Routing Topology Inference for Wireless Sensor Networks

Yao Liang
yliang@cs.iupui.edu

Rui Liu
liurui@iupui.edu

Department of Computer and Information Science
Indiana University-Purdue University Indianapolis

ABSTRACT

We consider an important problem of wireless sensor network (WSN) routing topology inference/tomography from indirect measurements observed at the data sink. Previous studies on WSN topology tomography are restricted to static routing tree estimation, which is unrealistic in real-world WSN time-varying routing due to wireless channel dynamics. We study general WSN routing topology inference where routing structure is dynamic. We formulate the problem as a novel compressed sensing problem. We then devise a suite of decoding algorithms to recover the routing path of each aggregated measurement. Our approach is tested and evaluated though simulations with favorable results. WSN routing topology inference capability is essential for routing improvement, topology control, anomaly detection and load balance to enable effective network management and optimized operations of deployed WSNs.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Wireless communication

General Terms

Algorithms

Keywords

Wireless sensor networks, routing topology, network inference, compressed sensing, recovery algorithms.

1. INTRODUCTION

Wireless sensor networks (WSNs) have been fundamentally changing today's practice of numerous scientific and engineering endeavors, including studies of environmental sciences, ecosystems, natural hazards, accurate agriculture, and smart building, by enabling continuous monitoring and sensing physical variables of interest at unprecedented high spatial densities and long time durations [1, 2, 3]. Network inference - also known as network tomography or inferential network monitoring - studies how to efficiently reconstruct the network structure (e.g., routing topology) and important dynamics (e.g., link performance, load balance) of large-scale networks from *indirect* measurements when direct measurements are either unavailable or impractical to collect due to resource constraints (e.g., [4, 5, 6, 7, 8, 9, 10, 11]). As WSNs are growing rapidly in both size and complexity, it becomes increasingly critical to monitor the WSN structure and dynamics and identify any internal problems using *indirect* measurements obtained at the WSN sink(s). Such network inference capability is essential for routing improvement, topology control, anomaly detection and load balance, enabling effective management and optimized operations for deployed WSNs consisting of a large number of unattended wireless sensor nodes. Compared to wireline network inference, WSN inference has its unique challenges because of the severe resource limitations (e.g., battery power, bandwidth, memory size, and CPU capacity) of tiny sensor nodes. Most environmental and natural hazard monitoring WSNs are deployed in harsh environments such as mountainous areas, forests, volcano areas, and oceans, and thus the replacement of battery for sensor nodes is usually impossible. Most existing research on WSN tomography has concentrated on link loss and delay monitoring [12, 13, 14, 15, 16], with the assumption that routing topology is given a priori. In contrast, studies on WSN topology tomography are few and restricted to *static* routing tree estimation [17, 18], which is unrealistic and problematic in real-world WSN deployments/applications where routing topology is *time-varying* due to *wireless channel dynamics* such as fading and interference. This lack of investigation into *realistic* and *dynamic* WSN routing topology inference/tomography may significantly undermine the foundation and values of the works on WSN loss/delay tomography.

In this paper, we study the general WSN routing topology inference for dynamic routing structure which is random and time-varying. Inspired by the recent breakthrough of compressed sensing (CS) theory [19, 20, 21], we formulate the problem as a novel CS problem. We then devise a suite of decoding algorithms to recover the routing path of each aggregated measurement at the sink. To our knowledge, no reported research on network inference addresses the challenge of time-varying routing topology structure. This work intends to bridge this important gap.

2. ROUTING TOPOLOGY MODEL

We consider *acyclic* routing topology for WSN data collection, which can be modeled by a directed acyclic graph $G = (V, E)$, where V is a set of n nodes (the sink s and $n-1$ sensor nodes), i.e., its cardinality $|V| = n$, and E is a set of edges. The sink s is the particular node where sensed data from individual sensor nodes should be periodically gathered. Sensor nodes (i.e., motes) are battery-operated while the sink is assumed not to be power-limited. A directed edge $e_{(u,v)}$ is an ordered pair $(u, v) \in V \times V$ representing the wireless communication link from node u to node v. Each edge $e_{(u,v)}$ is associated with a unique label $l_{(u,v)}$, given by a labeling function $L : E \rightarrow \mathbf{N}$, where $\mathbf{N}$ denotes the set of positive integers. For each sensor node i, let $p_i = \{e_{i,t_1}, e_{t_1,t_2}, \cdots, e_{t_j,s}\}$ denote its routing path through relay sensor nodes $t_1, t_2, \cdots, t_j$, to the sink node s. Let y_i denote an indirect path measurement of path p_i at the sink. Then, measurement vector $Y = \{y_1, y_2, \cdots, y_M\}^T$, where $M = n - 1$, denotes a complete set of path measurements for the WSN.

For a *static* routing, the routing topology can be represented as a directed spanning tree of WSN's complete directed connectivity graph, with the sink being the root of the routing tree. Let $T = (V, E^0)$ denote this spanning tree structure, where E^0 is the edge (i.e., link) set and $|E^0| = n - 1$. Clearly T is a special case of the routing topology model G given here, i.e., $T \subseteq G$.

In this paper, we assume WSN routing is *dynamic* in a cycle of data or measurements collection due to wireless link dynamics. A general acyclic dynamic routing topology G can be decomposed into a (directed) spanning tree and some additionally attached edge(s). These additionally attached directed edges are referred to as 'shortcuts' in the paper, and in this sense, a G can also be referred to as a (directed) Augmented 'Tree' (A-Tree). Let E^+ denote the set of shortcuts, then we have $E = E^0 \cup E^+$, with $|E| = |E^0| + |E^+| = n - 1 + |E^+|$. An A-Tree structure has the following properties due to shortcut(s): (1) A node may have more than one parent node; and (2) A node may have multiple paths to the sink.

Example 1 An illustration of an A-Tree routing structure with the sink node 0 is shown in Figure 1, where the presence of dotted link $e_{2,0}$ is due to WSN link dynamics during a data collection cycle. The A-Tree can be decomposed into a baseline spanning tree with a set of additionally shortcut(s) $e_{2,0}$.

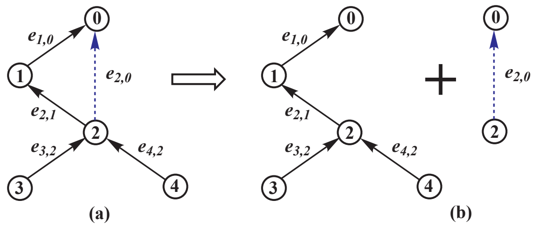

Figure 1: An illustration of an A-Tree.

3. PROBLEM FORMULATION

Consider to reconstruct the dynamic routing topology structure that evolves along time during a cycle of WSN data/measurements collection. Given the general routing topology model above, our objective is to recover the routing path p_i for each indirect path measurement packet y_i originated from sensor node i and received at the sink. When a complete set of M ($M = n - 1$) path measurements originated from individual $n - 1$ sensor nodes is received in one collection cycle, the entire routing topology $G(V, E)$ will be exactly reconstructed with $E = p_1 \cup p_2 \cup \cdots p_M$.

3.1 Formulation from CS Perspective

We formulate the problem of WSN routing topology inference from a CS perspective. The standard CS framework can be represented as

$$Y = \Phi X,$$

where X is an $N \times 1$ sparse discrete signal vector, Φ is an $M \times N$ measurement matrix and Y is the $M \times 1$ measurement vector. The CS theory allows, under certain conditions, to recover X from Y where $M \ll N$, as long as the signal X is sparse. This can be achieved (with probability close to one) by solving the following optimization:

$$\hat{X} = argmin_x \|X\|_p \text{ subject to } Y = \Phi X, \qquad (1)$$

where $\|X\|_p$ ($p = 0, 1$) denotes l_p-norm of X.

To formulate the WSN routing topology inference problem, we introduce a new concept of so-called *Base Topology* of a WSN, denoted by $G^*(V, E^*)$, where $|V| = n$. Given an arbitrary WSN routing topology model defined in Section 2 by $G_i(V, E_i)$, then $G^*(V, E^*)$ is simply defined by $\forall i \; E^* \supset E_i$. That is, the base topology of a WSN is the superset of all possible routing topologies of the WSN. For WSN upstream routing, outgoing links from the sink are excluded, and thus the total number of all *possible* directed wireless links (considering asymmetry wireless channel property) for the upstream base topology G^* should be $|E^*| = n(n-1) - (n-1) = (n-1)^2$.

Our key observation is that the actually used wireless links in a WSN routing topology G_i for a short-time interval (e.g., a measurement/data collection cycle) would be much *fewer* than the number of total potential choices in the upstream base topology G^*, i.e., $|E_i| \ll |E^*|$, because good wireless links are likely to be reused whenever possible to reduce any re-routing overhead and unnecessary retransmissions for energy conservation and reliable data transfer in the WSN. Let $N = |E^*| = (n-1)^2$. In our formulation, let X be a link (labeling value) vector of N dimension, which shall be *sparse*. Assume that in a given measurement/data collection cycle of a WSN of n nodes, the sink receives a complete set of path measurements, denoted as an $M \times 1$ vector $Y = \{y_1, y_2, \cdots, y_M\}^T$ where $M = n-1$. Thus the measurement matrix $\Phi = \{\varphi_{i,j}\}$ $(1 \le i \le M, 1 \le j \le N)$ can represent a routing matrix in the WSN where the ith row represents the ith path while the jth column represents the jth link, whose elements $\varphi_{i,j}$ are defined as

$$\varphi_{i,j} = \begin{cases} 1, & \text{the } i\text{th path traverse over the } j\text{th link;} \\ 0, & \text{otherwise.} \end{cases}$$

Note that $|E| = n - 1 + |E^+|$, where $|E^+|$ is the number of shortcuts in G (i.e., A-Tree), as discussed in Section 2. Since X is sparse, $|E_i^+|$ should be a relatively small number (e.g., $|E^+| < n$), a reasonable assumption in practical WSN for one cycle of data/measurements collection. Thus, we now have an innovative way to formulate the topology inference problem of dynamic routing from a CS perspective: *given a measurement vector Y at the WSN sink, recover the X and measurement matrix Φ, so that*

$$\hat{X} = argmin\|X\|_0 \textit{ subject to } Y = \hat{\Phi} X, \qquad (2)$$

where l_0-norm $\|X\|_0$ is the number of nonzero elements in the vector X, that is $\|X\|_0 = |E|$.

We point out that, unlike the traditional CS formulation (1), where the measurement matrix Φ is known a priori whether randomly or deterministically generated, the Φ in our problem formulation (2) is completely unknown, due to the un-deterministic real-world communication environment for that collection cycle. On the other hand, in contrast to the traditional CS formulation (1), we know each potential link's value a priori by the labeling function as described in Section 2. So, our problem formulation (2) is to recover Φ and therefore to reconstruct the sparseness pattern of the X, given a Y.

Example 2 Given a WSN of 5 nodes as shown in Figure 1(a), in which four paths originated from each sensor node are $p_1 = \{e_{1,0}\}$, $p_2 = \{e_{2,1}, e_{1,0}\}$, $p_3 = \{e_{3,2}, e_{2,1}, e_{1,0}\}$ and $p_4 = \{e_{4,2}, e_{2,0}\}$ respectively in a data/measurement collection cycle. Assume the representation format of X is given as $\{l_{1,0}, l_{1,2}, l_{1,3}, l_{1,4}, l_{2,0}, \cdots, l_{4,0}, l_{4,1}, l_{4,2}, l_{4,3}\}^T$; the link vector for the WSN routing topology will be $X = \{l_{1,0}, 0, 0, 0, l_{2,0}, l_{2,1}, 0, 0, 0, 0, l_{3,2}, 0, 0, 0, l_{4,2}, 0\}^T$ and the measurement matrix Φ will be

$$\begin{bmatrix} 1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0 \\ 1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&0&0&0&0&0&1&0 \end{bmatrix}.$$

As illustrated in this example, the link vector X has only 5 nonzero values among the total 16 elements. In general, using the common CS compression ratio $r = M/N$, we have

$$r = \frac{n-1}{(n-1)^2} = \frac{1}{n-1}, \tag{3}$$

where n is the total number of WSN nodes (including the sink). As the size of WSN grows, the compression ratio r becomes very small. Also, all potential $(n-1)^2$ directed wireless links in the WSN upstream base topology are considered without any pre-exclusion.

Due to the nature of CS formulation, we want to investigate the accuracy of our topology inference approach. As illustrated in Figure 2, it is possible that two routes satisfy the same measurement value y_i, which suggests these two routes be basically indistinguishable. We call this situation a tie, and the node i a tie node.

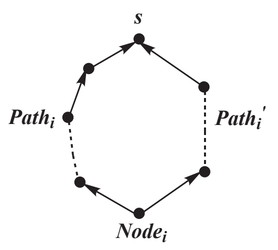

Figure 2: An example of a measurement tie, where two routes $Path_i$ and $Path_i'$ have the same measurement value.

3.2 Path Measurement Metric

A path measurement y_i is calculated in network based on the traversed link label values as a packet routed towards the sink. As in the traditional CS approaches, linear combination is adopted in our formulation. However, we employ modular summation (with mod m) (SUM$_m$) rather than regular summation, for efficient WSN in-network computing and scalable communications.

3.3 Edge Labeling Function

As defined in Section 2, each edge in G has a unique label $l_{u,v}$, given by a labeling function L. Since G is unknown and will be inferred at the sink, L should generate a unique labeling value on each edge in the base topology G^*, that is, $L : E^* \to \mathbf{N}$.

A good labeling function tries to reduce the probability of path measurement ties as much as possible. Here, we provide a simple but useful heuristic method to effectively reduce the potential tie probability: assign labeling values on all edges as odd positive integer numbers.

THEOREM 1. *For a directed acyclic graph $G = (V, E)$, if all edge labeling values are odd positive integers, any path of odd hops cannot tie with any path of even hops.*

PROOF. Let p_i denote a path originated from node i to the sink s. If the hop number of the path $|p_i|$ is odd, then its corresponding path measurement y_i will be an odd integer. Assume there is another path p_i' originated from the same node i and its $|p_i'|$ is even, then its measurement y_i' will be an even integer. Therefore, $y_i \neq y_i'$. □

Next, we devise a simple and effective labeling function which is given in Theorem 2. Our devised function generates a unique label value for any edge (u, v), if the two nodes u and v have unique odd integer IDs. The proof is omitted due to the page limit. Thus, any node receiving a packet can easily compute the label value of the link used by the packet on-the-fly, without any pre-stored link label table.

THEOREM 2. *Assume each node i has a K-bit unique and odd integer ID id_i, for any edge (u, v), the edge label $l_{(u,v)} = id_u \times 2^K XOR\ id_v + (id_v - id_u)$ is a $2K$-bit unique and odd integer value.*

4. RECOVERY ALGORITHM

To solve the problem of (2), a straightforward approach would be an exhaustive search through all the possible edge combinations to find the ones matching the given path measurements. The complexity of such a brute force approach would be $O((n-1)!)$ which is prohibitive. However, based on the sparsity assumption of X, effective recovery algorithms are possible. We first devise a basic Routing Topology Recovery (RTR) algorithm with the measurement metric of modular summation, and illustrate how basic RTR algorithm works. Then we extend the basic RTR algorithm by employing multiple path measurement metrics. Finally we develop a fast RTR algorithm.

4.1 Assumptions

Data/measurement packets are received at the sink in sequence. Good wireless links used in successful transmissions are intended to be reused for subsequent packets delivery whenever appropriate in a collection cycle. Based on the sparsity of link vector X, it would be reasonable to assume that in the dynamic routing model A-Tree $G(V, E)$, any routing path originated from an individual sensor node will not introduce more than two new wireless links in a collection cycle, that is, $|E| \leq 2(n-1)$ when $|V| = n$. In contrast, the static (i.e., spanning tree) routing has the assumption that only one new wireless link can be introduced per each routing path. Consequently, our assumption accommodates the prevalent wireless links' dynamics due to channel fading and interference, but at the same time, only allows the link dynamics to a limited extent, to exploit the sparsity of X. Recall that $|E| = n - 1 + |E^+|$, thus our assumption allows $|E^+| \leq n - 1$ shortcuts in the dynamic routing model. Clearly, if no any shortcut is allowed, the recovered routing topology will be exactly a spanning tree. As one can see, our assumption is indeed the most sparseness assumption for a *dynamic* routing topology.

4.2 Algorithm Design

Every measurement packet originated from a sensor node i contains the original node's unique ID id, and its path measurement y_i. This sink receives these packets in sequence from which two vectors are formed: a sequence vector $S = \{id_1, id_2, \cdots, id_M\}$ where the subscripts indicate the arriving order, and the corresponding measurement vector $Y = \{y_{i_1}, y_{i_2}, \cdots, y_{i_M}\}$. We present our RTR algorithm based on these two vectors. For convenience, we also use "recovering node i" to refer "recovering the path originated from node i" as for convenience. These two terms are exchangeable in this paper.

The basic idea of the RTR algorithm is for each incoming path measurement y originated from a new node *child*, the sink and all the previously recovered nodes could be its parent candidates *Candidates*. For each potential parent candidate *parent*, function $findTP$ tries to find the *child*'s all possible paths under our sparsity assumption in Section 4.1 (*i.e.*, either without new shortcut or with only one new shortcut) according to the currently recovered topology TP whose module sum aggregation matches the received y. For each match, update the topology TP by adding the edge between node *child* and node *parent* and the new shortcut if there is one. Notice, because of any tie situations, it is possible that there are multiple topology updates for the same new incoming node *child* and the same recovered topology *TP*. To ensure not to miss any true solution, record all recovered updates in a set *newSet* and every topology in *newSet* will be checked for the next new node. When no any match is found, it means that this topology TP is a "fake" one caused by a previous tie situation and it should not be considered any further. Figure 3 outlines our basic RTR algorithm.

Note that there could be two forms to represent a topology TP: one is just the A-Tree routing topology like $TP \leftarrow ATree$, and the other one will include one or multiple path recoveries(PR) like $TP \leftarrow \{ATree, \{PR_1, \cdots\}\}$. If the goal is only to recover the overall A-Tree topology, the tree only form will suffice. On the other hand, if each detailed route originated from each individual node is needed, it could be either recomputed based on the topology result of the RTR algorithm with the tree only TP form or recorded as a byproduct with the tree and path recoveries TP form. The former (i.e., based on the tree only TP form) will spend extra time for the recalculation while the latter will take some additional space to record those path recoveries. Another issue is that when multiple potential topologies are inferred for the same node, those topologies will be grouped before checking the next node to avoid redundant calculations. To group the topologies with the tree only form, it is just a simple union. For the tree and path recoveries form, all the path recoveries will be put in a set for the same tree structure.

Example 3 Figure 4 illustrates how the devised RTR algorithm works using a WSN with 5 nodes. In this network, the sink is node 0; sequence vector is $S = \{1, 2, 3, 4\}$; and the indirect path measurement vector (in the arriving order) is $Y = \{1, 4, 9, 20\}$. The labels assigned on edges are given in the figure. In Figure 4, (a) shows the initial topology state which only has the sink node 0. When the sink received the first measurement y_1 originated from node 1, node 1 did not have other parent choices except the sink node so its measurement must match the label of the edge $l_{1,0}$ as shown in (b). When node 2's measurement packet arrived at the sink, both node 0 and node 1 would be considered as its parent candidates. Since only $getPathSum(\{e_{2,1}, e_{1,0}\}) = 4$ matches y_2, node 1 is the parent of node 2 as shown in (c). For node 3, tie situation occurs. Both the sink and node 2 could be its parent nodes, so we will get two different potential topologies (d.1) and (d.2) at this step. For the next node, both these two potential topologies will be checked. Therefore, for node 4, RTR will find (e.1) based on (d.1), and (e.2.1) and (e.2.2) based on (d.2). In (e.2.2), $e_{3,0}$ is the new shortcut. In this figure, each recovered individual path for the incoming source node consists of solid bold edge(s) and possibly a blue dashed edge – a new shortcut introduced by the incoming node. Figure 4(a)-(e) represent the path recovery for all the nodes in sequence respectively, where sub-numbers such as (d.1) and (d.2) specify the two different topologies recovered for the same node at that step.

```
Notation
$getSize(s)$: return the size of the set $s$;
$s_1 \cup s_2$: join the two sets $s_1$ and $s_2$;
$group(s)$: group the same topologies in the set $s$;
$findTP$: return the set of all matched topologies.

Function RTR($S, Y, r$)
 1:  $TP \leftarrow \{\}; Set \leftarrow \{TP\}$; /*initial $TP$ and $Set$*/
 2:  for ($i \leftarrow 1; i \leq getSize(S); i++$)
 3:    $child \leftarrow S[i]$; $y \leftarrow Y[i]$; $newSet \leftarrow \{\}$
 4:    for all topologies $TP \in Set$ do
 5:      $Candidates \leftarrow \{r\} \cup S[1, \cdots, i-1]$;
 6:      for all candidates $parent \in Candidates$ do
 7:        $TPSet \leftarrow findTP(child, parent, y, TP)$;
 8:        if ($TPSet \neq \{\}$)/*exist matched topologies*/
 9:        then $newSet \leftarrow newSet \cup TPSet$;
10:      end for
11:    end for
12:    $Set \leftarrow group(newSet)$;
13:  end for
```

Figure 3: RTR algorithm to obtain a complete set of solution candidates.

4.3 RTR with Multiple Measurement Metrics

As we can see, the inferred potential solution set by our basic RTR algorithm will include the true routing topology formed in the WSN in a given collection cycle, but cannot distinguish the true solution from the other false ones when multiple candidates exist due to some path tie(s). Therefore, it is desirable to make the size of the inferred potential solution set to be as small as possible.

We propose to adopt additional path measurement metric(s) to reduce the size of the RTR solution set. In other words, in addition to the modular summation for indirect path measurement metric, supplemental measurement metric(s) could also be applied. Instead of using a single scalar measurement value y_i as before, now each measurement y_i

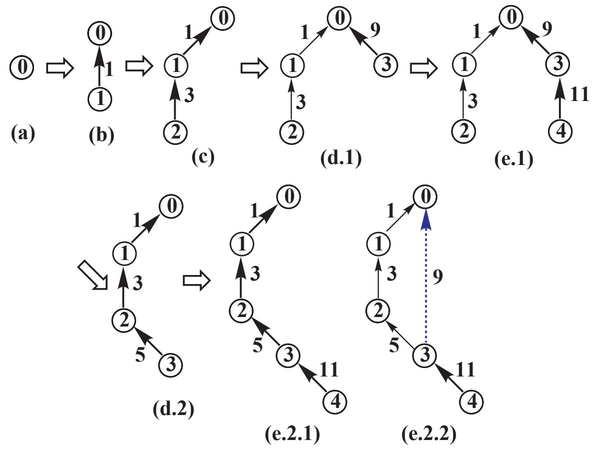

Figure 4: An illustrate example for RTR.

is a group of multiple values based on all measurement metrics, that is, $y_i = \{y_i^1, y_i^2, \cdots\}$. For example, exclusive-or (XOR) can be adopted as the secondary indirect measurement metric with the following minor modifications to the basic RTR algorithm given in Figure 3:

- Modify **Function** RTR(S, Y, r) in Figure 3, by replacing line 3 with :
  $child \leftarrow S[i]$; $y_1 \leftarrow Y[i, 1]$; $y_2 \leftarrow Y[i, 2]$; $newSet \leftarrow \{\}$;
- Extend **Function** $findTP$ by adding the secondary measurement y_2 based on the XOR metric.

4.4 Fast RTR Algorithm

RTR algorithm with multiple measurement metrics helps reduce the potential size of the solution set significantly. While the theoretical probability analysis is still an open question, we empirically observed that the probability of having more-than-one potential solutions should be extremely small using the RTR with both SUM$_m$ and XOR measurement metrics. Therefore, we also develop a Fast Routing Topology Recovery (FRTR) algorithm which only provides the first solution candidate found and then stop the further searching. With proper edge labeling value function and path measurement metrics, FRTR would be able to obtain the unique correct recovery with very high probability. The merit of FRTR algorithm is that it is significantly faster than RTR algorithm since RTR algorithm may waste resources trying to search either non-existent or duplicated solution candidates in its effort to obtain the complete set of potential solution candidates. The main changes introduced into the FRTR compared to the RTR algorithm are given below:

- Node *child* will stop testing other parent candidates *Candidates* as long as it finds one in the **for** loop (line 6-10) of **Function** RTR(S, Y, r) in Figure 3;
- **Function** $findTP$ will return the first match topology and stop searching the rest potential ones.

These changes enable us to improve the FRTR algorithm's performance by sorting the parent candidates *Candidates* and the corresponding path candidates in **Function** $findTP$ according to the properties of a given WSN routing mechanism. For instance, if nodes in a given network prefer to choose the shortest available paths, the parent candidates could be sorted in the ascending order of their levels and the path candidates could be sorted by the number of hops.

The complexity of FRTR algorithm is $O(N^2)$. The analysis and proof are omitted due to page limit.

Table 1: Parameter Ranges For Noise Generation

Parameter	Range
Baseline noise level average	[-98, -92]
Baseline noise level standard deviation	[1,3]
Burst offset average	[0,45]
Burst offset standard deviation	[1,3]
Burst sigma range	[1,3]
Burst duration average	[20,110]
Burst duration standard deviation	[5,20]
Burst frequency average	[0,3]
Burst frequency standard deviation	[1,2]

4.5 Simulation Study

We conducted simulations to validate the devised RTR and FRTR algorithms above, with the measurement metrics Sum$_m$ and XOR. The following setting is used: (1) each node i has a 12-bit unique odd ID id_i; (2) each edge label $l_{(u,v)}$ is generated according to Theorem 2; and (3) each metric of the module sum and XOR uses three bytes.

WSN routing topologies were simulated as follows with respect to random noises introduced in the simulations. An establishment of a wireless link is based on the checking of signal to noise ratio (SNR). If SNR is less than the predefined threshold [22], no link exists between the two sensor nodes. Here we used the same radio gain for all links in a WSN, and simulated both the random noises at short-time scales and the bursty noises at relatively long-time scales [22] independently for each link. Table 1 shows the ranges of noise parameters used in the simulations. Simulation results are listed in Table 2, where column **WSN Size** lists the total number of nodes in each simulated WSN; column **Hgt** shows the length of the longest routing path in terms of hops in the WSN; and column **SC Ratio** is the ratio of the number of shortcuts to the number of all edges (including shortcuts) in the routing topology A-Tree. These three columns show the basic structure of the WSN routing topologies in our simulations. The network sizes of the simulated WSNs range from 30 to 130 nodes. We can see from Table 2 that the SC ratios of these WSNs range from 0.07 (1/14) to 0.43(93/214), representing a good diversity of dynamic routing scenarios under our sparseness assumption. Column **RTR Size** indicates the size of the inferred potential solution sets by the RTR algorithm for each simulated WSN. In our simulations, the unique solution is obtained for all simulated WSNs by the RTR algorithm with two measurement metrics, although in general, there is no guarantee that the unique true solution can be always obtained. In contrast, the basic RTR algorithm with a single measurement metric SUM$_m$ may often produce more than one potential solution candidates. For example, the basic RTR algorithm produces 3 potential solutions for the first WSN test case listed in Table 2 (marked with *). When additional measurement metric XOR is adopted in our sophisti-

Table 2: Comparison between RTR and FRTR

WSN Size	Hgt	SC Ratio	RTR Size	RTR/FRTR
40	7	1/14	1*	2.7
56	9	24/79	1	2.8
63	10	33/95	1	3.2
66	8	12/77	1	2.8
76	10	46/121	1	3.4
85	10	23/107	1	3.5
96	9	1/6	1	2.7
105	9	29/133	1	2.6
112	10	70/181	1	3.0
122	14	93/214	1	3.0

cated RTR or FRTR algorithm, there is an overhead of three bytes for a path packet. The last column **RTR/FRTR** is the ratio of the CPU time of the RTR to the CPU time of the FRTR. As we can see, the result shows that FRTR is averagely about 2.9 times faster than RTR in the simulations.

5. CONCLUSIONS

In this paper, we have proposed a novel approach to WSN dynamic routing topology inference from indirect measurements observed at the data sink. We formulate the problem from compressed sensing perspective in an innovative way. We devise a suite of algorithms to recover routing topology at the sink. We conducted empirical studies on our devised recovery algorithms and the simulation results are promising. Our future work includes to study our proposed WSN dynamic routing topology inference with incomplete path measurement set in a collection cycle due to packet loss in real-world environments. We plan to further investigate our WSN dynamic routing topology inference approach for large-scale of WSNs consisting of thousands of nodes. We also plan to implement the proposed approach and test it thoroughly in a real-world WSN testbed. Based on the dynamic topology inference, current WSN link loss and delay inference schemes can be extended to deal with realistic WSNs under dynamic routing.

6. ACKNOWLEDGMENT

This work is supported in part by National Science Foundation under grant CNS-1252066.

7. REFERENCES

[1] D. Estrin, D. Culler, K. Pister, and G. Sukhatme. Connecting the physical world with pervasive networks. *Pervasive Computing*, pages 59–69, 2002.

[2] P. Bonnet, J. Gehrke, and P. Seshadri. Querying the physical world. *IEEE Personal Communications*, pages 10–15, Oct. 2002.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communications Magazine*, 40(8):102–116, Aug. 2002.

[4] M. Coates, A. Hero, R. Nowak, and B. Yu. Internet tomography. *IEEE Signal Processing Mag.*, pages 47–65, May 2002.

[5] Y. Chen, D. Bindel, H. Song, and R. Katz. An algebraic approach to practical and scalable overlay network monitoring. In *Proc. ACM SIGCOMM, Portland, Oregon*, Aug. 2004.

[6] D. Chua, E. Kolaczyk, and M. Crovella. Efficient monitoring of end-to-end network properties. In *Proc. INFOCOM*, pages 1701–1711, 2005.

[7] J. Ni, H. Xie, S. Tatikonda, and Y. R. Yang. Network routing topology inference from end-to-end measurements. In *Proc. IEEE INFOCOM*, pages 36–40, 2008.

[8] F. L. Presti, N. G. Duffield, J. Horowitz, and D. Towsley. Multicast-based inference of network-internal delay distributions. *IEEE/ACM Trans. Netw.*, 10(6):761–775, 2002.

[9] M. H. Firooz and S. Roy. Network tomography via compressed sensing. In *Proc. IEEE GLOBECOM*, pages 1–5, 2010.

[10] J. Gui, V. Shah-Mansouri, and V. W. S. Wong. Accurate and efficient network tomography through network coding. *IEEE Trans. Vehicular Tech.*, 60(6):2701–2713, 2011.

[11] C. Mark, P. Yvan, and R. Michael. Compressed network monitoring. In *Proc. IEEE Statistical Signal Processing*, pages 418–422, 2007.

[12] H. Nguyen and P. Thiran. Using end-to-end data to infer lossy links in sensor networks. In *Proc. IEEE INFOCOM*, 2006.

[13] Y. Lin, B. Liang, and B. Li. Passive loss inference in wireless sensor networks based on network coding. In *Proc. IEEE INFOCOM*, pages 1809–1817, 2009.

[14] G. Hartl and B. Li. Loss inference in wireless sensor networks based on data aggregation. In *Proc. of IPSN*, 2004.

[15] Y. Mao, F. R. Kschischang, B. Li, and S. Pasupathy. A factor graph approach to link loss monitoring in wireless sensor networks. *IEEE J. Selected Area in Comm.*, 23(4):820–829, 2005.

[16] V. Shah-Mansouri and V. W. S. Wong. Link loss inference in wireless sensor networks with randomized network coding. In *Proc. IEEE GLOBECOM*, pages 1–6, 2010.

[17] T. Zhao, W. Cai, and Y. Li. MPIDA: A sensor network topology inference algorithm. In *Proc. Int. Conf. on Computational Intelligence and Security*, pages 451–455, 2007.

[18] Y. Yang, Y. Xu, and X. Li. Topology tomography in wireless sensor networks based on data aggregation. In *Proc. Int. Conf. on Communications and Mobile Computing*, pages 37–41, 2009.

[19] E. Candes and T. Tao. Decoding by linear programming. *IEEE Trans. Info. Theory*, 51(12):4203–4215, 2005.

[20] E. Candes, J. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Info. Theory*, 52(2):489–509, 2006.

[21] D. L. Donoho. Compressed sensing. *IEEE Trans. on Information Theory*, 52(4):1289–1306, 2006.

[22] K. Srinivasan, M. A. Kazandjieva, S. Agawal, and P. Lewis. The β-factor: Measuring Wireless Link Burstiness. *Proc. 6th ACM Conf. on Embedded Networked Sensor Systems (SenSys)*, 2008.