vertex degrees). We average over samples produced using the rounding scheme of [6] in order to estimate the topology parameters. With these, we move into online operation of the system. For some initial training period, a probe of the network is made at each tick of the clock. A probe consists of sending a message between some known source and destination and observing the activated sensor set y and ordering distribution $P(\rho)$. We update routing parameters in response to the results of each new probe. After the training phase ends, we begin monitoring for suspect transmissions–i.e. transmissions whose source and destination are unknown. Each suspect measurement includes an activated sensor set y, ordering distribution $P(\rho)$, and prior over possible endpoints $P(s, d)$. When a suspect is observed, the tracking parameters are updated, which forces an update in the routing parameters. The best available estimates of the routing parameters may be used at any given time to build endpoint posteriors of observed suspects. An operational diagram of the system is given in Figure 4.

In the following, we will first define the precise estimation problems that we wish to solve. We then proceed to derive the estimators associated with each stage of the system: initialization (topology parameters), training (routing parameters), and monitoring (tracking parameters).

A. Estimation Objectives and Problem Statement

Assuming the probability of each edge A_{ij} is unknown, our goal in the initialization phase is to produce a suitable Monte Carlo approximation of the expectation in Eq. (7) in order to estimate the topology parameters. Once this is done, we move to online operation.

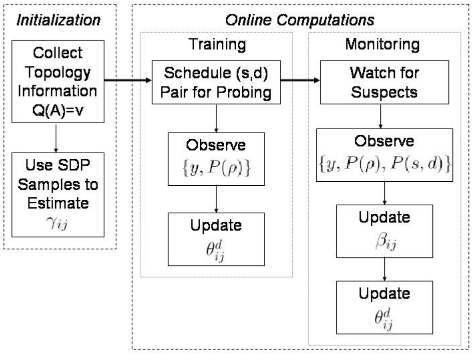


Fig. 4. Operational diagram of the online system. Heavy horizontal arrows indicate transitions between stages of operation (initialization, training, and monitoring), while light vertical arrows indicate the flow of computation within each stage. The system is initialized by formulating and solving a semidefinite program (SDP) associated with the prior equality constraints $Q(A) = v$ on the logical adjacency matrix A. Once online operation commences, we have a training phase in which probes are scheduled and routing parameter estimates are recursively updated in response to probe observations. Next we monitor the network for suspect transmissions, and update tracking and routing parameter estimates whenever a suspect is observed. Refer to Figure 3 for parameter definitions and relations.

We desire estimators of the routing and tracking parameters to be recursive (so as to avoid growing memory problems) and scalable (so that online computation does not become intractable as the size of the problem increases). We also want estimators that adapt to changes in routing protocols and suspect locations. With that in mind, we choose the following penalized likelihood objective for estimation of the routing parameters.

$$\phi_k(\theta) = \sum_t a^{k-t} l_t(\theta) + \log P(\theta|\hat{\beta}(k)), \tag{8}$$

where $a \in [0, 1)$ is a forgetting factor, and $\hat{\beta}(k)$ is a plug-in estimate of the prior parameters β at time k. Note that here and throughout the paper, k indexes the current clock tick. The log-likelihood is explicitly given by

$$l_t(\theta) = \log\left(\sum_\rho P(y_t|\rho, s_t, d_t, \theta) P_t(\rho)\right), \tag{9}$$

where $P(y_t|\rho, s_t, d_t, \theta)$ follows from the model in Eq. (1).

Note that the objective in Eq. (8) would be precisely the maximum a posteriori (MAP) objective under an i.i.d. measurement model if the forgetting factor was unity. The introduction of the factor $a < 1$ is a common heuristic used in the design of adaptive algorithms to reduce the effect of old measurements on current parameter estimates [12]. One might set a using knowledge of dynamic routing in the network; e.g. the more quickly standard routes are expected to change in the network, the closer to zero a should be set. Alternatively, there exists a Dirichlet type generative model for measurements taken over time for which the objective in Eq. (8) yields precisely the MAP estimate. It is straightforward to write down such a model, so we omit it here.

The Dirichlet prior on θ serves to condition the routing parameters θ_{ij}^{d} by the tracking parameters β_{ij}. A reasonable way to estimate β is to make use of the observed suspect transmissions. If the suspect frequently utilizes the link from i to j, we would like β_{ij} to be closer to one. In this way, we fill in gaps in the probing measurements by making direct use of the suspects to form a more complete picture of routing in the network. Note also, that by taking account of the links being used by the suspects, we are essentially tracking their positions in the network as characterized by sensor activations. The estimates of β can be combined with the endpoint posterior distribution to provide additional information about the suspects' locations.

The estimation of the tracking parameters β is formalized through the use of empirical Bayes techniques [12]. We choose a similar sort of adaptive MAP objective for estimation of the tracking parameters.

$$\phi_k(\beta) = \sum_t b^{k-t} l_t(\beta) + \log P(\beta|\hat{\gamma}). \tag{10}$$

Again, a forgetting factor $b \in [0, 1)$ is used to discount old suspect measurements, and $\hat{\gamma}$ is the static plug-in estimate of the prior parameters γ. The log-likelihood is given by

$$l_t(\beta) = \log \left(\sum_{s,d} \sum_{\rho} P(y_t|\rho, s, d, \beta) P_t(\rho) P_t(s, d) \right), \tag{11}$$

where $P(y_t|\rho, s, d, \beta)$ is the likelihood of some suspect measurement y_t given in Eq. (5). In order to allow for adaptation, one might tune b by knowledge of a suspect's motion through some network.

Ideally, our routing and tracking parameter estimates at time k, $\hat{\theta}(k)$ and $\hat{\beta}(k)$, would maximize the objective functions in Eqs. (8) and (10) respectively. The goal in online estimation is therefore to develop scalable, recursive algorithms to optimize these functions. We shall see that the key qualifiers 'scalable' and 'recursive' will necessitate certain approximations to be made. Our estimates will therefore only approximately optimize the selected objective functions. By plugging in routing parameter estimates, we can also compute the endpoint posterior distributions of suspect measurements as given in Eq. (2).

B. Initialization with Topology Information

If the edge probabilities $P(A_{ij} = 1)$ in the logical topology are unknown, except for some prior constraints on the logical adjacency matrix $Q(A) = v$, we must approximate the expectation of Eq.

(7). Let $\{A^m\}_{m=1}^M$ be the sample adjacencies produced from a semidefinite program (SDP) formulated to approximately solve the prior equalities $Q(A) = v$ as described in [6]. The samples will satisfy $\|Q(A^m) - v\| < \epsilon(Q, v)$ for a known tolerance guarantee $\epsilon(Q, v)$ and all m. We produce a Monte Carlo estimate of the expectation in Eq. (7) as follows

$$\hat{\gamma}_{ij} = \frac{1}{M} \sum_m A_{ij}^m. \tag{12}$$

Note that solving an SDP is typically a very demanding computational task ($O(N^7)$ for a matrix of size $N \times N$ [28]). Fortunately, we need only solve the SDP once, during an offline initialization phase.

C. Online Routing Parameter Estimation

The form of the likelihood in Eq. (9) suggests the EM algorithm as a natural candidate for implementing the estimator [29]. Exact maximization of Eq. (8) via EM would require storing all past probing measurements. In order to avoid this growing memory problem, we utilize a recursive form of the EM algorithm described in [8] to update the maximum value of an approximation to Eq. (8). Recursive EM approximates the likelihood term $\sum_t a^{k-t} l_t(\theta)$ by $L_k(\theta)$, which is obtained recursively as follows:

$$L_k(\theta) = \mathbf{E}\left[\log P(y_k|\rho, s_k, d_k, \theta) \mid y_k, s_k, d_k, P_k(\rho), \hat{\theta}(k-1)\right] + aL_{k-1}(\theta), \tag{13}$$

where $L_0(\theta) = 0$ and $P(y_k|\rho, s_k, d_k, \theta)$ is taken as 1 if a probe is not scheduled at time k (in order to remain consistent with the likelihood term in Eq. (8)). Evaluating the expectation in Eq. (13) over orderings ρ and regrouping terms yields

$$L_k(\theta) = \sum_{d,i,j} c_{ij}^d(\hat{\theta}(k-1); k) \log \theta_{ij}^d + aL_{k-1}(\theta), \tag{14}$$

with $c_{ij}^d(\theta; k)$ given by the following for $d = d_k$:

$$c_{ij}^d(\theta; k) = \frac{\sum_{\rho \mid (i,j) \in \chi_{k,\rho}} P(y_k|\rho, s_k, d_k, \theta) P_k(\rho)}{\sum_\rho P(y_k|\rho, s_k, d_k, \theta) P_k(\rho)}. \tag{15}$$

We have $c_{ij}^d(\theta; k) = 0$ if $d \neq d_k$; also $c_{ij}^d(\theta; k) = 0$ for all d if a probe is not scheduled at time k. If we define the recursion for $\bar{c}(k)$ as

$$\bar{c}_{ij}^d(k) = c_{ij}^d(\hat{\theta}(k-1); k) + a\bar{c}_{ij}^d(k-1), \tag{16}$$

with $\bar{c}^d_{ij}(0) = 0$ for all d, i, j, then we can express the function $L_k(\theta)$ simply as

$$L_k(\theta) = \sum_{d,i,j} \bar{c}^d_{ij}(k) \log \theta^d_{ij}. \tag{17}$$

The routing parameter estimates at time k are then given by

$$\begin{aligned} \hat{\theta}(k) &= \underset{\theta | \sum_j \theta^d_{ij} = 1 \forall d,i}{\arg\max} \quad \tilde{\phi}_k(\theta) \\ &= \underset{\theta | \sum_j \theta^d_{ij} = 1 \forall d,i}{\arg\max} \quad \textstyle\sum_{d,i,j} \left(\bar{c}^d_{ij}(k) + \beta_0 \hat{\beta}_{ij}(k) \right) \log \theta^d_{ij}. \end{aligned} \tag{18}$$

A simple application of the KKT conditions to this concave maximization gives the following routing parameter estimates:

$$\hat{\theta}^d_{ij}(k) = \frac{\bar{c}^d_{ij}(k) + \beta_0 \hat{\beta}_{ij}(k)}{\sum_l \bar{c}^d_{il}(k) + \beta_0 \hat{\beta}_{il}(k)}. \tag{19}$$

Eqs. (16) and (19) define the recursive routing parameter estimator. Note that these parameter estimates, although derived from a stochastic routing model, will indeed converge to a deterministic route if for a given element i we always observe a transition to element j_* when the destination is d. The forgetting factors $a, b < 1$ ensure that the estimates $\hat{\theta}^d_{ij}$ will be driven to some minimal value (depending on γ_{ij}, γ_0, and β_0) for all $j \neq j_*$ with almost all of the mass of $\hat{\theta}^d_i$ concentrated on $\hat{\theta}^d_{ij_*}$

D. Online Tracking Parameter Estimation

Comparing Eqs. (10) and (8) indicate tracking parameter estimation problem is almost identical to routing parameter estimation. The only fundamental difference in computation, is that a sum over source/destination pairs also appears inside the logarithm of Eq. (11). This is a consequence of our lack of knowledge of suspect endpoints. We again apply the recursive EM approximation for the likelihood term in the objective:

$$L_k(\beta) = \mathbf{E}\left[\log P(y_k | \rho, s, d, \beta) \mid y_k, P_k(s, d), P_k(\rho), \hat{\beta}(k-1) \right] + b L_{k-1}(\beta), \tag{20}$$

where $L_0(\beta) = 0$ and $P(y_k|\rho, s, d, \beta)$ is taken as 1 if a suspect is not observed at time k. After evaluating the expectation over ordering ρ and endpoints s, d, we can write $L_k(\beta)$ as

$$L_k(\beta) = \sum_{i,j} \bar{g}_{ij}(k) \log(1 + \beta_0 \beta_{ij}), \tag{21}$$

with $\bar{g}(k)$ defined recursively as

$$\bar{g}_{ij}(k) = g_{ij}(\hat{\beta}(k-1); k) + b\bar{g}_{ij}(k-1), \tag{22}$$

where $\bar{g}_{ij}(0) = 0$ for all i, j. The factor depending on the new measurement $g_{ij}(\beta; k)$ is given by

$$g_{ij}(\beta; k) = \frac{\sum_{\rho,s,d \mid (i,j)\in x_{k,\rho}} P(y_k|\rho, s, d, \beta) P_k(\rho) P_k(s, d)}{\sum_{\rho,s,d} P(y_k|\rho, s, d, \beta) P_k(\rho) P_k(s, d)}, \tag{23}$$

with $g_{ij}(\beta; k)$ taken as zero if a suspect is not observed at time k.

We replace the likelihood term $\sum_t b^{k-t} l_t(\beta)$ in Eq. (10) with $L_k(\beta)$ from Eq. (21) to arrive at the following expression for $\hat{\beta}(k)$

$$\begin{aligned} \hat{\beta}(k) &= \underset{\beta \mid \sum_j \beta_{ij} = 1 \forall i}{\arg\max} \quad \tilde{\phi}_k(\beta) \\ &= \underset{\beta \mid \sum_j \beta_{ij} = 1 \forall i}{\arg\max} \quad \textstyle\sum_{i,j} \bar{g}_{ij}(k) \log(1 + \beta_0 \beta_{ij}) + \gamma_0 \hat{\gamma}_{ij} \log \beta_{ij}. \end{aligned} \tag{24}$$

If one attempts to apply the KKT conditions as before, a system of quadratic equations results. We encounter this problem because of the sum inside the first logarithm. The familiar structure suggests a generalized EM framework wherein another EM iteration is used to increase the likelihood as an alternative to solving the quadratic system. We add $\bar{g}_{ij} \log \frac{2}{\beta_0 + 2}$ to the objective and combine with the first term to give

$$\hat{\beta}(k) = \underset{\beta \mid \sum_j \beta_{ij} = 1 \forall i}{\arg\max} \sum_{i,j} \bar{g}_{ij}(k) \log\left(\frac{2}{\beta_0 + 2} + \frac{\beta_0}{\beta_0 + 2} 2\beta_{ij}\right) + \gamma_0 \hat{\gamma}_{ij} \log \beta_{ij}. \tag{25}$$

We now recognize the first term as the logarithm of a uniform and linear mixture distribution. Applying EM to Eq. (25) in the standard way gives the operator f^k, defined component-wise as

$$f^k_{ij}(\beta) = \frac{\bar{g}_{ij}(k) \frac{\beta_0 \beta_{ij}}{1 + \beta_0 \beta_{ij}} + \gamma_0 \hat{\gamma}_{ij}}{\sum_l \bar{g}_{il}(k) \frac{\beta_0 \beta_{il}}{1 + \beta_0 \beta_{il}} + \gamma_0 \hat{\gamma}_{il}}. \tag{26}$$

The new estimate $\hat{\beta}(k)$ is the fixed point of the operator f^k. Provided $\hat{\gamma}_{ij} > 0$ for all i, j, the optimization in Eq. (25) is strictly concave. Since the Q-function from which Eq. (26) is derived is also continuous in both arguments, it follows that EM will converge to the unique global maximum [30]. Thus the fixed point of f^k is unique. In practice, we can obtain $\hat{\beta}(k)$ by initializing (with perhaps $\hat{\beta}(k-1)$) and then

Action	Online Computation
Probe Scheduled	1. $\bar{c}^d_{ij}(k) = \frac{\sum_{\rho \mid (i,j) \in x_{k,\rho}} P(y_k \mid \rho, s_k, d_k, \hat{\theta}(k-1)) P_k(\rho)}{\sum_{\rho} P(y_k \mid \rho, s_k, d_k, \hat{\theta}(k-1)) P_k(\rho)} + a\bar{c}^d_{ij}(k-1)$ 2. $\hat{\theta}^d_{ij}(k) = \frac{\bar{c}^d_{ij}(k) + \beta_0 \hat{\beta}_{ij}(k)}{\sum_{j'} \bar{c}^d_{ij'}(k) + \beta_0 \hat{\beta}_{ij'}(k)}$
Suspect Observed	1. $\bar{g}_{ij}(k) = \frac{\sum_{\rho, s, d \mid (i,j) \in x_{k,\rho}} P(y_k \mid \rho, s, d, \hat{\beta}(k-1)) P_k(\rho) P_k(s,d)}{\sum_{\rho, s, d} P(y_k \mid \rho, s, d, \hat{\beta}(k-1)) P_k(\rho) P_k(s,d)}$ $+b\bar{g}_{ij}(k-1)$ 2. $f^k_{ij}(\beta) = \frac{\bar{g}_{ij}(k) \frac{\beta_0 \beta_{ij}}{1 + \beta_0 \beta_{ij}} + \gamma_0 \hat{\gamma}_{ij}}{\sum_{j'} \bar{g}_{ij'}(k) \frac{\beta_0 \beta_{ij'}}{1 + \beta_0 \beta_{ij'}} + \gamma_0 \hat{\gamma}_{ij'}}$ 3. $\hat{\beta}(k) = f^k \circ f^k \circ \ldots f^k(\hat{\beta}(k-1)) = (f^k)^N(\hat{\beta}(k-1))$ 4. $\hat{\theta}^d_{ij}(k) = \frac{\bar{c}^d_{ij}(k) + \beta_0 \hat{\beta}_{ij}(k)}{\sum_{j'} \bar{c}^d_{ij'}(k) + \beta_0 \hat{\beta}_{ij'}(k)}$

TABLE I
SUMMARY OF ONLINE COMPUTATIONS.

successively applying the operator f^k until $||(f^k)^N(\beta) - (f^k)^{N-1}(\beta)|| < \epsilon$ for some tolerance ϵ. So that

$$\hat{\beta}(k) = f^k \circ f^k \circ \ldots f^k(\hat{\beta}(k-1)) = (f^k)^N(\hat{\beta}(k-1)), \tag{27}$$

where N is chosen large enough to satisfy a desired tolerance ϵ. In practice, we have observed that this internal EM iteration converges very quickly; our simulations indicated an N value of 2 or 3 was typically sufficient to obtain convergence with a tolerance of $\epsilon = 10^{-8}$. The recursions in Eqs. (22) and (27) define the tracking parameter estimator. Table I provides a summary of all online computations. Note that the recursions for $\bar{c}(k)$ and $\bar{g}(k)$ define first order discrete time systems with poles given by a and b respectively, with $a, b \in [0, 1)$. As mentioned before, a and b should therefore be chosen based on the rate at which old observations are expected to become obsolete. Old observations are appropriately downweighted in this way.

IV. Convergence Analysis

An asymptotic analysis of the unconstrained recursive EM algorithm is presented in [8] by using a quadratic expansion of the likelihood to relate the algorithm to a stochastic approximation method. When constraints are present, the mathematical form of the expansion's optimum no longer reveals an obvious mapping to the stochastic approximation. We will argue convergence of our algorithms directly, instead of attempting to derive such a mapping. We also show that the recursive approximation produces asymptotic estimates that are fixed points of the EM algorithm applied to the exact objective for MAP estimation.

Before beginning, note that the quantities we consider are in fact random variables, so that all equalities or inequalities hold with probability one. Consider first the sequences $\bar{g}(k)$ and $\bar{c}(k)$ as defined in Eqs. (22) and (16) respectively.

Lemma 1 *The sequence $\bar{g}(k)$ converges to some limit $\bar{g}(\infty)$ as $k \to \infty$. Similarly, $\bar{c}(k) \to \bar{c}(\infty)$ as $k \to \infty$.*

Proof: The recursive expression in Eq. (22) can be written in closed form as follows

$$\bar{g}_{ij}(k) = \sum_{t=1}^{k} b^{k-t} g_{ij}(\hat{\beta}(t-1);t). \tag{28}$$

Now from the definition in Eq. (23), we see that $0 \leq g_{ij}(\beta;k) \leq 1$ for all k because all involved probabilities are nonnegative and every term in the numerator sum also appears in the denominator sum. It follows that $\bar{g}_{ij}(k)$ is a monotone nondecreasing sequence. Furthermore, we have

$$\begin{aligned} \bar{g}_{ij}(k) &\leq \textstyle\sum_{t=1}^{k} b^{k-t} \\ &= \tfrac{1-b^k}{1-b} \\ &\leq \tfrac{1}{1-b}, \end{aligned} \tag{29}$$

since $b < 1$. Thus $\bar{g}_{ij}(k)$ is also bounded from above for all k. It follows that the sequence must converge to some value $\bar{g}_{ij}(\infty)$ as $k \to \infty$. We have thus established convergence of each i,j component; we therefore have $\bar{g}(k) \to \bar{g}(\infty)$. An identical argument establishes $\bar{c}(k) \to \bar{c}(\infty)$ as $k \to \infty$. ■

The parameter estimates as defined in Eqs. (18) and (24) are the optimal values of strictly concave functions. Given that $\bar{c}$ and $\bar{g}$ define these functions and converge, we argue convergence of the estimates through uniform convergence of the functions they optimize. The following lemma is key to this argument.

Lemma 2 *Let $\{f_k\}$ and f_∞ be strongly concave functions over a compact, convex set C such that f_k converges to f_∞ uniformly over C as $k \to \infty$. If x_k denotes the maximum of f_k over C for all $k \in \{1, 2, \ldots \infty\}$, then x_k exists and is unique for all k, and furthermore $x_k \to x_\infty$ as $k \to \infty$.*

Proof: Since each function f_k is strongly concave, we have immediately that its optimizer over a compact, convex set exists and is unique. Now suppose x_k does not converge to x_∞, so there exists $\epsilon_* > 0$ such that $||x_k - x_\infty|| \geq \epsilon_*$ for all $k < \infty$. A Taylor expansion of f_k about the maximum gives

$$f_k(x_\infty) = f_k(x_k) + \nabla f_k(x_k)^T (x_\infty - x_k) + \frac{1}{2}(x_\infty - x_k)^T \nabla^2 f_k(z)(x_\infty - x_k), \tag{30}$$

where $z = \alpha x_k + (1-\alpha) x_\infty \in C$ for some $\alpha \in [0,1]$. Optimality of x_k ensures $-\nabla f_k(x_k)^T (x_\infty - x_k) \geq 0$, and strong concavity of f_k implies there is some $m > 0$ such that $-\nabla^2 f_k(z) \succeq mI$ for all k [31]. We therefore rearrange Eq. (30) and apply these inequalities to arrive at

$$f_k(x_k) - f_k(x_\infty) \geq \frac{m}{2}\epsilon_*^2, \tag{31}$$

for all $k < \infty$.

Now uniform convergence of f_k to f_∞ ensures that for all $\epsilon > 0$, there is some $n(\epsilon)$ such that $|f_k(x) - f_\infty(x)| < \epsilon$ for all $k \geq n(\epsilon)$ and any $x \in C$. Consider any index k_* satisfying $k_* \geq n\left(\frac{m}{8}\epsilon_*^2\right)$; uniform convergence gives

$$\begin{aligned} f_{k_*}(x_{k_*}) \quad &< f_\infty(x_{k_*}) + \tfrac{m}{8}\epsilon_*^2 \\ f_{k_*}(x_\infty) \quad &> f_\infty(x_\infty) - \tfrac{m}{8}\epsilon_*^2. \end{aligned} \tag{32}$$

But the inequality in (31) must hold for all k, so we may substitute the inequalities from (32) into (31) to obtain

$$f_\infty(x_{k_*}) - f_\infty(x_\infty) \geq \frac{m}{4}\epsilon_*^2. \tag{33}$$

This contradicts the assumption that x_∞ is the unique maximizer of f_∞ over C. ■

In order to apply Lemma 2, we must decide on a compact, convex set C over which uniform convergence holds. Because of the $\log(\theta_{ij}^d)$ term in Eq. (18), it is convenient to bound the routing parameter estimates away from zero. Note that this can be achieved by including a sample adjacency matrix consisting of all ones into the sum of Eq. (12); the extra sample would ensure positivity while having a negligible effect on the topology parameter estimates, provided M is sufficiently large. Indeed, we have $\hat{\gamma}_{ij} \in [1/M, 1]$ for all i, j. This filters down to $\hat{\beta}(k)$ and $\hat{\theta}(k)$ through Eqs. (19) and (26). One can easily verify that $\hat{\beta}(k) \in C_\beta$ and $\hat{\theta}(k) \in C_\theta$ for all k, where C_β and C_θ are compact, convex sets defined by

$$\begin{aligned} C_\beta \quad &\equiv \left\{ \beta \in \left[\tfrac{\gamma_0(1-b)}{M(|\Gamma|+|\Delta|)(1+\gamma_0(1-b))}, 1\right]^{(|\Gamma|+|\Sigma|)\times(|\Gamma|+|\Delta|)} \mid \textstyle\sum_j \beta_{ij} = 1 \forall i \right\} \\ C_\theta \quad &\equiv \left\{ \theta \in \left[\tfrac{\gamma_0\beta_0(1-b)(1-a)}{M(|\Gamma|+|\Delta|)(|\Gamma|+1)(1+\gamma_0(1-b))(1+\beta_0(1-a))}, 1\right]^{(|\Gamma|+|\Sigma|)\times(|\Gamma|+1)\times|\Delta|} \mid \textstyle\sum_j \theta_{ij}^d = 1 \forall d, i \right\}. \end{aligned} \tag{34}$$

Enforcing positivity in this fashion allows us to ignore the inequality constraints in deriving the expressions in Eqs. (19) and (26), since they automatically satisfy the bounds by design.

This leads to the primary convergence theorem below.

Theorem 1 *The tracking and routing parameter estimates converge as $k \to \infty$; that is $\hat{\beta}(k) \to \hat{\beta}(\infty)$ and $\hat{\theta}(k) \to \hat{\theta}(\infty)$.*

Proof: We first show uniform convergence of $\tilde{\phi}_k(\beta)$ as defined in Eq. (24) to

$$\tilde{\phi}_\infty(\beta) \equiv \sum_{ij} \bar{g}_{ij}(\infty) \log(1 + \beta_0 \beta_{ij}) + \gamma_0 \hat{\gamma}_{ij} \log \beta_{ij}. \tag{35}$$

Lemma 1 implies that for all $\epsilon > 0$, there is some $n_g(\epsilon)$ such that $||\bar{g}(k) - \bar{g}(\infty)|| < \epsilon$ for all $k \geq n_g(\epsilon)$. In order to show uniform convergence, we take $n_1^* \equiv n_g\left(\frac{\epsilon}{(|\Gamma|+|\Sigma|)(|\Gamma|+|\Delta|)\log(1+\beta_0)}\right)$ for any given $\epsilon > 0$. For any $k \geq n_1^*$ we have

$$\begin{aligned} \left|\tilde{\phi}_k(\beta) - \tilde{\phi}_\infty(\beta)\right| &= \left|\textstyle\sum_{i,j} \log(1+\beta_0\beta_{ij})(\bar{g}_{ij}(k) - \bar{g}_{ij}(\infty))\right| \\ &\leq \textstyle\sum_{i,j} |\log(1+\beta_0\beta_{ij})||\bar{g}_{ij}(k) - \bar{g}_{ij}(\infty)| \\ &< \tfrac{\epsilon}{(|\Gamma|+|\Sigma|)(|\Gamma|+|\Delta|)\log(1+\beta_0)} \textstyle\sum_{i,j} |\log(1+\beta_0\beta_{ij})| \\ &\leq \epsilon, \end{aligned} \tag{36}$$

where $\beta \in C_\beta$. Thus we have uniform convergence of $\tilde{\phi}_k(\beta)$ to $\tilde{\phi}_\infty(\beta)$ over C_β. Since $\hat{\gamma}_{ij} \geq 1/M$ for all i, j, the functions $\tilde{\phi}_k(\beta)$ and $\tilde{\phi}_\infty(\beta)$ are strongly concave over C_β; so Lemma 2 immediately gives convergence of $\hat{\beta}(k)$ to $\hat{\beta}(\infty)$ (the maximum value of $\tilde{\phi}_\infty(\beta)$).

In a similar fashion, we show uniform convergence of $\tilde{\phi}_k(\theta)$ as defined in Eq. (18) to

$$\tilde{\phi}_\infty(\theta) \equiv \sum_{d,i,j} \left(\bar{c}_{ij}^d(\infty) + \beta_0 \hat{\beta}_{ij}(\infty)\right) \log \theta_{ij}^d. \tag{37}$$

Lemma 1 ensures that for all $\epsilon > 0$, there is some $n_c(\epsilon)$ such that $||\bar{c}(k) - \bar{c}(\infty)|| < \epsilon$ for all $k \geq n_c(\epsilon)$. And let $n_\beta(\epsilon)$ ensure $||\hat{\beta}(k) - \hat{\beta}(\infty)|| < \epsilon$ for all $k \geq n_\beta(\epsilon)$. To show uniform convergence, take $n_2^* \equiv \max\left\{n_c\left(\frac{\epsilon}{2\kappa|\log\theta_{min}|}\right), n_\beta\left(\frac{\epsilon}{2\beta_0\kappa|\log\theta_{min}|}\right)\right\}$ where $\kappa \equiv |\Delta|(|\Gamma| + |\Sigma|)(|\Gamma| + 1)$ and θ_{min} is the lower bound in C_θ of Eq. (34). We then have for any $k \geq n_2^*$

$$\begin{aligned} \left|\tilde{\phi}_k(\theta) - \tilde{\phi}_\infty(\theta)\right| &\leq \textstyle\sum_{d,i,j} |\log\theta_{ij}^d| \left(|\bar{c}_{ij}^d(k) - \bar{c}_{ij}^d(\infty)| + \beta_0|\hat{\beta}_{ij}(k) - \hat{\beta}_{ij}(\infty)|\right) \\ &< \epsilon. \end{aligned} \tag{38}$$

Again the functions are strongly concave over C_θ since $\hat{\beta}_{ij}(k) > 0$; we therefore apply Lemma 2 to give convergence of $\hat{\theta}(k)$ to the maximum value of $\tilde{\phi}_\infty(\theta)$, denoted $\hat{\theta}(\infty)$. ■

We can use the convergence results just established to analyze the relationship between the recursive approximation and the exact EM algorithm for large k. Before proceeding, we establish a useful lemma about the limit points of the sequences $\bar{g}(k)$ and $\bar{c}(k)$.

Lemma 3 *The limit points $\bar{g}(\infty)$ and $\bar{c}(\infty)$ alluded to in Lemma 1 are given explicitly by*

$$\begin{aligned} \bar{g}_{ij}(\infty) &= \lim_{k\to\infty} \textstyle\sum_{t=1}^{k} b^{k-t} g_{ij}(\hat{\beta}(\infty);t) \\ \bar{c}_{ij}^{d}(\infty) &= \lim_{k\to\infty} \textstyle\sum_{t=1}^{k} a^{k-t} c_{ij}^{d}(\hat{\theta}(\infty);t), \end{aligned} \tag{39}$$

for all d, i, j.

Proof: It is useful to first establish Lipschitz continuity of the functions $g_{ij}(\beta;k)$ and $c_{ij}^{d}(\theta;k)$ over C_β and C_θ respectively. The derivative of $g_{ij}(\beta;k)$ as in Eq. (23) satisfies

$$\begin{aligned} \left\| \tfrac{\partial g_{ij}}{\partial \beta}(\beta;k) \right\| &\le 2\left((|\Gamma|+|\Sigma|)(|\Gamma|+|\Delta|)\right)^{1/2} \Big(\textstyle\sum_{\rho,s,d} P(y_k|\rho,s,d,\beta) P_k(\rho) P_k(s,d) \Big)^{-2} \\ &\le 2\left((|\Gamma|+|\Sigma|)(|\Gamma|+|\Delta|)\right)^{1/2} (|\Gamma|+|\Delta|+\beta_0)^{2(|\Gamma|+1)} \\ &= L_g, \end{aligned} \tag{40}$$

where the second line follows from Eq. (5), with $\beta \in C_\beta$. We therefore have that L_g is a Lipschitz constant over C_β independent of k and i, j [32]. In a similar fashion, one can establish a Lipschitz constant for $c_{ij}^{d}(\theta;k)$ over C_θ as $L_c = 2\kappa^{1/2}\theta_{min}^{-2(|\Gamma|+1)}$ where θ_{min} and κ are as defined in the proof of Theorem 1.

We proceed now with the main result. By Theorem 1 $\hat{\beta}(k)$ converges, and Lipschitz continuity of $g_{ij}(\beta;t)$ implies that for all $\epsilon > 0$ there is some $n(\epsilon)$ such that $|g_{ij}(\hat{\beta}(k-1);t) - g_{ij}(\hat{\beta}(\infty);t)| < L_g\epsilon$ for all $k \ge n(\epsilon)$. For any given $\epsilon > 0$, take $n^* \equiv n\left(\frac{(1-b)\epsilon}{2L_g}\right) + \max\left\{1, \frac{\log((1-b)\epsilon/2)}{\log b} - 1\right\}$. We then have for all $k \ge n^*$

$$\begin{aligned} |\bar{g}_{ij}(k) - \textstyle\sum_{t=1}^{k} b^{k-t} g_{ij}(\hat{\beta}(\infty);t)| &\le \textstyle\sum_{t=1}^{k} b^{k-t} |g_{ij}(\hat{\beta}(t-1);t) - g_{ij}(\hat{\beta}(\infty);t)| \\ &< \tfrac{(1-b)\epsilon}{2} \textstyle\sum_{t=n(\frac{(1-b)\epsilon}{2L_g})}^{k} b^{k-t} + \textstyle\sum_{t=1}^{n(\frac{(1-b)\epsilon}{2L_g})-1} b^{k-t} \\ &< \tfrac{\epsilon}{2} + \tfrac{b^{k-n((1-b)\epsilon/(2L_g))+1}}{1-b} \\ &< \epsilon, \end{aligned} \tag{41}$$

where the second term in the second line follows because $g_{ij}(\beta;t) \in [0,1]$ for all β, t. The argument for $\bar{c}_{ij}^{d}(\infty)$ is identical. ■

Suppose we apply standard EM to optimize the tracking parameter objective $\phi_k(\beta)$ as in Eq. (10). Performing the E step averages over orderings and endpoints of each individual measurement and results in the following Q-function:

$$Q_k(\beta|\tilde{\beta}) = \sum_{i,j}\left(\gamma_0\hat{\gamma}_{ij}\log\beta_{ij} + \log(1+\beta_0\beta_{ij})\sum_{t=1}^{k} b^{k-t} g_{ij}(\tilde{\beta};t)\right). \tag{42}$$

If we wish to optimize the routing parameter objective $\phi_k(\theta)$ in Eq. (8) using exact EM, the E step averages over orderings only and gives a similar Q-function.

$$Q_k(\theta|\tilde{\theta}) = \sum_{d,i,j}\log\theta_{ij}^d\left(\beta_0\hat{\beta}_{ij}(k) + \sum_{t=1}^{k} a^{k-t} c_{ij}^d(\tilde{\theta};t)\right). \tag{43}$$

Note that each measurement defines $g(\beta;t)$ or $c(\theta;t)$ for a single clock tick t. Thus we require all past measurements in order to compute the Q functions. The EM algorithm then proceeds to iteratively maximize the Q functions until a fixed point is reached. The recursive approximation maintains only a summary of the past measurements in $\bar{g}$ and $\bar{c}$. The following theorem shows that the asymptotic estimates produced by the recursive approximation will in fact be fixed points of the exact EM algorithm as $k \to \infty$.

Theorem 2 *If $\beta_Q(k)$ and $\theta_Q(k)$ denote the maximizers of $Q_k(\beta|\hat{\beta}(\infty))$ over C_β and $Q_k(\theta|\hat{\theta}(\infty))$ over C_θ respectively, then $\beta_Q(k) \to \hat{\beta}(\infty)$ and $\theta_Q(k) \to \hat{\theta}(\infty)$ as $k \to \infty$.*

Proof: Notice the structure of $Q_k(\beta|\hat{\beta}(\infty))$ is the same as that of $\tilde{\phi}_k(\beta)$ as defined in Eq. (24), with the term $\bar{g}_{ij}(k)$ replaced by $\sum_{t=1}^{k} b^{k-t} g_{ij}(\hat{\beta}(\infty);t)$. Similarly, if we replace $\bar{c}_{ij}^d(k)$ in $\tilde{\phi}_k(\theta)$ of Eq. (18) with $\sum_{t=1}^{k} a^{k-t} c_{ij}^d(\hat{\theta}(\infty);t)$, we arrive at $Q_k(\theta|\hat{\theta}(\infty))$. Thus we can use Lemma 3 to construct an argument that exactly parallels the proof of Theorem 1. ■

Although the parameter estimates arrive at fixed points of the exact EM algorithm asymptotically, we are not guaranteed that these are in fact maxima of the appropriate objective functions. This is because EM might not converge to a maximum of the likelihood. The work in [33] gives an extensive analysis of this issue.

V. Permutation Clustering

The posterior computation in Eq. (2) and the update formulas in Eqs. (15), (23) require evaluating sums of the form

$$\sum_{\rho} P_k(\rho) \prod_{(i,j)\in\chi_{k,\rho}} v_{ij}, \tag{44}$$

where v is some parameter (e.g. θ^d or $1+\beta_0\beta$) and $\rho = 1, 2, \ldots |y_k|!$ indexes different permutations. The number of terms in this sum therefore grows exponentially with the number $|y_k|$ of activated sensors. It is not feasible to compute these sums online when more than 5 or 6 sensors are activated. However, we might have some ordering information that could rule out many of these permutations, i.e. $P_k(\rho) = 0$ for most of the orderings ρ. If no ordering information is available, computing the sums directly is hopeless for long paths. In the remainder of this section and the next, we formulate a combinatorial scheme for approximating the sums under such conditions where the path is long (say, $|y_k| > 6$) and $P_k(\rho) = 1/|y_k|!$ for all ρ. We assume a uniform ordering distribution throughout this discussion for simplicity. However, these techniques easily extend to the situation where some permutations have larger probabilities and the rest are equally likely by simply changing the weighting scheme.

A. Permutation Approximation Algorithm

A key point to notice in developing an approximation algorithm is that the index set of Eq. (44) satisfies $\chi_{k,\rho} \subset S_k \equiv y_k^2 \cup \Sigma \times y_k \cup y_k \times \Delta - \bigcup_{n=1}^{|y_k|}(y_k^n, y_k^n)$ for all ρ. The number of distinct terms in the product therefore grows only as $|y_k|^2$, even though the total number of terms in the sum grows exponentially in $|y_k|$. Suppose further that all of the parameters v_{ij} for $(i, j) \in S_k$ are similar. In this case, we could obtain a reasonable approximation to the sum in Eq. (44) by the following.

$$\sum_\rho \frac{1}{|y_k|!} \prod_{(i,j)\in\chi_{k,\rho}} v_{ij} \approx \bar{v}_0^{|y_k|+1}, \tag{45}$$

where $\bar{v}_0$ is the geometric mean of $\{v_{ij} \mid (i, j) \in S_k\}$. This approximation essentially clusters all $|y_k|!$ permutations into a single term. Note that we need only that the geometric mean of $\{v_{ij}|(i, j) \in \chi_{k,\rho}\}$ be similar for all ρ for the approximation in Eq. (45) to hold. Although this is a weaker condition than requiring v_{ij} be similar for all $(i, j) \in S_k$, it is much harder to verify precisely because there is an exponential number of orderings ρ. We can refine the approximation iteratively by removing elements from S_k and including all valid permutations over such elements in the sum. For example, we arrive at a first refinement of the approximation in Eq. (45) by setting $C_1 \equiv S_k - (i_1, j_1)$ and including $v_{i_1 j_1}$ explicitly in the sum.

$$\sum_\rho \frac{1}{|y_k|!} \prod_{(i,j)\in\chi_{k,\rho}} v_{ij} \approx \frac{|y_k|! - (|y_k|-1)!}{|y_k|!} \bar{v}_1^{|y_k|+1} + \frac{(|y_k|-1)!}{|y_k|!} v_{i_1 j_1} \bar{v}_2^{|y_k|}, \tag{46}$$

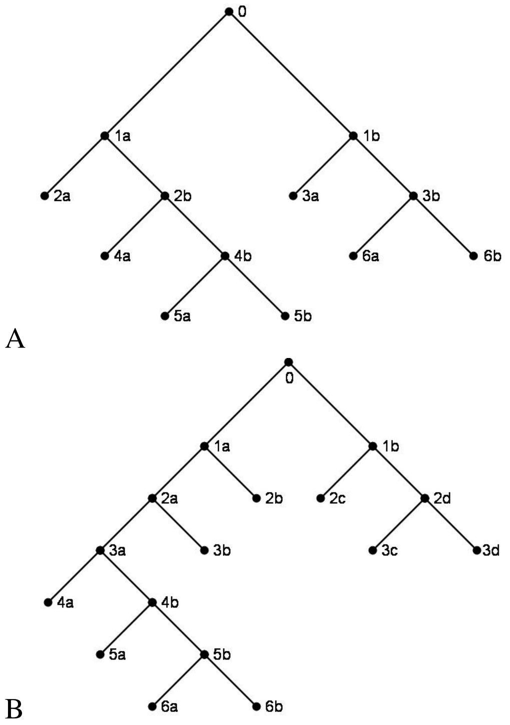
where $\bar{v}_l$ is the geometric mean of $\{v_{ij} \mid (i,j) \in C_l\}$, and C_2 is defined by all elements $(i,j) \in S_k$ such that the sequence (i,j) could exist in a valid permutation with the sequence (i_1, j_1) (we denote this by $(i,j) \sim (i_1, j_1)$). We could continue to produce refinements of the sum approximation in this way until S_k is empty and all $|y_k|!$ permutations appear in the sum. This describes the essential idea of the sum approximation algorithm.

We utilize an ordered version of the set S_k, denoted $\tilde{S}_k$, along with a binary tree to organize the terms in the sum. At each refinement step, the first element of $\tilde{S}_k$ is removed and used to update the binary tree. Each node of the tree is characterized by three quantities: Z, α, and C. The characteristic Z is the set of all $(i,j) \in S_k$ such that v_{ij} appears explicitly in the sum term represented by that node, α is the number of permutations that are clustered into the term, and C is the set of all $(i,j) \in \tilde{S}_k$ such that $(i,j) \sim Z$ (that is, the set of all (i,j) such that $(i,j) \cup Z$ might form a valid permutation). For example, the characteristics associated with first term in Eq. (46) are $Z = \phi$, $\alpha = |y_k|! - (|y_k| - 1)!$, and $C = S_k - (i_1, j_1)$. The pseudocode for the permutation clustering approximation is as follows.

Algorithm 1 *:*

```
• Given a parameter sequence $\tilde{S}_k$, define a tree root with characteristics $\alpha = |y_k|!$, $Z = \phi$, and $C = \tilde{S}_k$. Set $L = 1$.
• While $\tilde{S}_k$ is nonempty and $L \leq L_{\max}$:
    – Set $(i_*, j_*) = \tilde{S}_k(1)$ and $\tilde{S}_k = \tilde{S}_k - \tilde{S}_k(1)$.
    – For all leaves $l$ such that $(i_*, j_*) \in C_l$ and $\alpha_l > 0$:
        ∗ Add a left child to $l$ with characteristics $Z = Z_l$ and $C = C_l - (i_*, j_*)$.
        ∗ Add a right child to $l$ with characteristics $Z = Z_l \cup (i_*, j_*)$ and $C = \{(i,j) \in \tilde{S}_k \mid (i,j) \sim Z_l\}$.
    – Number all new leaves and those existing leaves with $\alpha > 0$ in order of increasing $|Z|$ with the integers $1, 2, \ldots, L$.
    – For $l = L, L-1, \ldots 1$:
        ∗ Set $\alpha_l = (|y_k| - |Z_l|)! - \sum_{i|i>l, Z_l \subset Z_i} \alpha_i$.
• Construct the sum represented by all leaves with $\alpha > 0$.
```

Example binary trees produced by this algorithm for two different parameter sequences are given in Figure 5. Since at each step we only form new leaves that might be permutations, it is clear that all per-



Node	α	Z	C
0	2	ϕ	$\{(1,2),(1,d),(s,1),$ $(2,1),(s,2),(2,d)\}$
1a	1	ϕ	$\{(1,d),(s,1),(2,1),$ $(s,2),(2,d)\}$
1b	1	$\{(1,2)\}$	$\{(s,1),(2,d)\}$
2a	0	ϕ	$\{(s,1),(2,1),(s,2),$ $(2,d)\}$
2b	1	$\{(1,d)\}$	$\{(2,1),(s,2)\}$
3a	0	$\{(1,2)\}$	$\{(2,d)\}$
3b	1	$\{(s,1),(1,2)\}$	$\{(2,d)\}$
4a	0	$\{(1,d)\}$	$\{(s,2)\}$
4b	1	$\{(2,1),(1,d)\}$	$\{(s,2)\}$
5a	0	$\{(2,1),(1,d)\}$	ϕ
5b	1	$\{(s,2),(2,1),(1,d)\}$	ϕ
6a	0	$\{(s,1),(1,2)\}$	ϕ
6b	1	$\{(s,1),(1,2),(2,d)\}$	ϕ

Node	α	Z	C
0	2	ϕ	$\{(1,2),(s,1),(2,d),$ $(2,1),(s,2),(1,d)\}$
1a	1	ϕ	$\{(s,1),(2,d),(2,1),$ $(s,2),(1,d)\}$
1b	1	$\{(1,2)\}$	$\{(s,1),(2,d)\}$
2a	1	ϕ	$\{(2,d),(2,1),(s,2),$ $(1,d)\}$
2b	0	$\{(s,1)\}$	$\{(2,d)\}$
2c	0	$\{(1,2)\}$	$\{(2,d)\}$
2d	1	$\{(s,1),(1,2)\}$	$\{(2,d)\}$
3a	1	ϕ	$\{(2,1),(s,2),(1,d)\}$
3b	0	$\{(2,d)\}$	ϕ
3c	0	$\{(s,1),(1,2)\}$	ϕ
3d	1	$\{(s,1),(1,2),(2,d)\}$	ϕ
4a	0	ϕ	$\{(s,2),(1,d)\}$
4b	1	$\{(2,1)\}$	$\{(s,2),(1,d)\}$
5a	0	$\{(2,1)\}$	$\{(1,d)\}$
5b	1	$\{(s,2),(2,1)\}$	$\{(1,d)\}$
6a	0	$\{(s,2),(2,1)\}$	ϕ
6b	1	$\{(s,2),(2,1),(1,d)\}$	ϕ

Fig. 5. Example permutation clustering trees. Here, just two sensors 1 and 2 are activated. The tree A utilizes the parameter sequence $\tilde{S}_k = ((1,2),(1,d),(s,1),(2,1),(s,2),(2,d))$, while tree B uses the sequence $\tilde{S}_k = ((1,2),(s,1),(2,d),(2,1),(s,2),(1,d))$. Nodes 1a and 1b are formed after the first element in the sequence ($(1,2)$ for both A and B) is appended, nodes 2a, 2b, 2c, and 2d are formed after the second element in the sequence ($(1,d)$ for A and $(s,1)$ for B) is appended, and so on. Each tree produces a complete enumeration of the permutation set with characteristic quantities given in the tables to the right. For tree A, nodes 5b and 6b give the complete permutation set, while nodes 3d and 6b are the complete permutation set in tree B. It is clear from this example that the order of the parameter sequence $\tilde{S}_k$ will have a large impact on the formation of the tree. A greedy heuristic for selecting this is described and justified in the next section.

mutations will have been enumerated once $\tilde{S}_k$ is empty. One might contrast this method with the standard permutation generating tree due to [13]. The standard method enumerates all permutations of $\{1,2,\dots n\}$ by recursively forming children $(k+1,\pi_1,\pi_2,\dots,\pi_k),(\pi_1,k+1,\pi_2,\dots,\pi_k),\dots(\pi_1,\pi_2,\dots,\pi_k,k+1)$ to a given parent node $(\pi_1,\pi_2,\dots,\pi_k)$. Note that our method is not strictly a generating tree, since the α characteristic of a child may depend on nodes other than its parent [14]. Our method is desirable, however, in that it allows more direct control over the order in which permutations (or partial permutations) are generated–through the ordering of the set $\tilde{S}_k$–without the need for complicated backtracking through the tree. This is important because we rarely generate the entire tree in the case of long paths. Indeed, once some maximum number of leaves $L_{\max}$ are accumulated, we truncate the tree to obtain the following approximation:

$$\sum_{\rho} \prod_{(i,j)\in\chi_{k,\rho}} v_{ij} \approx \sum_{l=1}^{L} \alpha_l \bar{v}_l^{|y_k|+1-|Z_l|} \prod_{(i,j)\in Z_l} v_{ij}, \tag{47}$$

where $\bar{v}_l$ is the geometric mean of $\{v_{ij} \mid (i,j) \in C_l\}$.

In the numerators of Eqs. (15) and (23), it is necessary to compute restricted versions of the above sum; instead of summing over all ρ we only consider ρ such that some $(i,j) \in \chi_{k,\rho}$. The approximation is exactly as in Eq. (47) if it happens that the particular (i,j) has been removed from $\tilde{S}_k$ and added to Z_l for some leaf l in the tree before truncation. If we truncate before adding (i,j) to the tree, then we approximate by considering a weighted sum over leaves that might form a valid permutation with (i,j) (i.e. all leaves l such that $(i,j) \in C_l$). The restricted sum approximation is therefore

$$\sum_{\rho|(i,j)\in\chi_{k,\rho}} \prod_{(i',j')\in\chi_{k,\rho}} v_{i'j'} \approx \begin{cases} \sum_{l|(i,j)\in Z_l} \alpha_l \bar{v}_l^{|y_k|+1-|Z_l|} \prod_{(i',j')\in Z_l} v_{i'j'} & \text{if } (i,j) \notin \tilde{S}_k \\ \frac{(|y_k|-1)!}{\sum_{l|(i,j)\in C_l} \alpha_l} \sum_{l|(i,j)\in C_l} \alpha_l \bar{v}_l^{|y_k|+1-|Z_l|} \prod_{(i',j')\in Z_l} v_{i'j'} & \text{if } (i,j) \in \tilde{S}_k. \end{cases} \tag{48}$$

When considering sums over permutations with different source/destinations, as in Eqs. (2) and (23), we can simply follow this procedure to approximate the sum over ρ for each pair (s,d).

B. Permutation Approximation Analysis

The sequence $\tilde{S}_k$ and the truncation limit $L_{\max}$ will determine approximations to the functions $c(\theta;k)$ and $g(\beta;k)$ as defined in Eqs. (15) and (23), respectively. Provided matching $\tilde{S}_k$ and $L_{\max}$ values are used for the recursive and a comparable non-recursive EM iteration as in Eqs. (43) and (42), one can easily verify that all necessary properties of the functions $c(\theta;k)$ and $g(\beta;k)$ hold to ensure validity of the previous convergence analysis. The permutation clustering approximation greatly decreases the complexity associated with computing $c(\theta;k)$ and $g(\beta;k)$. Full computation of these functions requires $O(|y_k|!)$ time. The permutation approximation reduces this to a low order polynomial in $|y_k|$. Suppose we fix $L_{\max}$ so that it does not grow with $|y_k|$. Note that each time an element from $\tilde{S}_k$ is added to the tree, the number of leaves at most doubles so that we can always ensure truncation before $L_{\max}$ is exceeded. The only operation that scales is computation of the characteristic C associated with new right children, and the geometric mean $\bar{v}$ corresponding to this set. In determining C, we need to check that the conditions defining a permutation are not violated if any of the pairs $(i,j) \in C$ are added to the set Z. This can be done recursively by simply removing any elements from the parent's C characteristic that might result in any repetitions or incomplete paths from s to d after augmenting the parent's Z characteristic with (i_*, j_*). The cardinality of any C is at most $|y_k|^2 + 2|y_k|$. It follows that the permutation clustering approximation reduces the complexity from exponential in $|y_k|$ to $O(|y_k|^2)$.

We now develop some bounds on how well the permutation clustering approximation agrees with the full sum over all permutations. First note that all permutations ρ clustered into a given leaf l must satisfy $\chi_{k,\rho} \subset Z_l \cup C_l$, and $Z_l \cap C_l = \phi$ by definition of these characteristics. If we define $\bar{v}_{l,\min}$ as the geometric mean of the $|y_k| + 1 - |Z_l|$ smallest elements of $\{v_{ij} \mid (i,j) \in C_l\}$ and $\bar{v}_{l,\max}$ as the geometric mean of the $|y_k| + 1 - |Z_l|$ largest elements of this set, then we have the following inequalities for any ρ such that $\chi_{k,\rho} \subset Z_l \cup C_l$.

$$\bar{v}_{l,\min}^{|y_k|+1-|Z_l|} \prod_{(i,j)\in Z_l} v_{ij} \leq \prod_{(i,j)\in\chi_{k,\rho}} v_{ij} = \left(\prod_{(i,j)\in\chi_{k,\rho}\cap C_l} v_{ij}\right)\left(\prod_{(i,j)\in Z_l} v_{ij}\right) \leq \bar{v}_{l,\max}^{|y_k|+1-|Z_l|} \prod_{(i,j)\in Z_l} v_{ij}. \tag{49}$$

It is also obvious that $\bar{v}_l^{|y_k|+1-|Z_l|} \prod_{(i,j)\in Z_l} v_{ij}$ lies within the bounds of Eq. (49), since $\bar{v}_l$ is the geometric mean of *all* elements in $\{v_{ij} \mid (i,j) \in C_l\}$. Now, the leaves represent a partition of the permutation set, so we have

$$\sum_{\rho} \prod_{(i,j)\in\chi_{k,\rho}} v_{ij} = \sum_{l} \sum_{\rho|\chi_{k,\rho}\subset Z_l\cup C_l} \prod_{(i,j)\in\chi_{k,\rho}} v_{ij}. \tag{50}$$

We can then combine Eq. (50) with the inequalities in (49) and realize that α_l is the number of permutations ρ that satisfy $\chi_{k,\rho} \subset Z_l \cup C_l$ to arrive at the following bound on the approximation error in Eq. (47).

$$\left|\sum_{\rho} \prod_{(i,j)\in\chi_{k,\rho}} v_{ij} - \sum_{l=1}^{L} \alpha_l \bar{v}_l^{|y_k|+1-|Z_l|} \prod_{(i,j)\in Z_l} v_{ij}\right| \leq \sum_{l=1}^{L} \alpha_l \left(\bar{v}_{l,\max}^{|y_k|+1-|Z_l|} - \bar{v}_{l,\min}^{|y_k|+1-|Z_l|}\right) \prod_{(i,j)\in Z_l} v_{ij}. \tag{51}$$

One can arrive at bounds for the approximation error associated with Eq. (48) in a similar fashion. When $(i,j) \notin \tilde{S}_k$, the form of the bound is almost identical to that in Eq. (51) with the only difference being a restriction on the sum (only over l such that $(i,j) \in Z_l$). The bound is looser when $(i,j) \in \tilde{S}_k$ because we do not know the correct proportions for including each leaf in the sum. There is some loss associated with the weighted sum approximation in Eq. (48). It is straightforward to apply these results to determine bounds on the actual estimators when the permutation clustering approximation is used. Clearly, there is no approximation error if the geometric mean of the $|y_k|+1-|Z_l|$ smallest elements of $\{v_{ij} \mid (i,j) \in C_l\}$ is equal to the geometric mean of the $|y_k| + 1 - |Z_l|$ largest elements of the set for all l. Since C_l is always a subset of $\tilde{S}_k$, this suggests a reasonable strategy is to choose the ordering $\tilde{S}_k$ so as to reduce the range of $\{v_{ij} \mid (i,j) \in \tilde{S}_k\}$ as much as possible each time an element is removed from $\tilde{S}_k$. This is a simple greedy approach to the problem of selecting the parameter sequence, however one might pose some optimization problem for selecting the sequence that is best in a nonmyopic setting. In most cases, such an optimization would result in additional online computational strain.

VI. Online Probe Scheduling

Here we propose some methods for online probe scheduling. Previously, we assumed the training phase occurred before the monitoring phase. In this section, it is necessary to consider a different training paradigm wherein probes of the network are scheduled during observation downtime, that is, when a suspect observation is not observed. These consist of scheduled transmissions from some known source to some known destination and noting the activated sensor set and ordering distribution. Although they might seem secondary to observing suspects, probes are necessary for us to learn the routing parameters of the network. It might not be clear which are the best probes to make until we go online and begin recording measurements. A rapid, online scheduling algorithm is certainly advantageous in this paradigm.

We model the probe scheduling problem as a multiarmed bandit. Each different source/destination pair, that is each distinct element of $\Sigma \times \Delta$, is a separate arm of the bandit. The reward associated with scheduling some pair (s,d) is given by the information gained as a result of the probe. We use the change in entropy of the suspect endpoint posteriors as a measure of information gain. The reward r_{sd} for scheduling (s,d) is therefore given by

$$r_{sd} = \sum_t \lambda_t \Delta H(P(s,d|y_t,\hat{\theta}(k))), \tag{52}$$

where λ_t are constants that sum to one and allow a weighted average of the entropy change ΔH in all observed suspect posteriors. Given the reward function, we can directly apply the Exp3 algorithm of [17] for control of the multiarmed bandit. Exp3 uses a parameter $\delta \in (0,1]$ and is based on the following recursions:

$$\begin{aligned} p_i &= (1-\delta)\tfrac{w_i(k-1)}{\sum_j w_j(k-1)} + \tfrac{\delta}{|\Sigma\times\Delta|} \\ w_i(k) &= \begin{cases} w_i(k-1)\exp\left(\frac{\delta r_{sd}}{p_{sd}|\Sigma\times\Delta|}\right) & \text{if } i=(s,d) \\ w_i(k-1) & \text{else,} \end{cases} \end{aligned} \tag{53}$$

where $w_i(0) = 1$ for all $i \in \Sigma \times \Delta$. It is clear that p is a mixture distribution over the endpoint pairs consisting of a uniform component and a component shaped by the rewards. At each time step, the endpoint pair to be scheduled is chosen from p. There are several versions of the algorithm with slightly different asymptotic performance guarantees. We refer the reader to [17] for a thorough theoretical treatment.

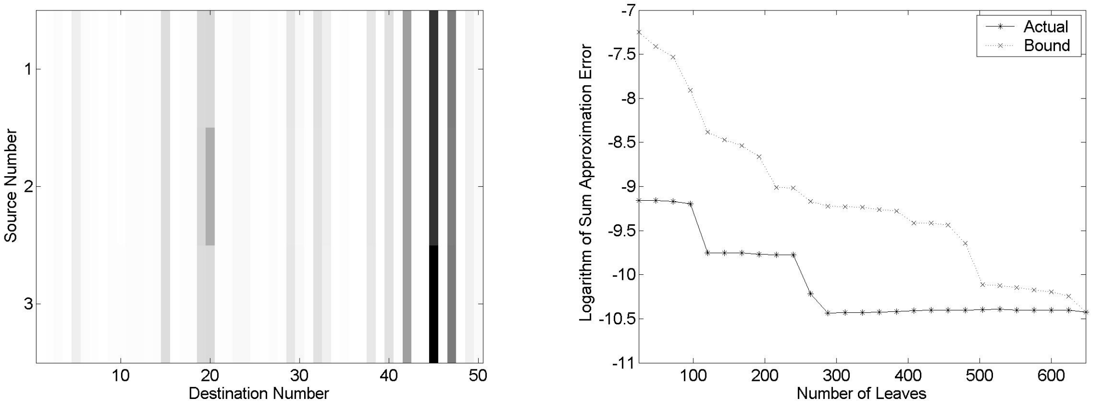
VII. Experimental Results

We applied the new online estimation methods to the `traceroute` data presented in [7]. The data was obtained from `traceroute` probes initiated on October 12, 2005 from three sources located at the University of Wisconsin-Madison, the Instituto Superior Tecnico in Lisbon, Portugal, and Rice University in Houston, Texas to fifty destination web servers of various companies, universities, and governments around the world. We treat the routers encountered as sensors, and ignore all ordering information, so that the ordering distributions $P_k(\rho)$ associated with all measurements ρ are uniform. After processing the data to collapse identical routers–that is routers that are always activated together across the 150 measurements–we were left with 241 routers and path lengths ranging from 2 to 14 hops, with an average of 7.5. In light of the long paths and uniform ordering distributions, enumeration of permutations was infeasible so we had to apply the permutation clustering approximations to compute parameter estimates. For purposes of initialization, we assumed all edge probabilities γ_{ij} were set to 0.5. Precision parameter values of $\gamma_0 = 0.0002$ and $\beta_0 = 1$ were used for all experiments.

Our first simulation illustrates the accuracy of the permutation clustering approximation that is used in subsequent experiments. We next present the core simulation of a moving suspect who is transmitting with spoofed IP's. The suspect moves through the network transmitting from different sources, and we are able to track its position using the tracking parameter estimates along with the endpoint posteriors. This sort of situation might arise in a variety of law enforcement scenarios, among others. After investigating this application, we simulate another real-world situation: a sensor failure. We show how adaptation of the routing parameter estimates is able to detect this. Finally, we investigate the effectiveness of the multiarmed bandit scheduling algorithm by analyzing the evolution of the distribution from which scheduled probes are drawn.

A. Permutation Clustering Approximation Error

We devised an experiment to test the accuracy of the permutation clustering method for approximating combinatorial sums that arise in our estimators. After initializing the system as described above, we trained using observed paths between all 150 possible source/destination combinations. Training was done with



A: True Endpoint Posterior

B: Permutation Clustering Appx. Error

Fig. 6. Illustration of the permutation sum approximation accuracy. In A, we have the exact endpoint posterior of a suspect transmission activating six sensors. Darker color indicates higher value in this two dimensional distribution. The true endpoints of this suspect were source 3 and destination 45; so we see that the correct destination is clearly pinpointed while there is a bit of ambiguity in the source estimate. Plot B shows the error (in a logarithmic scale) when permutation clustering is used to approximate this endpoint posterior. The number of leaves in the clustering tree were varied from 24 up to 648 in steps of 24. Asterisks connected by a solid line indicate the actual error (as on the left side of Eq. (51)), while X's connected by a dotted line indicate the error bound on the right side of Eq. (51).

minimal forgetting–that is, a forgetting factor $a = 0.999999$. Then a single suspect transmission passed between source 3 and destination 45 was observed during the monitoring stage. This suspect activated a total of six sensors, thus we were able to compute its exact endpoint posterior as in Eq. (2) by summing over all 720 orderings. The exact posterior is shown in Figure 6A. We then used the permutation clustering method to approximate the endpoint posterior as in Eq. (47) with number of leaves L ranging from 24 up to 648 in steps of 24. The absolute error as on the left side of Eq. (51) is plotted in Figure 6B, along with the derived error bound.

We see that the permutation clustering approximation performs quite well, falling into the realm of round-off error after about 300 leaves are used in the tree. Furthermore, the actual approximation error is about a hundred times smaller than the worst case bound for fewer leaves (24, 48, 72). This suggests it is reasonable to proceed with application of this method in the following simulations. In all remaining simulations, we utilize permutation clustering trees having at most 24 leaves for parameter updates and posterior computations.

B. Suspect Tracking

This experiment simulates the movement of a suspect through the network and illustrates the tracking abilities of the proposed methods. As before, we begin by initializing the system and training it using all 150 source/destination pairs with minimal forgetting. Then, for the first 100 clock ticks of the monitoring phase, we observe suspect transmissions emanating from source 1 and terminating at random destinations. We observe transmissions from source 2 to random destinations for the next 100 clock ticks. For the final 100 clock ticks of monitoring, the suspect moves to source 3 and transmits to random destinations. A forgetting factor of $b = 0.9$ is used throughout in estimation of the tracking parameters β.

Our goal is to determine which source node the suspect is transmitting from at each tick of the clock. One natural indicator of location is simply the instantaneous source posterior distribution given by

$$P_s(k) \propto \sum_d \sum_\rho \prod_{(i,j) \in \chi_{k,\rho}} \hat{\theta}^d_{ij}(k), \tag{54}$$

where a uniform endpoint prior $P(s,d)$ is assumed, and proportionality (rather than equality) is used because we have omitted a normalization constant. In addition to the instantaneous source posterior, one might look at the values of the tracking parameters associated with sensors that are exclusive to each source. In particular, we say a sensor is exclusive to source s if it is only activated when s is probed. We are able to use our probing measurements from the training phase to determine the exclusivity of the various sensors. Based on this notion, we define the average entering probability $E_s(k)$ at time k associated with source s as follows.

$$E_s(k) \propto \frac{1}{|\{j|\ j \text{ is exclusive to } s\}|} \sum_{j|\ j \text{ is exclusive to } s} \sum_i \hat{\beta}_{ij}(k). \tag{55}$$

The 'entering probability' nomenclature follows from an analogy to Markov chains: since $\hat{\beta}(k)$ is the transition matrix of a Markov chain, the quantity defined in Eq. (55) can be interpreted as the probability of suspect measurements entering sensors exclusive to source s. Thus the larger $E_s(k)$ is, the more messages are entering sensors exclusive to s, and thus it is more likely that the suspect is transmitting from s. Also, each row of the matrix $\hat{\beta}(k)$ is a probability distribution itself, with $\hat{\beta}_{ij}(k)$ representing the probability a suspect measurement exits element i and arrives in element j. We can utilize the average exit distribution entropy $H_s(k)$ of sensors exclusive to s as another location indicator. This quantity is

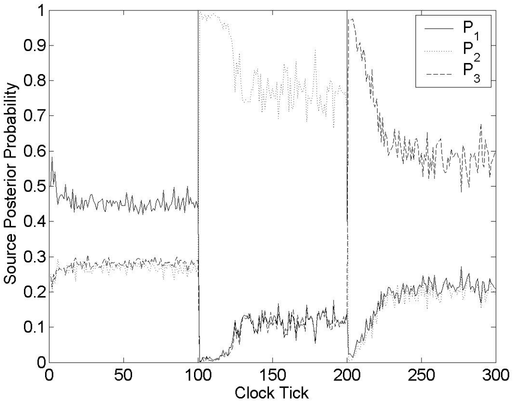


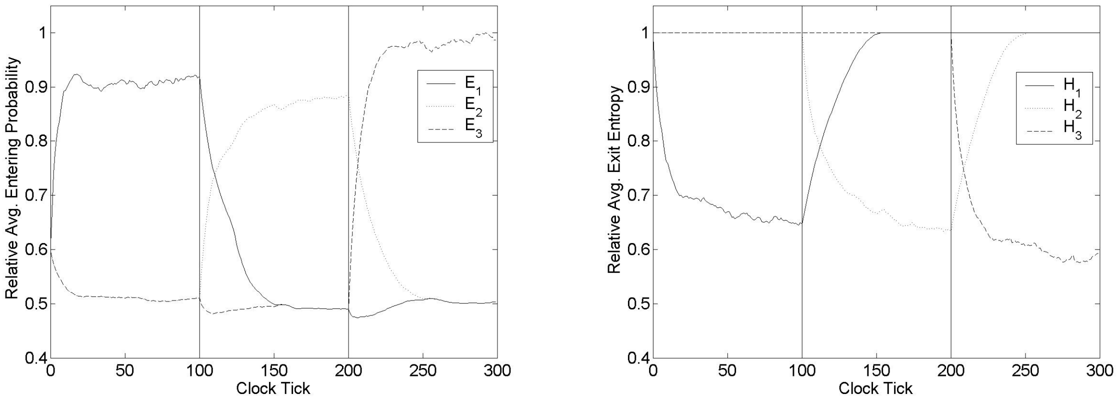
Fig. 7. Instantaneous source posterior probability of Eq. (54) as a function of clock tick. The solid line represents $P_1(k)$, the dotted is $P_2(k)$, and the dashed is $P_3(k)$. Vertical lines are drawn at each transition time (from source 1 to source 2, and from source 2 to source 3). We see that the estimator is able to correctly locate the suspect at each point in time, as indicated by the larger value of $P_s(k)$ for the correct s.

defined as

$$H_s(k) \propto \frac{1}{|\{i|\ i \text{ is exclusive to } s\}|} \sum_{i|\ i \text{ is exclusive to } s} \sum_j -\hat{\beta}_{ij}(k) \log \hat{\beta}_{ij}(k). \tag{56}$$

We interpret the value of $H_s(k)$ as follows: the smaller $H_s(k)$ is the more information we have about exit probabilities of sensors exclusive to s, this indicates more suspect messages are departing from sensors exclusive to s, and it is therefore more likely that the suspect is transmitting from s. Recall that a uniform initialization is used (all $\gamma_{ij} = 0.5$) so that all exit distributions are nominally uniform in the absence of suspect measurements.

We repeated this experiment 30 times, each time choosing independent random destinations, in order to average over the effect of randomly chosen destinations. The indicator quantities of Eqs. (54), (55), and (56) were recorded and averaged over these 30 trials. The averaged quantities are plotted versus clock tick in Figures 7 and 8. We see that the instantaneous source posterior probability pinpoints the correct suspect location during each 100-tick period. The average entering probability and exit entropy indicators also point to the correct source at the correct time. There is, however, some characteristic decay time in these quantities determined by the forgetting factor b. These simulations suggest our algorithms would be quite useful when applied to source tracking problems in the Internet.



A: Relative Avg. Entering Probability

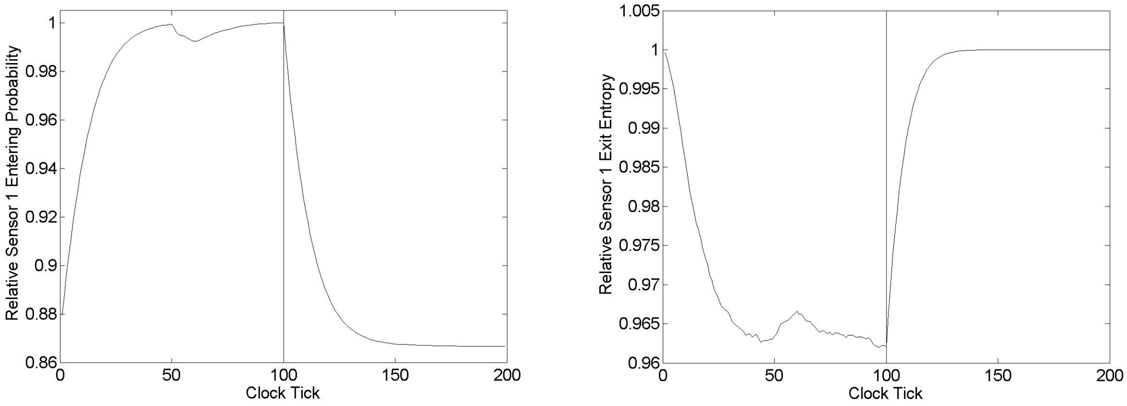
B: Relative Avg. Exit Entropy

Fig. 8. Plots of average entering probability in A and average exit distribution entropy in B as defined in Eqs. (55) and (56) respectively. The values are are normalized to the maximum in each plot. The solid line represents quantities associated with source 1, the dotted represents source 2, and the dashed represents source 3. Vertical lines are drawn at each transition time (from source 1 to source 2, and from source 2 to source 3). These indicators also point to the correct source location at the correct time as indicated by a rise in the appropriate entering probability $E_s(k)$, and a drop in the appropriate exit distribution entropy $H_s(k)$. At transition points, there is a decay of the previous extreme quantity with decay time determined by the forgetting factor b.

C. Sensor Failure

This simulation shows how one might use the routing parameter estimates in an interleaved probing paradigm such as that described in the online scheduling section (where probes occur during clock ticks when suspects are not observed). If suspects are constantly arriving, one might not have enough downtime for excessive probing. In this case, it is useful to monitor the evolution of the routing parameters for significant deviation from their nominal values. A large change in the network, such as failure of a sensor, would prompt such a deviation. It is then necessary to halt monitoring long enough to train the parameters to the new routing dynamics in the network.

We simulate such a sensor failure in this example and show how the failure is reflected in the routing parameter estimates. We suppose probes of the network from source 1 to random destinations are scheduled for 200 consecutive clock ticks with a forgetting factor of $a = 0.9$. Sensor 1 is positioned such that transmissions from source 1 to any destination always pass it. At time 100, sensor 1 fails– meaning that messages are still routed through it, but it does not activate in response to their passing. We computed average entering probability $E(k)$ and exit distribution entropy $H(k)$ for sensor 1. Similar



A: Relative Avg. Entering Probability

B: Relative Avg. Exit Entropy

Fig. 9. Plots of average entering probability in A and average exit distribution entropy in B as defined in Eq. (57) for sensor 1. The values are are normalized to the maximum in each plot. A vertical line is drawn at the point where sensor 1 fails. We see a drop in the entering probability and a rise in the exit entropy beginning at the failure point; of course there is a decay time determined by the forgetting factor a.

to the definitions in Eqs. (55) and (56), these are defined as

$$\begin{aligned} E(k) & \propto \textstyle\sum_d \sum_i \hat{\theta}^d_{i1}(k) \\ H(k) & \propto \textstyle\sum_d \sum_j -\hat{\theta}^d_{1j}(k) \log \hat{\theta}^d_{1j}(k). \end{aligned} \tag{57}$$

We averaged over the effect of random probe orders by repeating this experiment 30 times with independent random probes from source 1 and averaging the quantities in Eq. (57) over those 30 trials. The results are plotted in Figure 9.

We observe in Figure 9 a significant change in the nominal values of entering probability and exit entropy associated with sensor 1 after the failure point. In the alternative training scheme discussed above, one might set some allowed tolerance around the nominal. Once this is exceeded, we would have to schedule several probes to learn the new routing dynamics of the network.

D. Online Scheduling

We investigate the utility of the multiarmed bandit control algorithm of [17] applied to online probe scheduling in this example. Here, a single suspect transmission is observed initially. Then the online scheduling algorithm as described in Section VI is used to schedule probes for 200 clock ticks. At each probe, the reward (determined by the resulting change in entropy of the suspect endpoint posterior) is used in the recursions of Eq. (53) to update the distribution from which the next probe is drawn. Using a parameter $\delta = 0.1$, we are interested in how quickly the shaped component $w(k)$ of the distribution p is

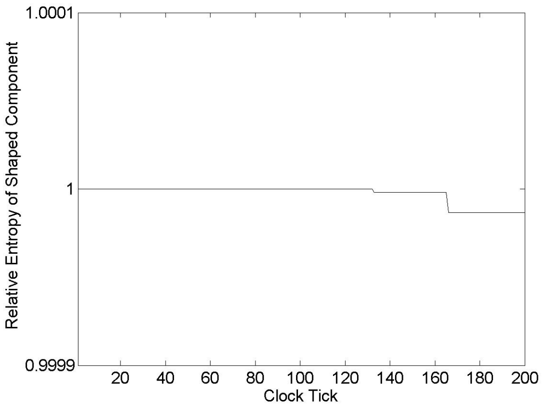


Fig. 10. Reward shaped distribution entropy $-\sum_i w_i(k) \log w_i(k)$ as a function of clock tick were $w(k)$ is defined in Eq. (53). The entropy is normalized by its initial value. We see that the entropy deviates very little from its maximum value in 200 clock ticks. This indicates that probes are essentially drawn from a uniform distribution throughout the simulation (since the other component of p in Eq. (53) is uniform).

able to concentrate on the best probing strategy for this particular suspect. We therefore plot the entropy of w (given by $-\sum_i w_i(k) \log w_i(k)$) as a function of clock tick to measure the concentration. This plot is shown in Figure 10.

We observe from Figure 10 that probes are essentially drawn from a uniform distribution through the entire simulation. The shaped component $w(k)$ concentrates slightly toward the end, as indicated by a small drop in entropy. One might suggest simply scaling the reward to speed up the process, however, the theory of [17] requires a reward between 0 and 1. This simulation seems to indicate that the proposed bandit scheduling algorithm requires a rather lengthy period of time to be effective. However, additional investigation of the utility of the algorithm is certainly justified for future work.

VIII. Summary and Future Work

We have presented online techniques for adaptively estimating the source and destination of a suspect transmission through a network based on the activation pattern of sensors placed on network components. In addition to a thorough theoretical development, we applied the new methods to several tracking experiments involving real Internet data obtained using `traceroute`. Speedy and accurate results were observed.

In the way of future work, one might analyze further the permutation clustering algorithm; in particular, issues related to selection of the parameter sequence $\tilde{S}_k$ and tree truncation level. We suggested a heuristic for choosing $\tilde{S}_k$ based upon the derived performance bound. Also, we assumed a given number of allowed leaves $L_{\max}$ before truncating the tree. One might consider linking these two ($\tilde{S}_k$ and $L_{\max}$) and solving some optimization problem to give a parameter sequence and truncation level that balances approximation accuracy and computational burden. The methods presented here could be applied with few changes to perform topology inference online, as an alternative to the offline approach of [7]. The probe scheduling method might also be extended to topology inference with a graph edit distance used to reward source/destination pairs that activate network segments similar to some prior structure of interest [34].

Acknowledgements

This work was supported by the National Science Foundation under ITR contract CCR-0325571. The first author is also grateful to Dan Ruan for valuable discussions related to this work and to Mike Rabbat for providing the `traceroute` data.

References

[1] CERT, "TCP SYN flooding and IP spoofing attacks," *CERT advisory CA-96.21*, Sept. 1996.

[2] ——, "Smurf IP denial-of-service attacks," *CERT advisory CA-98.01*, Jan. 1998.

[3] A. Yaar, A. Perrig, and D. Song, "StackPi: new packet marking and filtering mechanisms for DDoS and IP spoofing defense," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 10, pp. 1853–1863, Oct. 2006.

[4] H. Burch and B. Cheswick, "Tracing anonymous packets to their approximate source," *Proc. Usenix LISA*, pp. 319–327, Dec. 2000.

[5] J. Treichler, M. Larimore, S. Wood, and M. Rabbat, "Determining the topology of a telephone system using internally sensed network tomography," *Proc. of 11th Digital Signal Processing Workshop*, Aug. 2004.

[6] D. Justice and A. Hero, "Estimation of message source and destination from network intercepts," *IEEE Transactions on Information Forensics and Security*, vol. 1, no. 3, pp. 374–385, Sept. 2006.

[7] M. Rabbat, M. Figueiredo, and R. Nowak, "Network inference from co-occurrences," Department of Electrical and Computer Engineering, Univ. of Wisconsin, Madison, WI, Tech. Rep. ECE-06-2, April 2006.

[8] D. Titterington, "Recursive parameter estimation using incomplete data," *J. Royal Statistical Society, Series B*, vol. 46, no. 2, pp. 257–267, 1984.

[9] S. Chretien and A. Hero, "Kullback proximal algorithms for maximum-likelihood estimation," *IEEE Transactions on Information Theory*, vol. 46, no. 5, pp. 1800–1810, Aug 2000.

[10] Y. Matsuyama, "The alpha-em algorithms: surrogate likelihood maximization using alpha-logarithmic information measures," *IEEE Transactions on Information Theory*, vol. 49, no. 3, pp. 692–706, Mar 2003.

[11] H. Steck and T. Jaakkola, "On the Dirichlet prior and Bayesian regularization," Artificial Intelligence Lab, MIT, Cambridge, MA, Tech. Rep. 2002-014, Sept. 2002.

[12] B. Ripley, *Pattern Recognition and Neural Networks*. New York: Cambridge University Press, 1996.

[13] F. Chung, R. Graham, V. Hoggatt, and M. Kleiman, “The number of Baxter permutations,” *Journal of Combinatorial Theory, Ser. A*, vol. 24, pp. 382–394, 1978.

[14] V. Vatter, “Finitely labeled generating trees and restricted permutations,” *Journal of Symbolic Computation*, vol. 41, pp. 559–572, 2006.

[15] E. Barcucci, A. D. Lungo, E. Pergola, and R. Pinzani, “ECO: a methodology for the enumeration of combinatorial objects,” *Journal of Difference Equations and Applications*, vol. 5, pp. 435–490, 1999.

[16] G. Beylkin and M. Mohlenkamp, “Numerical operator calculus in higher dimensions,” *Proc. National Academy of Sciences*, vol. 99, no. 16, pp. 10 246–10 251, Aug. 2002.

[17] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. Schapire, “The nonstochastic multiarmed bandit problem,” *SIAM Journal on Computing*, vol. 32, no. 1, pp. 48–77, 2002.

[18] H. Robbins, “Some aspects of the sequential design of experiments,” *Bull. Amer. Math. Soc.*, vol. 55, pp. 527–535, 1952.

[19] K. Hintz, “A measure of the information gain attributable to cueing,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 21, no. 2, pp. 237–244, 1991.

[20] C. Kreucher, K. Kastella, and A. Hero, “Sensor management using an active sensing approach,” *Signal Processing*, vol. 85, no. 3, pp. 607–624, Mar. 2005.

[21] Y. Vardi, “Network tomography: estimating the source-destination traffic intensities from link data,” *J. Amer. Stat. Assoc.*, vol. 91, pp. 365–377, 1996.

[22] R. Caceres, N. Duffield, J. Horowitz, and D. Towsley, “Multicast-based inference of network-internal loss characteristics,” *IEEE Transactions on Information Theory*, vol. 45, no. 7, pp. 2462–2480, July 1999.

[23] N. Duffield, J. Horowitz, F. L. Presti, and D. Towsley, “Multicast topology inference from measured end-to-end loss,” *IEEE Transactions on Information Theory*, vol. 48, no. 1, pp. 26–45, Jan 2002.

[24] N. Duffield, “Network tomography of binary network performance characteristics,” *IEEE Transactions on Information Theory*, vol. 52, no. 12, pp. 5373–5388, Dec 2006.

[25] M. Newman, “The structure and function of complex networks,” *SIAM Review*, vol. 45, pp. 167–256, 2003.

[26] A. Tanenbaum, *Computer Networks*, 3rd ed. Upper Saddle River, NJ: Prentice Hall PTR, 1996.

[27] P. Kumar and P. Varaiya, *Stochastic Systems: Estimation, Identification, and Adaptive Control*. Englewood Cliffs, NJ: Prentice Hall, Inc., 1986.

[28] C.-J. Lin and R. Saigal, “A predictor corrector method for semidefinite linear programming,” Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI, Tech. Rep. TR95-20, Oct. 1995.

[29] A. Dempster, N. Laird, and D. Rubin, “Maximum likelihood from incomplete data via the EM algorithm,” *Journal of the Royal Statistical Society, Series B*, vol. 39, pp. 1–38, 1977.

[30] M. Figueiredo and R. Nowak, “An EM algorithm for wavelet-based image reconstruction,” *IEEE Transactions on Image Processing*, vol. 12, no. 8, pp. 906–916, Aug. 2003.

[31] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York: Cambridge University Press, 2004.

[32] H. Khalil, *Nonlinear Systems*, 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2002.

[33] C. Wu, “On the convergence properties of the EM algorithm,” *The Annals of Statistics*, vol. 11, no. 1, pp. 95–103, Mar. 1983.

[34] D. Justice and A. Hero, “A binary linear programming formulation of the graph edit distance,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 8, pp. 1200–1214, Aug. 2006.