Loss Inference in Wireless Sensor Networks Based on Data Aggregation

Gregory Hartl
hartl@eecg.toronto.edu

Baochun Li
bli@eecg.toronto.edu

Department of Electrical and Computer Engineering
University of Toronto
Toronto, Ontario M5S 3G4
Canada

ABSTRACT

In this paper, we consider the problem of inferring per node loss rates from passive end-to-end measurements in wireless sensor networks. Specifically, we consider the case of inferring loss rates during the aggregation of data from a collection of sensor nodes to a sink node. Previous work has studied the general problem of network inference, which considers the cases of inferring link-based metrics in wireline networks. We show how to adapt previous work on network inference so that loss rates in wireless sensor networks may be inferred as well. This includes (1) considering the per-node, instead of per-link, loss rates; and (2) taking into account the unique characteristics of wireless sensor networks. We formulate the problem as a Maximum-Likelihood Estimation (MLE) problem and show how it can be efficiently solved using the Expectation-Maximization (EM) algorithm. The results of the inference procedure may then be utilized in various ways to effectively streamline the data collection process. Finally, we validate our analysis through simulations.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations; C.4 [**Performance of Systems**]: Fault tolerance, Measurement; G.3 [**Probability and Statistics**]

General Terms

Measurement, Performance, Reliability

Keywords

Sensor Networks, Tomography, Data Aggregation

1. INTRODUCTION

Recent technological advances have made the development of low cost sensor nodes possible. This has allowed the deployment of large-scale networks of these sensor nodes to become feasible. The variety of possible uses of these networks has resulted in the area of sensor networks being actively researched. A survey of the current research and open challenges in wireless sensor networks is provided in [1]. Several of the characteristics of wireless sensor networks create challenging problems. Three of these characteristics are of particular interest:

(1) *Sensor nodes are prone to failures.* Due to the inherent instability and energy constraints of sensors, sensor nodes are prone to failures. It would thus be useful to determine which set of nodes or which areas within the network are experiencing high loss rates. Such information is potentially valuable to the design of fault tolerant protocols or monitoring mechanisms, so that the problem areas may be re-deployed, and critical data may be rerouted to avoid these areas (or nodes) suffering high loss rates. These are just a few of the many possible applications of per node loss rate information to streamline the data flow or enhance the reliability of large-scale sensor networks.

(2) *Sensor nodes use a broadcast communication paradigm and have stringent bandwidth constraints.* Therefore, one cannot rely on the use of active acknowledgments, which are not scalable or bandwidth-efficient, in the design of sensor network protocols. This renders the direct collection of loss rate data impossible in sensor networks. Furthermore, it would also be infeasible, due to limited bandwidth, for individual sensor nodes to collect and transmit loss rate data to a centralized location for processing.

(3) *Sensor nodes have limited resources.* Sensor nodes have limited power, memory and computational power. Therefore, any algorithms developed for wireless sensor networks must not rely on the assumption of unlimited resources, and must sparingly use the limited resources that do exist.

Motivated by the needs (fault-tolerance and reliability) and constraints (bandwidth and computational power) illustrated above, in this paper, we concentrate on the problem of efficiently determining per node loss rates in wireless sensor networks. Particularly, we attempt to efficiently determine per node loss rates based on the *data aggregation* communication paradigm. Due to the obvious need of centralized sensor data processing and monitoring, the paradigm of data aggregation, also referred to as *data fusion*, has been critical to the effective operation of sensor networks. Work in this area has been previously presented (e.g., [1, 11, 12, 13, 17]) and continues to be actively researched. In the process of data aggregation, a subset of nodes in the network attempt to forward the sensor data they have collected back to a *sink* node via a *reverse multicast tree*. The sink node is then able to communicate such collected data to the

IPSN'04, April26–27, 2004, Berkeley, California, USA.

user via a task manager node which is connected to the *sink* via an Internet or satellite connection.

More specifically, in the process of data aggregation, before a node sends its data to the next node in the path to the *sink*, it waits to receive data from all of its child nodes in the reverse multicast tree (or until a specified period of time has elapsed). The node then aggregates its own data with the data it has received from its child nodes, and forwards this aggregated data to the *sink* via the reverse multicast tree. Information about which nodes' data is present in the aggregated data must also be sent to the *sink*. Thus, data fusion saves communication overhead at the cost of additional computation and memory resources. Fig. 1 depicts a simple example of a sensor network using the data aggregation paradigm. To understand how data aggregation may conserve communication overhead, consider the simple example of the sink collecting data from nodes A, B and C. Node B sends it data, (B), destined for the *sink*, to node A. Node C similarly sends its data, (C), destined for the *sink* to node A. Node A then aggregates is own data, (A), with that of nodes B and C, and sends the fused data, (A,B,C) to the *sink*. With data aggregation, each node is only required to transmit once per data collection round. However, without data aggregation, node A would have to transmit three times per data collection round: once to send its own data to the *sink*, once to forward node B's data, and once to forward node C's data.

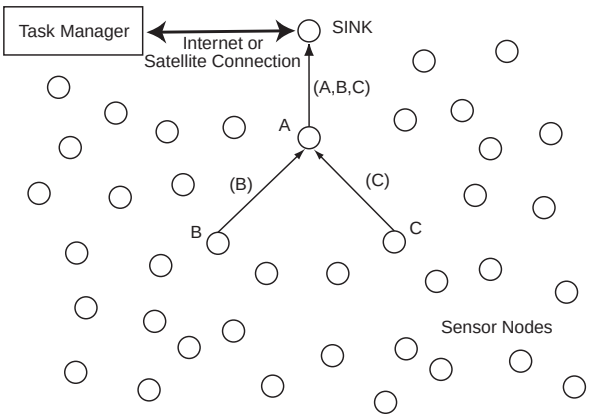


Figure 1: Data aggregation (fusion) in wireless sensor networks: an example

The field of network inference, also referred to as network tomography, involves estimating network performance parameters using measurements from only a small subset of nodes. That is, network tomography is the inference of internal network statistics using only end-to-end path-level measurements. Since the use of end-to-end or per-hop acknowledgments, or the collection and transmission of data loss reports by each node is infeasible in sensor networks, due to the limited resources of the nodes, the proposal of performing measurement-based computations only at end points is particularly appealing. In this paper, we show how the techniques of network inference may be used to infer internal node loss rates using only measurements taken at the *sink*. Our inference procedure does not require the use of acknowledgments or data loss reports to be maintained and transmitted by internal sensor nodes. Therefore, we are able to perform the inference of per node loss rates without incurring any additional overhead at internal nodes.

The original contributions of this paper are the following:

(1) *Adapting basic inference techniques for use in wireless sensor networks.* To the best of our knowledge, we believe that this work is the first to consider applying network tomography techniques in wireless sensor networks. Past research in network tomography has concentrated solely on wide-area wireline networks, and no previous work has yet attempted to address how the fundamental differences between wireline and wireless communications affect the problem and process of network inference. As a result of these differences, we have adapted network inference techniques to infer per node instead of per link statistics. In link based tomography, links with a common sender or receiver are treated independently. However, in wireless communications, all data being sent from or to the same node cannot be treated independently as it uses the same communication medium. Therefore, inferring node statistics requires considering all data flows into and out of each node instead of just the data flow between two nodes. For example, in Fig. 2 (a), a single loss rate is used to represent the probability that data sent from either node 2 or node 3 is successfully received by node 1. In contrast, in a wireline network, separate loss rates would be used to represent the probability that data is successfully received from each node. Also, although there is an apparent need for internal node loss rate data in sensor networks, we are unaware of any efficient solutions to this problem. This paper shows that network tomography can be used in wireless sensor networks and that it can be used to provide a good estimate of internal node loss rates with minimal overhead.

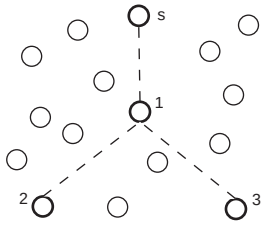
(2) *The use of tomography on reverse multicast trees.* We believe that there does not exist previous work that considers network tomography using reverse multicast trees for data aggregation. All network tomography research has dealt with the traditional multicast and unicast communication paradigms where data is sent from a single source to one or several receivers. In the reverse multicast case we consider, multiple sources are sending data to a single receiver. We show how network tomography can be efficiently adapted to work when the data aggregation communication paradigm employing reverse multicast trees is used.

The remainder of the paper is organized as follows. Section 2 presents the network model that will be used throughout the paper. It also discusses how we model the loss of data. Section 3 formulates the problem of inferring per node loss rates as a Maximum-Likelihood Estimation (MLE) problem and shows how this problem can be efficiently solved using the EM algorithm. Section 4 presents the results of several model simulations that were conducted. A discussion of the results is also provided. Section 5 discusses related work in network tomography and wireless sensor networks. Finally, Section 6 concludes the paper.

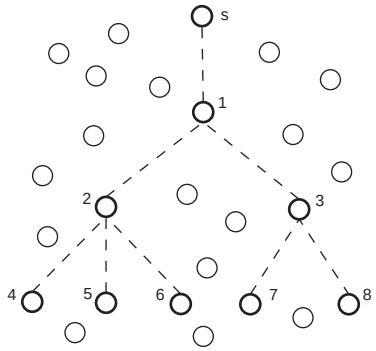
2. NETWORK AND LOSS MODELS

2.1 Reverse Multicast Trees

Let $\mathcal{T} = (V, E)$ denote the reverse multicast tree from all nodes to the *sink* for a given data collection task. $\mathcal{T}$ consists of a set of nodes, V, and a set of *links*, E. The set V contains scattered sensor nodes that are involved in a particular data collection task. Each node has the capability of collecting and routing data back to the *sink* via a wireless multi-hop infrastructureless routing protocol. The special node referred to as the *sink*, denoted by s, is responsible for collecting all of the data for a given operation and is assumed to have greater resources available than the other nodes. The set of links, E, contains ordered pairs $(i, j) \in V \times V$ such that node i sends its data, destined for the sink, directly to node j. An example of a simple 4 node reverse multicast tree is provided in Fig. 2(a). In this reverse multicast tree, nodes 1, 2 and 3 are attempting to route data to node s.



(a) A simple 4 node reverse multicast tree



(b) A 9 node reverse multicast tree

Figure 2: Reverse multicast trees: Two examples

Let $c(j)$ denote the set of children of node j. That is $c(j) = \{k \in V | (k,j) \in E\}$. For each node $j \in V \setminus \{s\}$, there is a unique parent node $k = \pi(j)$ such that $(j,k) \in E$. Consider a node j. Any node k on the path from j to s is referred to as an ancestor of j. Similarly, j is referred to as a descendant of k. If $j \neq k$ then j is a proper descendant of k, and k is proper ancestor of j. We may also recursively define the notion of a node j being a proper descendant of node k as follows. Let $\pi^n(j) = \pi(\pi^{n-1}(j))$. It then follows that j is a proper descendant of k if $\exists n > 0 : k = \pi^n(j)$. Denote the set of proper descendants of a node j by $d(j)$. That is $d(j) = \{k \in V | \exists n > 0 : j = \pi^n(k)\}$. Denote the set of descendants of a node j by $\overline{d}(j)$. That is, $\overline{d}(j) = d(j) \cup \{j\}$ Finally, denote the set of leaf nodes in the tree by $l(\mathcal{T})$. That is, $l(\mathcal{T}) = \{k \in V | c(k) = \emptyset\}$. For example, in the tree depicted in Fig. 2(b), the proper descendants of node 2 are nodes 4, 5 and 6 ($d(2) = \{4,5,6\}$). Similarly, the descendants of node 2 are nodes 2, 4, 5, and 6 ($\overline{d}(2) = \{2,4,5,6\}$). In this case, the leaf nodes are nodes 4, 5, 6, 7 and 8 ($l(\mathcal{T}) = \{4,5,6,7,8\}$).

It is assumed that $\mathcal{T}$ is known and remains static for the entire loss rate inference process. That is, the topology of the reverse multicast tree is known and remains relatively static. This is a reasonable assumption even if the topology is not known a priori since it is also possible to infer the topology of a network using end-to-end measurements. This process is presented in [4], [7], [9] and [15].

Further, it is reasonable to assume that nodes remain relatively stationary in a wireless sensor network (refer to [1]). The topology of the network need only remain static long enough to collect sufficient data for the loss rate inference process. Our simulations have shown that it is possible to obtain reasonable results using only a small number of data collection rounds. It is also likely that planned movement of a node will be the result of an order from the task manager which was sent through the *sink*. This means that the *sink* will likely be aware of when the topology is about to change. With this knowledge, the *sink* can ensure that the correctness of the loss inference process is maintained. The new topology can be considered as a separate reverse multicast tree with possible overlapping regions with the previous tree. Using the techniques presented in [2], it is possible to combine the loss rate data collected under the previous topology with the data that will be collected from the new topology. This combined data may then be used to infer the loss rates of each node that is present in at least one of the trees.

2.2 Data Loss Model

In wireline loss rate inference, the loss of data in the reverse multicast tree is modeled by a set of independent Bernoulli processes, one for each *link* $\in E$. For example, in Fig. 2(a), consider the case of node 3 sending data to node 1. In a wireline network, node 3's data may be lost due to a buffer overflow at node 1. However, in a wireless network there are several additional causes of data loss. Data may be lost due to interference at node 1, which may be the result of another node attempting to send data to node 1, node 1 sending data to its parent node, or communication between two entirely different nodes in the neighborhood. Since sensor nodes are prone to failures, data may also be lost due to the temporary failure of either node 1 or node 3.

Associate with every node $j \in V$ a probability $\alpha_j \in [0,1]$. Let α_j be the probability that data sent from node k, $k \in c(j) \cup \{j\}$, to node j is received successfully by node j. The *Loss Rate* of node j is therefore defined to be $\overline{\alpha_j} = 1 - \alpha_j$. Notice that this definition includes the possibility that a node j does not receive data from itself. This corresponds to the case where node j is experiencing a failure and is unable to even forward its own data to its parent node. This case is not considered in wireline network tomography but must be considered for wireless sensor network tomography since sensor nodes are prone to failures. Finally, let $\alpha = (\alpha_j)_{j \in V}$ denote the set of probabilities for all nodes.

The flow of data through a reverse multicast tree is modeled by a stochastic process $X = (X_{i,j})_{i \in V, j \in \overline{d}(i)}$, where each $X_{i,j} \in \{0,1\}$. $X_{i,j} = 1$ means that data sent from node j was successfully received by intermediate node i in the path $j \rightsquigarrow s$. Similarly, $X_{i,j} = 0$ means that data sent from node j did not successfully reach intermediate node i. Since a node i aggregates its data with that of all of its descendants before sending it to the *sink*, $X_{i,j} = 0$ implies that $X_{i,k} = 0, \forall k \in d(j)$. That is, if the *sink* does not receive data from a node, then it also will not receive data from all of the descendants of that node. For example, in Fig. 2(b) if the *sink* does not receive data from node 3 then it will also not receive data from nodes 7 and 8. If $X_{i,j} = 1$, then $\forall k \in c(j)$, $X_{i,k} = 1$ with probability α_j. Notice that since $X_{i,i}$ is defined, the loss rate of a node will also include data lost because node i failed and was unable to transmit its own data. This case is not considered in wireline network tomography where nodes are assumed to never fail and hence are always able to send their own data.

2.3 Data Collection For Loss Rate Inference

Consider the collection of data by the *sink* to be an experiment. Each round of data collection will be considered a trial within this experiment. The outcome of each trial will be a record of which nodes the *sink* received data from in that round. In terms of the stochastic process X defined in the previous subsection, each trial outcome is a random value $X_d = (X_{s,j})_{j \in V} \in \Omega = \{0,1\}^{N-1}$, where N is the number of nodes in the reverse multicast tree. Let P_α denote the distribution of the outcomes X_d for a given set of node probabilities $\alpha = (\alpha_j)_{j \in V}$. Finally, let $p(x;\alpha) = P(X_d = x)$ denote the probability mass function for a single outcome $x \in \Omega$.

Let the experiment consist of n data collection rounds. For each possible outcome $x \in \Omega$, let $n(x)$ be the number of data collection rounds for which x was the outcome. The probability of observing $x^1, \ldots, x^n$ in n data collection rounds is given by:

$$p(x^1, \ldots, x^n) = \prod_{m=1}^{n} p(x^m; \alpha) = \prod_{x \in \Omega} p(x; \alpha)^{n(x)} \quad (1)$$

where each $x^i = (x_j^i)_{j \in V}$ and each $x_j^i = X_{s,j}$ in data collection round i.

Section 3 will show how to use the record of which data was collected by the *sink* to infer the internal node loss rates.

3. LOSS RATE INFERENCE

In this section, we show how to formulate the problem of inferring internal node loss rates from end-point measurements as a Maximum-Likelihood Estimation (MLE) problem. We then show how the problem can be efficiently solved using the EM algorithm. Table 1 summarizes the important mathematical notations introduced in Section 2 as well as some which will be introduced in this section.

Table 1: List of Mathematical Notations

Parameter	Definition
$c(j)$	Set of node j's child nodes
$d(j)$	Set of node j's proper descendants
$\overline{d}(j)$	Set of node j's descendants
$l(\mathcal{T})$	Set of leaf nodes
Ω	Set of possible outcomes which the *sink* may observe
α_j	Probability of successfully sending data directly to node j
$\overline{\alpha_j}$	Loss rate of node j, i.e., $1 - \alpha_j$
N	number of nodes in the reverse multicast tree
n	number of data collection rounds
$n(x)$	number of data collection rounds outcome $x \in \Omega$ was observed by the *sink*
$n_{j,k}$	number of data collection rounds node j received data from descendant node k

Recall that the *sink* must keep a record of which nodes it received data from in each data collection round. It is clear that these are end point measurements and require no additional data transmission or collection by internal nodes. All that is required is additional space at the *sink* to store the record of data received. It is possible to store this data in a small one-dimensional array (or similar dynamic data structures), since it is only necessary to keep a count of how often each of the finite number of outcomes has occurred.

For example, for the network in Fig. 2(a), there are eight possible outcomes, *i.e.*, $|\Omega| = 2^3 = 8$, with each $x \in \Omega = \{0, 1\}^3$. Let the first bit of Ω represent whether the *sink* received data from node 1. Similarly, the second and third bits represent the record of the data received from nodes 2 and 3 respectively. However, only a small number of these outcomes can possibly occur. Clearly, the outcomes (011), (010) and (001) cannot occur since the *sink* cannot receive data from nodes 2 and 3 without also receiving data from node 1. In practice, dynamic storage, such as a linked list, is used to keep track of the number of times each outcome is observed during a data collection period. This limits the number of outcomes which must be stored to the number of data collection rounds, since at most n unique outcomes may occur in any n data collection rounds. The amount of storage required will likely be even less since each outcome is likely to occur more than once. Therefore, the amount of memory needed to store the frequency of the outcomes will be relatively small for most networks and data collection scenarios. Further, only the sink node is required to store this information, and its small size should not pose a burden on the expanded resources available to the *sink*. Therefore, the loss rate inference procedure presented in this section does not deplete the already scarce resources present in a sensor network.

As discussed in Section 2, $p(x; \alpha)$ is the probability mass function of a single outcome $x \in \Omega$. In our case, α is unknown and is the quantity we wish to estimate. If there are n data collection rounds, let $x_s^1, \ldots, x_s^n$ be the n outcomes that are observed at the *sink*. Maximum likelihood estimation consists of estimating α by $\hat{\alpha}$ such that $\hat{\alpha}$ maximizes the likelihood of observing the n outcomes. That is, we wish to choose $\hat{\alpha}$ such that $p(x_s^1, \ldots, x_s^n; \hat{\alpha})$ is maximized. If sufficient data can be observed, it may be possible to analytically solve the MLE problem and find an expression for the values we wish to infer which uses only observed data. However, in general, more sophisticated estimation techniques are required. Loss rate inference in wireless sensor networks is one such case which requires the use of an advanced estimation technique.

The EM Algorithm is a general algorithm that provides an efficient iterative procedure for performing maximum-likelihood estimation for situations in which missing data makes an analytical solution to the maximum-likelihood estimation overly complex or impossible. That is, the algorithm attempts to find an estimate, $\hat{\alpha}$, of α from the complete data X_c. The EM algorithm considers the observed data to be incomplete and an observable function of the complete data. Consider the reverse multicast tree depicted in Fig. 2(a). If the *sink* receives data from node 1, but does not receive data from either node 2 or 3, it is impossible for the *sink* to determine whether nodes 2 and 3 failed or if their data was dropped by node 1. Similarly, if the *sink* does not receive data from any of the nodes, this does not imply that all of the interior nodes failed. In fact, nodes 2 and 3 may have successfully transmitted their data to node 1 but the aggregated data of all three nodes was lost during the transmission from node 1 to the *sink*. Complete information of the data collection process would include knowledge of which data was received by each node for each data collection round. If complete information regarding how far data from each node traversed up the reverse multicast tree was available, then the loss rate inference problem would be trivial to solve.

The equation for calculating the estimated loss rate of a node, $\hat{\alpha}_j$, with complete information would simply be

$$\hat{\alpha}_j = \begin{cases} \sum_{k \in \overline{d}(j)} n_{j,k} \Big/ \sum_{k \in \overline{d}(j)} n & \text{if } j \text{ is a leaf node} \\ \sum_{k \in \overline{d}(j)} n_{j,k} \Big/ \sum_{k \in \overline{d}(j)} n_{r,k} & \text{otherwise} \end{cases} \quad (2)$$

where $n_{j,k}$ represents the number of data collection rounds for which node j received data from node k, n is the total number of data collection rounds, and r is the child node of j from which node j receives nodes k's data. We also define $n_{r,j} \doteq n$. That is, the success rate for a node is the ratio of the amount of data that was received from itself and its descendant nodes to the amount of data that it should have received from itself and its descendant nodes. If complete information regarding how far data from each node traversed up the reverse multicast tree was available, then the loss rate inference problem would be trivial to solve. However, obtaining complete information would require each sensor node

to record which descendants it received data from in each round. These records would then need to be transmitted to the *sink* for centralized processing to determine the loss rates. This in an unattractive solution in a wireless sensor network environment, as it wastes precious bandwidth and power resources. In practice, it is only feasible to determine which nodes the *sink* received data from in each round and then use the information to infer per node loss rates.

We follow the approach used in [2] and [5] and employ the EM algorithm by augmenting the incomplete observable data with unobservable, but complete data. Fig. 3 shows the data that is observable and unobservable by the sink at each node in the path $5 \rightsquigarrow s$. For example, $n_{1,5}$ is a count of how many rounds data from node 5 reached node 1. Node 1 would also have similar counts for all of its descendants. Complete knowledge of the observable and unobservable data makes it possible to determine where the data from a node was lost if it did not reach the *sink*. However, as discussed previously, it is only feasible to record data at the *sink*. That is, we only know $n_{s,j}; j \in V \setminus \{s\}$.

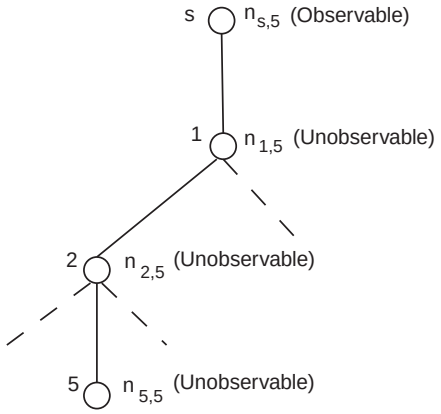


Figure 3: Observable and unobservable data in the path from node 5 to the sink

Since the complete counts, $n_{j,k}; k \in d(j) \cup \{j\}$, are not known, the EM algorithm iteratively augments the observed data with an estimate of the unobserved data. These estimated counts are then used to compute an estimate of the per node loss rates, $\hat{\alpha}^{(l)}$, where $\hat{\alpha}^{(l)}$ is the estimate for the loss rates after iteration l.

Let $X_s = (x_s^1, \ldots, x_s^n)$, with x_s^i representing the observed data at the *sink* for data collection round i, be the set of data received at the *sink*. The probability of the n independent data observations X_s is therefore

$$\begin{aligned} p(X_s;\alpha) &= \prod_{i=1}^{n} p(x_s^i;\alpha) \\ &= \prod_{x\in\Omega} p(x;\alpha)^{n(x)} \end{aligned} \tag{3}$$

The goal of loss rate inference is to estimate α by the maximizer of (3). That is,

$$\hat{\alpha} = \arg\max_{\alpha} p(X_s;\alpha) \tag{4}$$

However, it is more convenient to work with the log-likelihood function of the complete data. Let X_c be the set of complete data and x_c^i be the complete data for data collection round i. The log-likelihood equation then becomes

$$\mathcal{L}(X_c;\alpha) = \log p(X_c;\alpha) \tag{5}$$

The probability of observing $x_c^1, \ldots, x_c^n$ in n data collection rounds is therefore

$$\begin{aligned} p(x_c^1, \ldots, x_c^n;\alpha) &= \prod_{(j,k)_{k\in\overline{d}(j)\wedge j\notin l(\mathcal{T})}} \alpha_j^{n_{j,k}} \cdot \overline{\alpha}_j^{n_{r,k}-n_{j,k}} \\ &\quad \cdot \prod_{(j,j)_{j\in l(\mathcal{T})}} \alpha_j^{n_{j,j}} \cdot \overline{\alpha}_j^{n-n_{j,j}} \end{aligned} \tag{6}$$

where r is the child node from which node j receives node k's data. That is, r is the child of j in the path $k \rightsquigarrow j$, which implies that the data from node k follows the path $k \rightsquigarrow r \rightarrow j \rightsquigarrow s$.

Combining Equations (5) and (6) we obtain an expression for the log-likelihood function for loss rate inference in wireless sensor networks. To allow the log-likelihood to be easily maximized, we split the log-likelihood equation into two cases and consider them separately. In one case we consider only leaf nodes while in the other we consider only non-leaf nodes. This results in the following expressions for leaf nodes

$$\mathcal{L}(X_c;\alpha) = \sum_{j\in l(\mathcal{T})} \left[n_{j,j}\log\alpha_j + n\log\overline{\alpha}_j - n_{j,j}\log\overline{\alpha}_j\right] \tag{7}$$

and non-leaf nodes

$$\mathcal{L}(X_c;\alpha) = \sum_{(j,k)_{k\in\overline{d}(j)}} \left[n_{j,k}\log\alpha_j + n_{r,k}\log\overline{\alpha}_j - n_{j,k}\log\overline{\alpha}_j\right] \tag{8}$$

Maximizing Equations (7) and (8) yields the following two equations. The first is for the leaf node case

$$0 = \sum_{j\in l(\mathcal{T})} \left[\frac{n_{j,j}}{\alpha_j} - \frac{n}{\overline{\alpha}_j} + \frac{n_{j,j}}{\overline{\alpha}_j}\right] \tag{9}$$

while the second is for non leaf nodes

$$0 = \sum_{(j,k)_{k\in\overline{d}(j)\wedge j\notin l(\mathcal{T})}} \left[\frac{n_{j,k}}{\alpha_j} - \frac{n_{r,k}}{\overline{\alpha}_j} + \frac{n_{j,k}}{\overline{\alpha}_j}\right] \tag{10}$$

Solving Equations (9) and (10) for $\hat{\alpha}_j$ produces the expected expression

$$\hat{\alpha}_j = \begin{cases} n_{j,j}/n & j \in l(\mathcal{T}) \\ \sum_{k\in\overline{d}(j)} n_{j,k} \Big/ \sum_{k\in\overline{d}(j)} n_{r,k} & j \notin l(\mathcal{T}) \end{cases} \tag{11}$$

Using the expressions we have just developed we may now employ the EM algorithm to solve the loss rate inference problem. The EM Algorithm has four steps:

1) *Initialization*: Select initial node loss rates $\hat{\alpha}^{(0)}$.
2) *Expectation*: Estimate the unobservable $n_{j,k}$ counts using the conditional expectation under the probabilities $\hat{\alpha}^{(l)}$ and the observable data.
3) *Maximization*: Compute the new estimate $\hat{\alpha}^{(l+1)}$ using the estimated $n_{j,k}$ values.

4) *Iteration*: Iterate steps 2 and 3 until a termination criterion is met.

Steps 1 and 4 are implementation details that are not specific to wireless sensor network tomography. They will be explained in more detail in Section 4. Steps 2 and 3 for use in loss rate inference in wireless sensor networks will now be explained in more detail.

For iteration $l+1$, Step 2 requires estimating the $n_{j,k}$ counts using the observed data, X_s, and the probabilities, $\alpha^{(l)}$, from iteration l. Let $\hat{n}_{j,k}$ denote the estimated counts for the current iteration.

$$\begin{aligned}\hat{n}_{j,k} &= \mathrm{E}_{\hat{\alpha}^{(l)}}[n_{j,k}|X_s] \\ &= \sum_{i=1}^{n} \mathrm{P}_{\hat{\alpha}^{(l)}}[X_{j,k}=1|X_s=x_s^i] \\ &= \sum_{x\in\Omega} n(x)\cdot \mathrm{P}_{\hat{\alpha}^{(l)}}[X_{j,k}=1|X_s=x_s^i] \qquad (12)\end{aligned}$$

where

$$\mathrm{P}_{\hat{\alpha}^{(l)}}[X_{j,k}=1|X_s=x_s^i] = \begin{cases} 1 & \text{if } x_s^i \text{ indicates that the sink received data from node k} \\ \prod_{m\in k\rightsquigarrow j} \hat{\alpha}_m & \text{if } x_s^i \text{ indicates that the sink did not receive data from node k} \end{cases} \qquad (13)$$

For example, in the network depicted in Fig. 2(b), $\hat{n}_{1,5} = n(x)\cdot\hat{\alpha}_5\cdot\hat{\alpha}_2\cdot\hat{\alpha}_1$ if the outcome x being considered indicates that the *sink* did not receive data from node 5.

Step 3 involves finding the maximizer of Equations (7) and (8). This can be accomplished by substituting the estimated counts, $\hat{n}_{j,k}$, from Step 2 into Equation (11). The values computed during this step become $\hat{\alpha}^{(l+1)}$. Therefore, the equation used by Step 3 of the EM Algorithm becomes

$$\hat{\alpha}_j = \begin{cases} \hat{n}_{j,j}/n & j\in l(\mathcal{T}) \\ \sum_{k\in\overline{d}(j)} \hat{n}_{j,k} \Big/ \sum_{k\in\overline{d}(j)} \hat{n}_{r,k} & j\notin l(\mathcal{T}) \end{cases} \qquad (14)$$

The EM algorithm has several properties that make it especially beneficial in wireless sensor network tomography. It is numerically stable and reliably converges to a local maximizer from an arbitrary starting point $\hat{\alpha}^{(0)}$ in almost all cases. The EM algorithm is also easy to implement and requires little storage space and computational power. The reader is referred to [14] for a more complete discussion of the EM algorithm and its properties. Therefore, it should be possible to efficiently implement a loss rate inference procedure using the EM algorithm in sensor nodes with limited resources.

4. SIMULATIONS

In this section we attempt to verify the validity and assess the accuracy of the loss rate inference procedure presented in Section 3. We accomplish this by performing several model simulation experiments. The experiments consider several loss scenarios for two different networks.

A program was created to randomly generate sensor network topologies and to simulate the flow of data through these networks. MATLAB was then used to perform the loss rate inference using the previously generated topologies and data flow records. To create a random topology, The desired number of nodes were randomly placed within a 10 unit by 10 unit square region, with the sink being placed in the center of the region. An example of a randomly generated topology is provided in Fig. 7. A reverse multicast tree for the random topology was then generated. Next, the flow of data through the network was simulated, including a record of the outcomes that the *sink* would observe. For each data collection round, whether a node successfully received data sent to it by its child nodes was determined randomly but with a specified intended loss rate for each node. That is, as the number of data collection rounds increases the actual loss rate of each node should converge to the intended loss rate. The actual loss rate for each node was also recorded.

Two networks were used in the simulations. One consisted of 115 nodes while the other contained 9 nodes. Fig. 2(b) shows the topology of the 9 node network. An intended success rate of 0.99 (i.e. a loss rate of 1%) was chosen for all normally operating nodes in the 115 node network. An intended success rate of 0.98 was used for normally operating nodes in the 9 nodes network. Each simulation consisted of 500 data collection rounds. Once all of the data was collected, the per node loss rates were inferred using the results of Section 3.

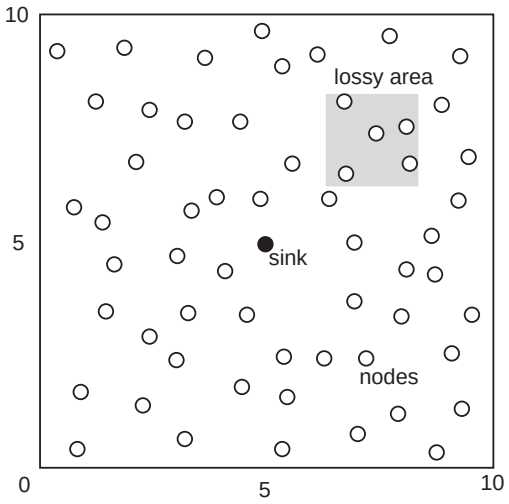


Figure 7: A random sensor network for simulations: an example

As discussed in Section 3, the *initiation* and *iteration* steps of the EM Algorithm are implementation specific details. In all of the simulations $\alpha^{(0)}$ was chosen such that the initial loss rate of each node was 0.85. The choice of initial loss rates does not appear to affect the inferred loss rates to which the algorithm converges. This is a property of the EM algorithm which was also observed by [2]. Finally, steps 2 and 3 were iterated until the inferred loss rate of each node changed by less than 0.0001 between consecutive iterations.

Four different loss scenarios were used to test the robustness of the inference procedure. These four cases cover all of the possible data loss scenarios that may occur in a real sensor network. The scenarios were:

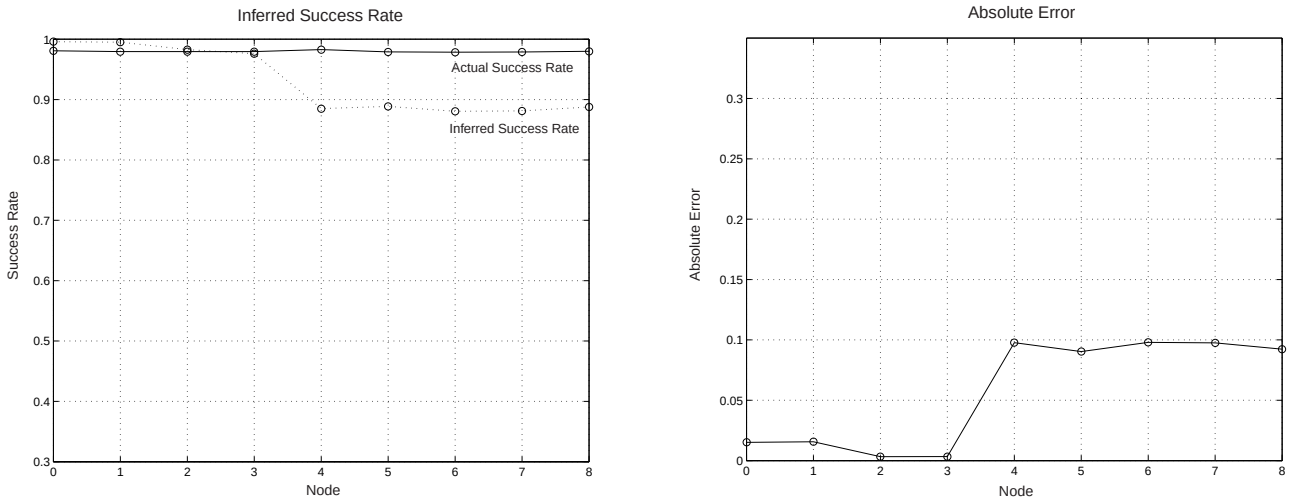


Figure 4: Simulation results: Equal losses on all nodes

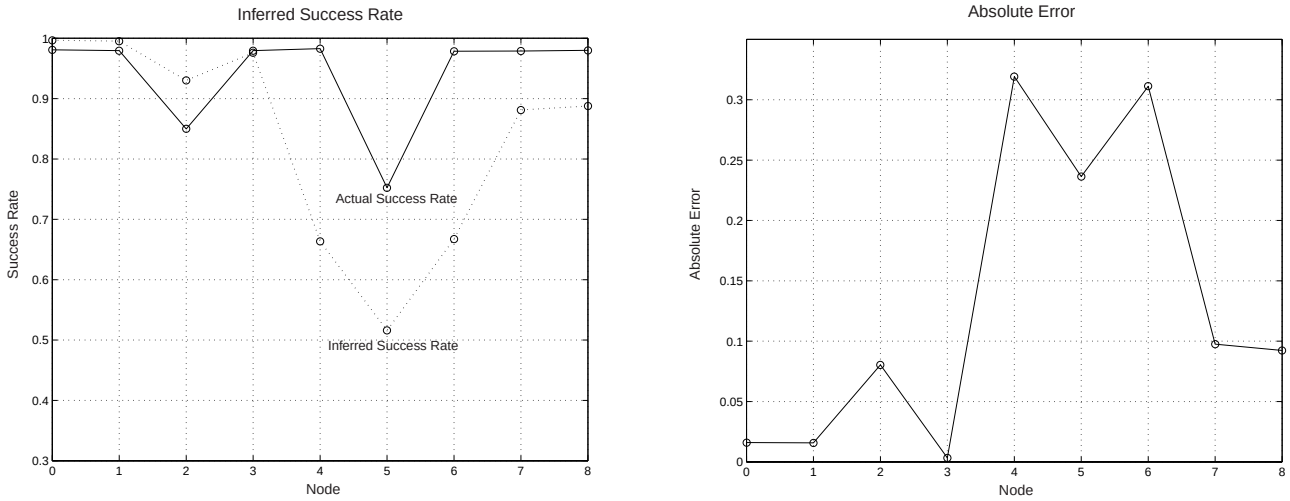


Figure 5: Simulation results: Cascaded losses (Heavy losses at nodes 2 and 5)

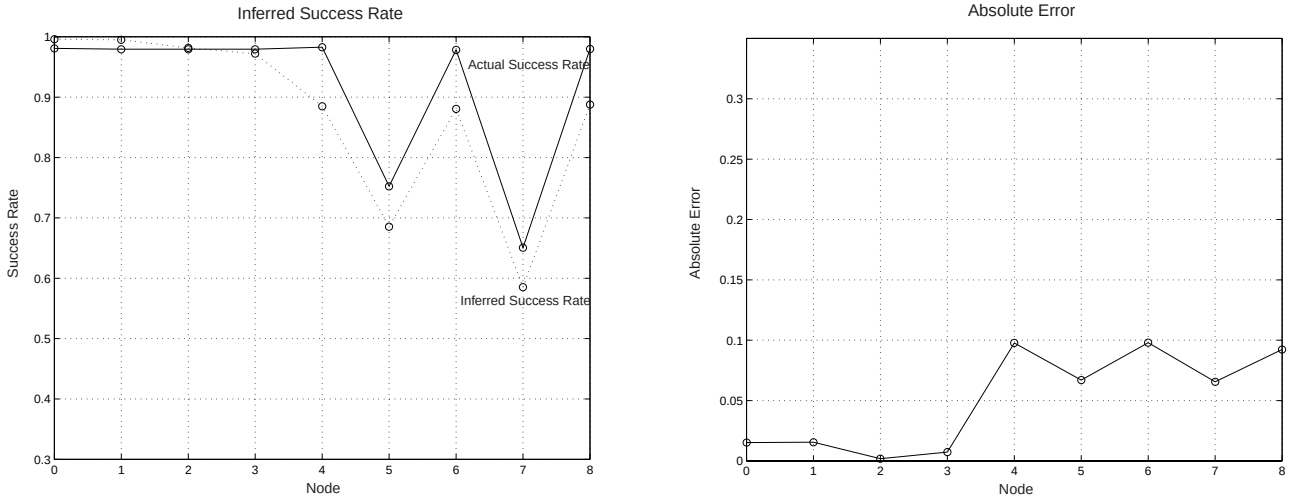


Figure 6: Simulation results: Heavy losses in separate branches (Heavy losses at nodes 5 and 7)

Table 2: Summary of Mean and Maximum Absolute Errors: 115 node network

Loss Scenario	Mean Absolute Error	Maximum Absolute Error
Equal losses	0.0617	0.2220
Lossy area	0.0724	0.4640
Cascaded losses	0.0625	0.2220
Heavy losses in separate branches	0.1427	0.4200

1. Equal losses throughout the network
2. Lossy area. i.e. Heavy losses at nodes located in the same physical region.
3. Heavy losses in separate branches of the tree
4. Cascaded losses. i.e., Heavy losses at nodes on the same path to the sink.

To simulate the equal losses throughout the network scenario, the intended success rate of each node was set to be that of a normally operating node. This scenario corresponds to the normal operation of the network. To simulate the lossy area scenario, all nodes within a predefined 4 square unit region had their intended success rates set to 0.75. The lossy area scenario was only simulated for the 115 nodes network since only one node would fall within the lossy area for the 9 node network. In the 115 node network, four nodes were located within the lossy area. This scenario corresponds to the case when a physical obstruction, misbehaving node or other interference is causing nodes in a certain geographic region to perform poorly.

The next two scenarios simulate the independent failure of nodes. The cascaded losses scenario was simulated by randomly selecting two nodes from the same branch of the reverse multicast tree. The intended success rates of these nodes were then set to 0.75. For the 9 node network, the intended success rates of nodes 2 and 5 were set to be 0.85 and 0.75 respectively. Finally, the heavy losses in separate branches of the tree scenario was simulated by randomly selecting several nodes from different branches of the reverse multicast tree. For the 115 node network, the intended success rates of the selected nodes were then set to 0.75. In the 9 node network, the intended success rates of nodes 5 and 7 were set to be 0.75 and 0.65 respectively.

A plot of the inferred and actual success rates as well as the absolute error for each node for the equal losses scenario is provided in Fig. 4. In this scenario the average absolute error was only 0.057. The leaf nodes all experience an error of approximately 0.10 while the rest of the nodes experience very small errors. Although the leaf nodes all have inferred success rates that are lower than their actual values, the inferred success rates are still very high and would not lead to falsely concluding that any of these nodes are experiencing heavy losses.

The results of the cascaded losses scenario for the 9 node network are presented in Fig. 5. In this case, the average absolute error was 0.1302. The error was most significant for nodes 4 through 6 as some of the losses that should have been attributed to node 2 were instead attributed evenly amongst node 2's child nodes. This can be seen in the plot as the inferred success rate of node 2 is higher than its actual success rate while the success rates of the child nodes are lower than their actual rates. However, since each of the child nodes experiences a similar absolute error as a result of the overestimation of node 2's success rate, it is still possible to determine that node 5 is in fact experiencing the heaviest losses.

Fig. 6 contains a plot of the actual and inferred success rates for each node in the 9 node network for the heavy losses in separate branches scenario. Fig. 6 also contains a plot of the absolute error for each node. We were able to successfully infer the success rate of each node with an average absolute error of only 0.0512. More importantly, it is clear from the plot of inferred success rates that our inference procedure was able to determine which nodes were experiencing heavy losses.

In the lossy area scenario, we were able to infer which nodes were in the lossy area. The descendants of the nodes in the lossy area had inferred success rates that were lower than their actual values. This is because some of the losses that should have been attributed to the nodes in the lossy area were instead being attributed to their descendants. However, it is still possible to determine the general area which is experiencing high loss rates.

Table 2 provides the mean and maximum absolute error of the inferred success rates for all of the loss scenarios for the 115 node network. These results show that our loss rate inference procedure scales well. However, the accuracy of the inferred values does decrease as the network size increases. A possible solution to this problem is to designate some of the nodes in the network as *inference nodes*. These nodes would then act as sink nodes for their non-inference node descendants in the inference process. They would perform the loss rate inference for all of their descendants and then forward the results to the actual *sink*. This should allow our inference procedure to scale to even larger networks, at the expense of additional overhead, without sacrificing accuracy.

5. RELATED WORK

There has been much research in the area of network tomography for wide-area wireline networks. A summary of this research is provided by [4]. Current research has considered the problem of using both multicast (e.g., [3], [7], [9], [15]) and unicast (e.g., [5], [10] and [16]) probes to infer internal network characteristics. The type of probe used should reflect the type of traffic for which measurements are required. That is, if one would like information regarding the performance of unicast packet transmission in the network, then unicast probes should be used.

The basic concept behind tomography is that there is correlation in the losses (and delays) experienced by the data that is intended for each receiver. For example, consider the network depicted in Fig. 2(b). If the *sink* sends a multicast packet destined for nodes 4 and 5, then the losses and delays experienced on the common portions of the multicast tree, links (s,1) and (1,2), are expected to be the same since they are in fact the same packet. That is, if the data is lost on the link between nodes 1 and 2 then neither node 4 nor node 5 will receive the multicast data. Also, the difference in the delay measured by nodes 4 and 5 will only be the difference between the delays on links $(2, 4)$ and $(2, 5)$.

In all cases, the network characteristics being inferred are for links as opposed to the node based characteristics we wish to infer in wireless sensor networks. The most common characteristics to infer are loss rates (e.g., [2], [3], [5], [10]), delays ([8]), delay

distributions ([6]), and topology (e.g., [7], [9]). Previous work has shown that inference techniques used to infer one characteristic can often be easily adapted to infer other characteristics. We therefore only consider the inference of per node loss rates in this paper. Our results should also be applicable to the inference of additional characteristics such as per node delay.

Most of the current network inference research involves the use of actively sending probe packets into the network. However, the use of passive network inference has been considered by [16]. The use of passive tomography techniques is especially important for sensor networks. The limited resources of sensor nodes makes the sending of large numbers of tomography probe packets into the network unappealing. It is likely that the tomography process would deplete more resources than the use of its results would be able to save.

There has also been significant research in the dissemination and propagation of data in wireless sensor networks. The reader is referred to [11], [12], [13] and [17] for discussions of some of the dissemination and propagation techniques proposed.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we propose to apply wireline network tomography techniques for use in wireless sensor networks, especially when using the data aggregation communication paradigm with reverse multicast trees. To the best of our knowledge, this is the first work to consider applying network tomography techniques in wireless sensor networks, targeting per-node loss rate inference rather than a per-link analysis. We show our detailed proposal of formulating the problem of loss inference as a Maximum-Likelihood Estimation problem, and performing the inference with the efficient EM algorithm commonly adopted in a large body of tomography literature. Via simulations, we validate our claims that per-node loss rates in the reverse multicast tree may be inferred with accuracy by the sink node, without any of the internal nodes incurring the additional overhead of active probes, per-hop acknowledgments, or loss reports. Our proposal minimizes the overhead (in fact, there is no overhead of actively injected protocol traffic), and efficiently solves the problem of loss inference. The results may be used in a variety of applications, such as when re-routing around problem areas that suffer from high loss rates, or when designing robust fault-tolerant protocols. In our future work, we intend to refine the inference process and experiment with more loss scenarios in even larger sensor networks. We would also like to extend our proposed algorithm to the inference of per-node transmission latency.

7. REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A Survey On Sensor Networks. *IEEE Communications Magazine*, 40(8):102–114, August 2002.

[2] T. Bu, N. Duffield, F. L. Presti, and D. Towsley. Network Tomography on General Topologies. In *Proceedings of ACM Sigmetrics 2002*.

[3] R. Caceres, N. Duffield, J. Horowitz, and D. Towsley. Multicast Based Inference of Network Internal Loss Characteristics. *IEEE Trans. on Information Theory*, 45:2462–2480, 1999.

[4] M. Coates, A. H. III, R. Nowak, and B. Yu. Internet Tomography. *IEEE Signal Processing Magazine*, 19(3):47–65, May 2002.

[5] M. Coates and R. Nowak. Network Loss Inference Using Unicast End-to-End Measurement. In *ITC Seminar on IP Traffic, Measurement and Modelling*, September 2000.

[6] M. Coates and R. Nowak. Network Delay Distribution Inference from End-to-end Unicast Measurement. In *Proceedings 2001 IEEE Int. Conf. Acoust., Speech, and Signal Processing*, May 2001.

[7] N. Duffield, J. Horowitz, F. L. Presti, and D. Towsley. Multicast Topology Inference From Measured End-to-End Measurements. In *ITC Seminar on IP Traffic, Measurement and Modelling*, September 2000.

[8] N. Duffield, J. Horowitz, F. L. Presti, and D. Towsley. Network Delay Tomography from End-to-end Unicast Measurements. In *Proceeding of the 2001 Internation Workshop on Digital Communications*, September 2001.

[9] N. Duffield, J. Horowitz, F. L. Presti, and D. Towsley. Multicast Topology Inference From Measured End-to-End Loss. *IEEE Trans. on Information Theory*, 48:26–45, 2002.

[10] N. Duffield, F. L. Presti, V. Paxson, and D. Towsley. Inferring Link Loss Using Striped Unicast Probes. In *Proceedings of IEEE INFOCOM 2001*.

[11] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-Efficient Communication Protocol for Wireless Microsensor Networks. In *Proceedings IEEE Hawaii International Conf. on System Sciences*, pages 1–10, 2000.

[12] W. R. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive Protocols for Information Dissemination in Wireless Sensor Networks. In *Proceedings ACM MobiCom '99*, pages 174–185, 1999.

[13] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva. Directed Diffusion for Wireless Sensor Networking. *IEEE Trans. on Networking*, 11(1):2–16, February 2003.

[14] G. J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. John Wiley and Sons, Inc., New York, 1997.

[15] S. Ratnasamy and S. McCanne. Inference of Multicast Routing Trees and Bottleneck Bandwidths using End-to-End Measurements. In *Proceedings IEEE INFOCOM 1999*.

[16] Y. Tsang, M. Coates, and R. Nowak. Passive Network Tomography Using EM Algorithms. In *Proceedings 2001 IEEE Int. Conf. Acoust., Speech, and Signal Processing*, volume 3, pages 1469–1472, May 2001.

[17] F. Ye, H. Luo, J. Cheng, S. Lu, and L. Zhang. A Two-tier Data Dissemination Model for Large-Scale Wireless Sensor Networks. In *Proceedings of ACM Mobicom 2002*, pages 148–159, September 2002.