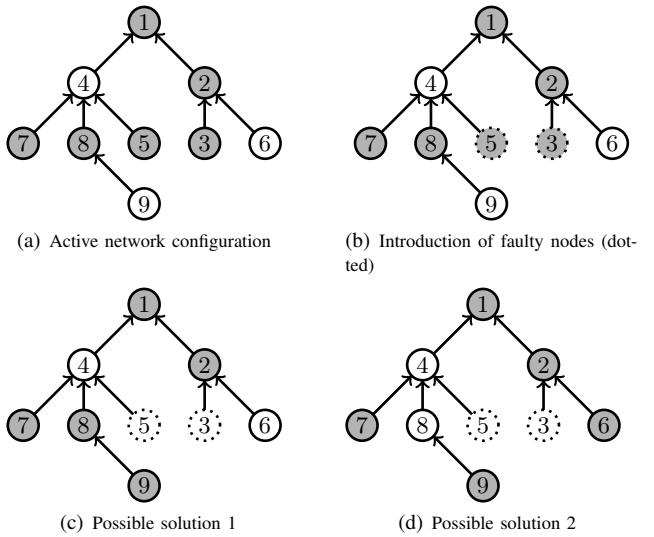
ERUPT: Energy-efficient tRUstworthy Provenance Trees for Wireless Sensor Networks

S. M. Iftekharul Alam
School of ECE
Purdue University
Email: alams@purdue.edu

David K. Y. Yau
Information Systems Technology & Design
Singapore University of Technology & Design
Email: david_yau@sutd.edu.sg

Sonia Fahmy
Department of Computer Science
Purdue University
Email: fahmy@cs.purdue.edu

***Abstract*—Sensor nodes are inherently unreliable and prone to hardware or software faults. Thus, they may report untrustworthy or inconsistent data. Assessing the trustworthiness of sensor data items can allow reliable sensing or monitoring of physical phenomena. A *provenance-based* trust framework can evaluate the trustworthiness of data items and sensor nodes based on the intuition that two data items with similar data values but with different provenance (i.e., forwarding path) can be considered more trustworthy. Forwarding paths of data items generated from redundantly deployed sensors should consist of trustworthy nodes and remain dissimilar. Unfortunately, operating many sensors with dissimilar paths consumes significant energy. In this paper, we formulate an optimization problem to identify a set of sensor nodes and their corresponding paths toward the base station that achieve a certain trustworthiness threshold, while keeping the energy consumption of the network minimal. We prove the NP-hardness of this problem and propose ERUPT, a simulated annealing solution. Testbed and simulation results show that ERUPT achieves high trustworthiness, while reducing total energy consumption by 32-50% with respect to current approaches.**



(a) Active network configuration (b) Introduction of faulty nodes (dotted)

(c) Possible solution 1 (d) Possible solution 2

Fig. 1. An example 3 × 3 grid wireless sensor network (intermediate white nodes can be forwarder/relay; leaf white nodes are inactive)

I. Introduction

Planet-wide sensor networks [1], sensor networks for large-scale urban environments [2], and physical infrastructure systems [3] indicate potential deployments of multi-hop networks consisting of hundreds of sensor nodes. The applications involving sensor networks are becoming diverse, ranging from ecosystem management [4] and life saving coal mine monitoring systems [5] to SCADA systems [6]. Sensor nodes are usually left unattended and are vulnerable to hardware/software faults [7] which may cause them to report erroneous or corrupted data. Enabling seamless operation of critical services requires a correct assessment of the trustworthiness of sensor data reports.

In order to sense the physical environment reliably, sensors are deployed redundantly and their reports are collectively processed at the base station. The intuition is that when measurements referring to the same event have similar values, these measurements are trustworthy. *Provenance-based trust frameworks* [8], [9] can evaluate the trustworthiness of sensor data. Such frameworks utilize data provenance along with the sensed data values to compute the trust score of each data item. Provenance of a data item includes knowledge of the originator and processing path of data since its generation. In addition to value similarity, provenance-based trust frameworks rely on the principle that two data items having different provenance can be considered "supportive" to each other. A node that generates or delivers highly trustworthy data is considered "trusted."

With more sensor nodes active and the forwarding paths (i.e., provenance) of the data items generated by these nodes trustworthy and dissimilar, trustworthiness of the data items can be high. We conducted simulations to explore how the trust scores increase with more active sensors and with data items having dissimilar provenance. Our study exposes that dissimilar provenance comes at the cost of transmission overhead since a different forwarding path from the default usually requires a higher number of expected transmissions. In other words, trust comes at the expense of (i) the sensing energy consumption of the redundant sensors, and (ii) the transmission energy consumption to make provenance dissimilar.

In this paper, we aim to reduce the set of active sensing nodes and select energy-efficient paths (having low expected number of transmissions) from the selected nodes to the base station while ensuring that the selected paths contain trustworthy nodes and exhibit low correlation among them. In order to understand this goal, consider an example 3 × 3 grid network. When all the nodes are trustworthy (initially), a set of nodes and their corresponding paths are selected as shown in Fig. 1(a). We note that gray nodes are actively sensing whereas white nodes are not. However, gray and white intermediate nodes can be relay nodes if they are present on the forwarding

paths from other active sensing nodes to the base station. All the other nodes (white leaves) remain inactive. The subsequent figure shows a scenario with two faulty (dotted) nodes 3 and 5, which will eventually lower the the trust score of the generated data items. To overcome this problem, there are two alternative network configurations shown in Fig. 1(c) and 1(d). While both configurations can improve trustworthiness, Fig. 1(d) is preferable (assuming both require similar numbers of expected transmissions) since it offers higher provenance dissimilarity, and an additional faulty node cannot affect the two newly active nodes (6 and 9) in the future. The resulting network will be energy-efficient and trustworthy.

We consider that the trustworthiness of a particular path is governed by the least trustworthy node on that path and define overall trustworthiness of the network as a weighted sum of the trustworthiness of all paths and the dissimilarity among the paths. Note that trust scores of nodes evolve over time reflecting their historical behavior (e.g. reporting correct or false data etc.). Thus a set of nodes with better path diversity and consistency in reporting correct data will form a network with better trustworthiness. We pose an optimization problem to determine a set of nodes and their corresponding paths that minimize expected transmissions and achieve a certain threshold for the trustworthiness of the network. Depending on the reliability requirements of the application, an appropriate threshold for trustworthiness can be chosen.

We provide an integer programming formulation of the optimization problem described above and prove its NP-hardness. We solve the problem using simulated annealing. Our solution, ERUPT, outputs a routing tree rooted at the base station that achieves a specified trustworthiness level with a low expected number of transmissions (i.e., low energy consumption). We implement the proposed solution in Java at the base station, and integrate it with sensor nodes on TOSSIM. We further port our implementation to TelosB motes and perform experiments on a testbed consisting of 30 motes. Both testbed and simulation results show that ERUPT achieves high average trustworthiness, while reducing total energy consumption by 32-50% compared to existing approaches.

The remainder of this paper is organized as follows. Section II discusses related work. Section III gives some background on provenance-based trust frameworks. We give the motivation behind our work in Section IV. Section V gives our system model and problem formulation. We explain ERUPT, our simulated annealing-based solution, in Section VI. Section VII and VIII present TOSSIM simulation and testbed results respectively. We conclude in Section IX.

II. Related Work

In addition to the provenance-based trust frameworks discussed in the next section, a few trust frameworks have been proposed in the literature [10], [11] where nodes monitor the behavior of their neighbors in order to establish trust relationships. *TIBFIT* [12] estimates the trust of a node based on the fidelity of its previous event reports as seen by the base station. However, none of these approaches considers provenance dissimilarity to assess the trust score of data items.

Secure routing and aggregation techniques [13] can make decision with input from trust frameworks. Leligou et al. [14] use a weighted routing cost function to balance trust and location information. However, existing methods lack energy-awareness. A few trust and energy-aware routing protocols [15], [16] base routing decisions on both the trust and residual energy of neighboring nodes (in addition to the routing cost). This eventually achieves load balancing and higher network lifetime. However, unlike our approach, they do not consider the trade-off between the trustworthiness and the energy cost associated with using redundant sensors.

While redundant sensors are placed to improve the probability of target coverage, a number of methods [17], [18], [19] have been proposed to extend the network lifetime by selectively activating a set of sensors covering the desired area. Like our approach, these methods save energy and extend network lifetime by keeping only a subset of nodes active at a time. However, the solutions to the coverage problem are mainly concerned with organizing sensors into multiple sets, reducing communication overhead, and ensuring better connectivity. In contrast, our focus is on finding a subset of trustworthy nodes and regulating their routes toward the base station such that these routes exhibit low expected number of transmissions and low correlation.

III. Background

In this section, we discuss redundant placement of sensor nodes, potential sources of inconsistent sensor data values and provenance-based trust frameworks.

A. Grouping Data

To assess trustworthiness, it is important to identify data items that pertain to the same event. Intuitively, reports from a particular region should be similar and thus represent the same event, though the boundaries of a region differ from one application to another. Node proximity (determined by a clustering scheme, e.g., [20]) can be used to determine the set of nodes that should report the same event and similar values.

To validate this, we analyze a real-world public sensor data set available from a prototype sensor network deployed by Intel-Berkeley Research Lab [21]. There are 54 sensors arranged in a lab of size 40 m $\times$ 30 m. Each sensor reports temperature, light, and voltage values every epoch of 31 seconds. When all the nodes are operating correctly between epochs 500 and 600, we find that the differences between data values reported by node 1 and the other nodes vary between 0 and 10. The differences between data values reported by node 1 and its closest 10 nodes are negligible ($0 \sim 0.35$). The sensor nodes deployed at a high density are expected to report similar values during an event [22].

B. Sources of Inconsistent Data

Inconsistent sensor data can be produced by faulty sensor nodes with hardware or software bugs. Sensor nodes are typically inexpensive and not equipped with fault recovery mechanisms. Thus unreliable hardware components or incomplete software updates may cause sensors to report corrupted data either intermittently or for a time period. It has been observed that many sensor network deployments (e.g., James Reserve [23], Intel Berkeley [21], the CENS deployment in

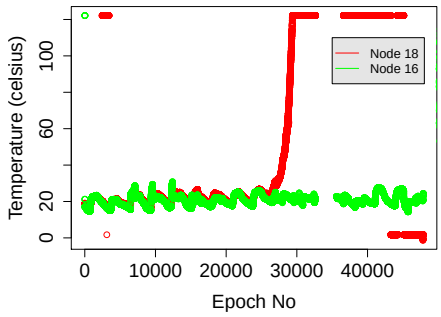


Fig. 2. Data values reported by two nearby sensor nodes

Bangladesh [24]) exhibit a large number of malfunctioning nodes due to faults in software and hardware [7].

Fig. 2 shows temperature values collected from two nearby sensor nodes (18 and 16) in the Intel Berkeley testbed. After the initial setup, both nodes report similar data values. However, at some point, node 18 starts reporting unusual temperature values ($\sim$ 120 or 0 degree Celsius) whereas node 16 reports correct values. This shows that faulty nodes are present even in a controlled environment.

C. Provenance-based Trust Frameworks

Provenance-based trust frameworks provide an effective way to evaluate the trustworthiness of data items and their sources, even in the presence of inconsistent data reported by faulty nodes. In a multi-hop network, provenance includes knowledge of the originator and processing path of data since its generation. A *trust score* is computed for each data item and each node: trust scores of data items affect the trust scores of nodes, and vice versa. Each set of data items pertaining to the same physical event and received at the base station within a specified time window should have similar values. Trust scores are continuously updated upon reception of data items. Next we describe a variant of trust frameworks proposed in [8], [9].

Assume that a collection of data items, $C^{(e)}$, pertaining to an event e are received at the base station. $T^{(e)}(i)$ indicates the adjusted trust score of item $i \in C^{(e)}$ after the occurrence of the e-th event. There is also a trust score estimate $R^{(e)}(n) \in [0,1]$, for each node n in the network, that is updated after every event e. Initially, $R^{(0)}(n) = 0.5$ for all nodes in the network.

The trust score of each data item, $T^{(e)}(i)$, is adjusted based on the *value similarity* and *provenance dissimilarity* of the data items present in the collection $C^{(e)}$. If similar data values have different provenance, the trustworthiness of data items increases. Two data items having different provenance can be considered independent supportive evidence. In contrast, if they share a similar provenance, the support for trustworthiness is less strong. Thus, both *value similarity* and *provenance dissimilarity* of a data item contribute to its trust score.

We consider the following definition of *provenance dissimilarity score*, ($\rho_i^{(e)}$):

$$\rho_i^{(e)} = \frac{\sum_{j \in C^{(e)}, j \neq i} D(i,j)}{|C^{(e)}| - 1}$$

where $D(i,j)$ indicates the provenance difference between two data items i and j.

$$D(i,j) = 1 - \frac{1}{2}\left(\frac{|G_i \cap G_j|}{|G_i|} + \frac{|G_i \cap G_j|}{|G_j|}\right)$$

Here, G_i and G_j indicate provenance corresponding to the data items i and j.

The value similarity score $\theta_i^{(e)}$ is computed for each item $i \in C^{(e)}$,

$$\theta_i^{(e)} = \frac{\sum_{j \in |C^{(e)}|, j \neq i} \delta(i,j)}{|C^{(e)}| - 1}$$

where, $\delta(i,j) = -||v_i - v_j||^2$. Here, v_i and v_j indicate the values of the items i and j, respectively.

Both provenance dissimilarity and value similarity scores are normalized to map between -1 and 1. The adjustment is calculated as $\Delta T_i^{(e)} = \exp^{-\frac{1}{|C^{(e)}|}} \rho_i^{(e)} \theta_i^{(e)}$, where the term $\exp^{-\frac{1}{|C^{(e)}|}}$ indicates that the more items there are in the collection, the more influence this adjustment should have.

Finally, the trust score $T^{(e)}(i)$ of data item i is calculated based on the trust score of the nodes present on the provenance of i and the adjustment calculated above is:

$$T^{(e)}(i) = w_1 \min_{m \in G_i} R^{(e-1)}(m) + (1 - w_1)\Delta T_i^{(e)}, \ \ \forall_i \in C^{(e)}$$

Now assume that $C_n^{(e)}$ includes all the data items in $C^{(e)}$ that originate from or pass through node n. Then the trust score of each node n is updated as follows:

$$R^{(e)}(n) = w_2 R^{(e-1)}(n) + (1 - w_2)\frac{\sum_{j \in C_n^{(e)}} T^{(e)}(j)}{|C_n^{(e)}|}$$

Here, w_1, w_2 are two weight parameters.

IV. Motivation

We now discuss how increasing the network size and increasing data provenance dissimilarity enhances the trustworthiness of data values reported by sensor nodes. However, this increased trustworthiness comes at the expense of increased energy consumption, which motivates us to explore the trade-off between trustworthiness and energy consumption.

A. Effect of Network Size

To understand the effect of the number of sensor nodes on trust scores of data items, we conduct TOSSIM simulations on a 7×7 grid network by varying the number of active sensing nodes from 5 to 48. The base station is located at (0,0) and all the sensing nodes send the same data values. Fig. 3 shows the average trust scores of the data items after 20 iterations. The average trust score increases as the number of participating sensor nodes increases. However, increasing the number of active sensor nodes increases the overall energy consumption of the network.

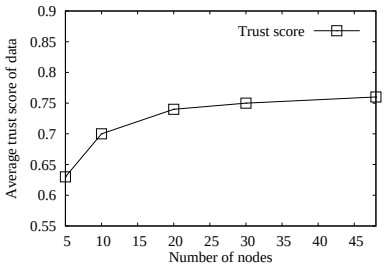


Fig. 3. Trust score of data items for different numbers of nodes

B. Effect of Dissimilar Provenance Values

Provenance (i.e., forwarding paths) of sensor data items are defined and maintained by a routing protocol among the sensor nodes. These paths or routes are usually built as part of a minimum cost tree rooted at the base station. The cost of a route is the sum of the costs of individual links toward the base station. Using expected number of transmissions (ETX) [25] as the cost metric, a forwarding node will route the data item to the next hop that will require the smallest estimated number of transmissions (to reach the base station) compared with other neighbors. This reveals an interesting trade-off between trustworthiness (or provenance dissimilarity) and transmission overhead: making the provenance more dissimilar increases the transmission overhead in general.

TABLE I. EFFECT OF PROVENANCE DISSIMILARITY ON TRUST SCORE

ETX threshold (th_{etx})	**Provenance dissimilarity**	**Average trust of data items**	**Number of transmissions**
0	0.52	0.78	1626
10	0.53	0.79	1757
20	0.66	0.84	1964
30	0.81	0.9	2559

We borrow the *controlled routing* method [26] to observe the interaction between provenance dissimilarity and corresponding transmission overhead. The *controlled routing* method replaces the *next hop* function of the TinyOS routing protocol CTP [25] with a parameterized function that takes an argument called *ETX threshold (th_{etx})*. This function considers all the nodes in the routing table to be eligible forwarders as long as they have an estimated ETX lower than the smallest ETX plus th_{etx}. While forwarding a packet, a hash is performed on the node identifier of the source node of the packet to generate a value i between 1 and the number of eligible forwarders. The i-th node in the eligible forwarder list is returned as the next hop for the packet.

By increasing the value of th_{etx}, it is possible to increase the degree of dissimilarity among the provenance of a collection of data items at the cost of higher transmission overhead, and vice versa. We perform TOSSIM simulations using the *controlled routing* method over a 7×7 grid network, varying th_{etx} between 0 and 30. We make nodes 2, 3, and 4 data originators and keep the data values reported from them fixed over different packets, in order to focus on the role of provenance similarity. For each ETX threshold value, we collect average trust score of data items after 20 events.

Data items share provenance at lower th_{etx} and exhibit different provenance at higher th_{etx}. Table I shows the effect of provenance dissimilarity on the trustworthiness of data items: a higher dissimilarity in provenance increases the trustworthiness. As the ETX threshold increases, dissimilarity among provenance increases at the expense of increased energy consumption resulting from the additional transmission overhead.

V. PROBLEM FORMULATION

We consider a multi-hop sensor network where nodes are deployed densely and report sensed data about the same event to the base station (BS). Each sensor node can be a sensing node, a forwarder (i.e., relay) node, or both. According to the trust framework, the trustworthiness of data items depends on their value similarity, provenance dissimilarity, and the trust scores of related (sensing and relay) nodes. Though the data values reported by sensor nodes cannot be controlled, it is possible to enhance the trustworthiness of data items by adding more sensor nodes and choosing dissimilar routing paths containing the trustworthy nodes. In this regard, we define the trustworthiness estimate of the network as a weighted sum of two terms: (i) aggregate trust score of the least trustworthy nodes present on the paths from all the sensing nodes to the BS, and (ii) aggregate dissimilarity score of all possible pairs of paths. Our goal is to minimize the number of active sensing nodes and the expected number of transmissions along their routes to the BS, while ensuring a certain level of trustworthiness in the network. Administrators may choose an appropriate trustworthiness threshold depending on the reliability requirements of their applications.

A. Network Model

Each node reports its one-hop neighbor information to the BS after deployment, and each sensing node generates data periodically in a particular round of reporting, where each round is driven by the occurrence of an event:

- We model the sensor network as a directed graph $G = (N, E)$ where N is the set of all nodes, $S \subseteq N$ is the set of sensing nodes, and E is the set of edges between the nodes. $N = \{i | i \text{ is a network node}\}$. $E = \{e_{ij} | e_{ij} \text{ is the edge between node } i \text{ and } j\}$.
- Q_i is the set of all possible paths from node $i \in S$ to BS, labeled $j = 1 \cdots |Q_i|$.
- P_{ij} is the j-th possible path from i to BS, $j = 1 \cdots |Q_i|$.
- Cost of a single path, P_{ij} = $C(P_{ij})$ = Expected number of transmissions along that path.
- $R(i) \in [0, 1]$ is the reputation/trust score of node i.
- Trust of a single path, P_{ij} = $T(P_{ij})$ = $\min_{e_{mn} \in P_{ij}} R(n)$.
- $D(\cdot, \cdot) \in [0, 1]$ indicates path/provenance dissimilarity between two paths, $D(P_{ij}, P_{kl}) = 1 - \frac{|P_{ij} \cap P_{kl}|}{2}(\frac{1}{|P_{ij}|} + \frac{1}{|P_{kl}|})$.

B. Fault Model

We assume the following model: (1) The BS is trusted, but any other node can be faulty; (2) a faulty node reports false data or forwards corrupted data resulting from internal bugs;

(3) the majority of the data reports are truthful; (4) complete removal of provenance is not allowed since that would make the affected data item highly suspicious; and (5) provenance information satisfies the security properties of confidentiality, integrity, and freshness based on existing methods [27].

C. Problem Statement

We define a set of 0-1 variables,

$$X_{ij} = \begin{cases} 1, & P_{ij} \text{ is selected} \\ 0, & \text{otherwise} \end{cases}$$

Thus we find that the:

- Number of paths selected, $Z = \sum_{i \in S} \sum_{j \in Q_i} X_{ij}$.
- Aggregate trust score estimate for all paths = $\sum_{i \in S} \sum_{j \in Q_i} X_{ij} T(P_{ij})$.
- Path dissimilarity of any path P_{ij} from other paths = $\frac{1}{Z-1} \sum_{k \in S} \sum_{l \in Q_k} X_{kl} D(P_{ij}, P_{kl})$.
- Aggregate path dissimilarity estimate = $\frac{1}{Z-1} \sum_{i \in S} \sum_{j \in Q_i} \sum_{k \in S} \sum_{l \in Q_k} X_{ij} X_{kl} D(P_{ij}, P_{kl})$.
- Overall normalized trust estimate of the network =

$$\frac{1}{|N|}(c_1 \sum_{i \in S} \sum_{j \in Q_i} X_{ij} T(P_{ij}) + \frac{c_2}{Z-1} \sum_{i \in S} \sum_{j \in Q_i} \sum_{k \in S} \sum_{l \in Q_k} X_{ij} X_{kl} D(P_{ij}, P_{kl})),$$

  where c_1, c_2 are two weight parameters.

We want to select a set of paths that constitute a tree having a minimum expected number of transmissions while ensuring a certain level of trustworthiness, H, of the network.

$$\begin{aligned} \min & \sum_{i \in S} \sum_{j \in Q_i} X_{ij} C(P_{ij}) \\ s.t. \quad & 0 \leq \sum_{j \in Q_i} X_{ij} \leq 1, \ \ \forall_i \in S, \\ & Z = \sum_{i \in S} \sum_{j \in Q_i} X_{ij}, \\ & \frac{1}{|N|}(c_1 \sum_{i \in S} \sum_{j \in Q_i} X_{ij} T(P_{ij}) + \\ & \frac{c_2}{Z-1} \sum_{i \in S} \sum_{j \in Q_i} \sum_{k \in S} \sum_{l \in Q_k} X_{ij} X_{kl} D(P_{ij}, P_{kl})) \geq H, \\ & 0 \leq c_1 \leq 1, 0 \leq c_2 \leq 1 \end{aligned}$$

D. Proof of NP-hardness

We define the decision version of our problem as: $ERUPT(N, S, C(\cdot), \{P_{ij}\}, T(\cdot), D(\cdot, \cdot), c_1, c_2, H, V)$: Is there a set of paths that constitutes a tree, has total number of expected transmissions at most V, and has overall normalized trust score at least H?

Now we reduce the $0/1$-Knapsack problem [28], which is NP-hard, to our problem. The $0/1$-Knapsack problem takes a collection of items $g_1, g_2, \cdots g_n$ as input, where item g_j has integer weight $w_j > 0$ and gives an integer benefit $b_j > 0$. There is a maximum weight W and minimum benefit B. The decision problem is defined as: is there a subset of the items whose total weight does not exceed W, and whose total benefit is at least B?

Knapsack problem: $n = 4$, $W = 7$, $A = 50$, $B = 23$

Items	g_1	g_2	g_3	g_4
Weight	2	3	5	6
Benefit	10	15	7	18

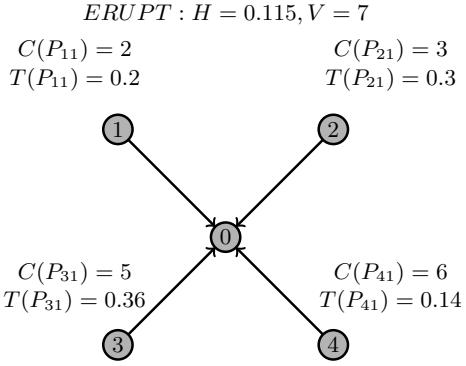


Fig. 4. An example mapping between an instance of Knapsack and our problem

We start with a graph, G, which has a single vertex 0 (the root node). Now for each item g_j in the knapsack, we add one node j connected to the root node 0 in G. Thus, there is a single path available from each node to the root node, and any combinations of these paths form a tree. Assuming that the summation of benefits over all items in the Knapsack problem equals A, we construct an instance of our problem, $ERUPT(N, S, C(\cdot), \{P_{ij}\}, T(\cdot), D(\cdot, \cdot), c_1, c_2, H, V)$, where,

- $N = S = \{1, 2, \cdots, n\}$,
- $C(P_{i1}) = w_i$, $i \in S$,
- $T(P_{i1}) = \frac{b_i}{A}$, $i \in S$,
- $D(P_{i1}, P_{j1}) = 1$, $i, j \in S$,
- $c_1 = 1, c_2 = 0$,
- $H = \frac{B}{An}$,
- $V = W$.

Fig. 4 shows an example construction of an instance of our problem from a given Knapsack problem. The construction can be done in polynomial time. The equivalence between the two problems is clear: if there is a solution to the Knapsack problem, we can select paths corresponding to the items chosen in the Knapsack and determine whether the selected paths form a tree with cost = $\sum w_i \leq W = V$ and trust = $\sum \frac{b_i}{An} \geq \frac{B}{An} = H$, and vice versa.

VI. ERUPT Design and Implementation

Since our formulated problem is NP-hard, we propose to use simulated annealing [29]. The solution requires knowledge of all possible paths from all nodes to the BS. This can be obtained by collecting 1-hop neighbor information from all the nodes. The BS will continually estimate the trustworthiness of the network based on feedback from the trust framework. Whenever the estimate goes below the given threshold, our

proposed solution identifies a set of nodes and corresponding paths that will achieve the desired trustworthiness. Information about the selected nodes and routing (i.e., path) updates will be propagated from the BS to the relevant nodes using a data dissemination protocol.

A. Simulated Annealing Basics

Simulated annealing is a probabilistic method to minimize the cost function of a combinatorial optimization problem even if it possesses many local minima. It resembles the physical process in which a metal is slowly cooled down, so that its structure will freeze in a minimum energy configuration. The notion of slow cooling is implemented as a slow decrease in the probability of accepting worse solutions while the algorithm explores the solution space. Accepting worse solutions with positive probability gives the algorithm a chance to get out of local optima.

B. Design Decisions

To use simulated annealing for our problem, we make four important design decisions: (i) defining the search space; (ii) finding a feasible solution as a starting point to initiate the search procedure; (iii) defining the neighborhood structure of candidate solutions; and (iv) defining the cost function that allows to determine the merit of the candidate solutions. We discuss these decisions in the following.

1) Search space: For each sensor node, we collect 1-hop neighbor information (neighbor identifier and link quality) and construct a graph. This graph is likely to contain redundant and cyclic paths from certain nodes to the designated root node that is the base station (BS). We need a simple acyclic path from every node to the BS, and wish to avoid paths that are costly. Towards these objectives, we apply a k-shortest path algorithm [30] over the constructed graph, and consider at most k paths for each node. These paths form the search space of the proposed simulated annealing algorithm.

2) Initial solution: We start with an empty solution and iteratively add a randomly selected path to a tentative solution until it becomes feasible. A solution is considered feasible when its set of paths form a tree, and the trustworthiness estimate for the set of paths and member nodes exceeds the given threshold.

3) Neighborhood solution: We define a solution or configuration T' to be a neighbor of a certain solution T if T' can be obtained from T with a change that preserves the tree property of the solution. More specifically, the following three options are equally likely to be selected to transform a given solution T to T': (i) remove a path from T that has the highest cost-to-trust ratio so that all the sensors on the removed path become either inactive (if they have no descendants) or relay only in T', maintaining the existing tree structure (ii) replace a path from T that has the highest cost-to-trust ratio with a randomly selected path until the new configuration T' becomes a tree; and (iii) add a randomly selected path to T until the new configuration T' becomes a tree.

4) Cost function: Our problem statement in the previous section provides a natural definition of the cost function, which is the expected number of transmissions along the tree constructed from the set of paths in the candidate solution.

Algorithm 1 Finding a neighboring solution

```
function GETNBRSOLUTION(T, Paths, H)
    repeat
        r ← rand()
        p ← Paths(Paths.size() * rand())
        q ← path ∈ T with highest cost to trust ratio
        if r ≤ 1/3 then
            T' ← T − q
        else if 1/3 < r ≤ 2/3 then
            T' ← T − q
            T' ← T' ∪ p
        else
            T' ← T ∪ p
        end if
    until T' is a tree
     return T'
end function
```

C. Implementation

Our simulated annealing algorithm, ERUPT, requires information about all possible paths from sensor nodes to the BS, and their associated costs and trustworthiness scores. The sensor network starts with a default routing (e.g., CTP [25] in TinyOS) and a provenance-based trust framework. Since provenance is transmitted along with data, we exploit the provenance field of data packets to include 1-hop neighbor information of each node (i.e., ID of the neighbor as well as the expected number of transmissions to reach that neighbor). The BS will thus maintain information about the topology of the entire network, which allows it to retrieve provenance information indirectly without embedding provenance in the data packets explicitly. Taking this provenance information into consideration, the trust framework calculates the trustworthiness of data received at the BS and the reputation of nodes responsible for generating and forwarding this data.

Once we have knowledge of the entire network in the form of a graph, we apply the k-shortest path algorithm over the graph, in which the cost of a path indicates the expected number of transmissions from its starting point to the BS. Information on these paths is passed to the simulated annealing algorithm along with the trustworthiness threshold, H, as shown in Fig. 5. The simulated annealing starts by picking a feasible initial solution and setting it as both the current and best solution so far. In each iteration, say t, we select a neighboring solution according to the algorithm 1 and update the current solution to be this new neighbor if it has a lower cost than the current one ($\Delta C < 0$) or $\exp^{-\frac{\Delta C}{t}} > rand()$, where ΔC is the difference between the new and original costs and $rand()$ is a uniform random number $\in [0, 1]$. If the new solution has a trust estimate greater than the threshold H and a lower cost than the best so far, the algorithm notes the new one as the best so far. When the simulated annealing stops, it outputs the best solution so far, which consists of the selected set of nodes and their corresponding paths.

We use a data dissemination protocol to propagate route information to the relevant nodes to override default routing decisions. Since the trust framework continually evaluates the trustworthiness of the data items and their associated nodes, our system compares the overall trustworthiness of the network

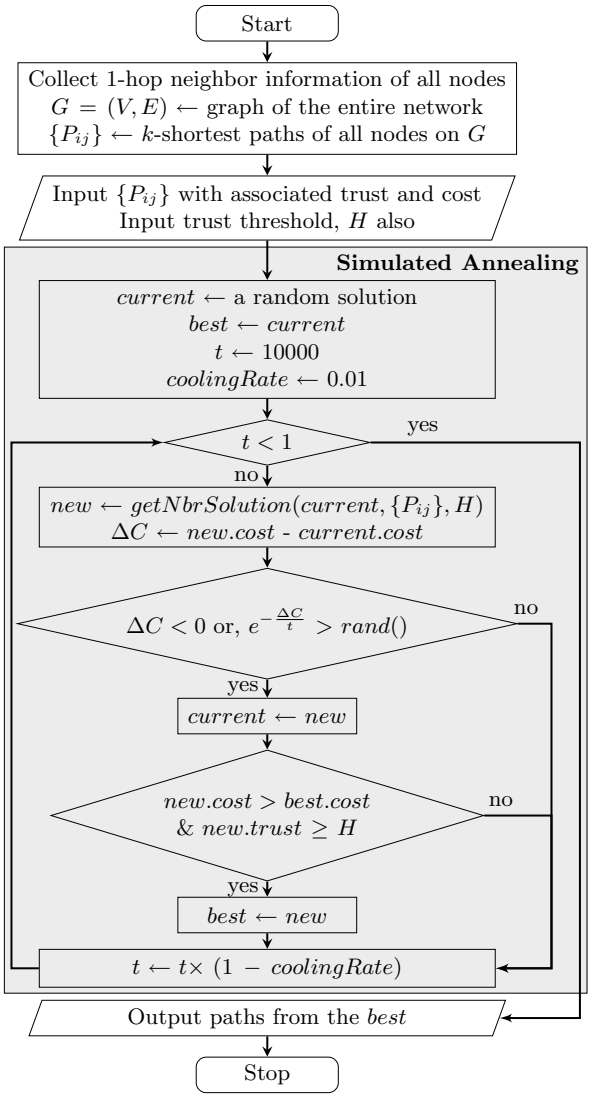


Fig. 5. Flowchart of our proposed solution

against the given threshold, H, as well. If the estimated trustworthiness goes below H, the simulated annealing is invoked again to find a new set of paths that achieves the desired trustworthiness with few expected transmissions.

D. Practical Challenges

We collect 1-hop neighbor information (neighbor identifier and associated link cost in terms of expected number of transmissions) from the Link Estimation Exchange Protocol (LEEP) of TinyOS 2.x which is run as a part of CTP. In order to send 1-hop neighbor information to the base station, a node may need to use more than one packet depending on the size of the provenance field. For example, in a network of 225 nodes, a typical 8-byte provenance field can be exploited to send information about 4 neighbors. In this case, a node having 8 neighbors splits its 1-hop information and embeds into two packets. In our implementation, a node considers another node as a neighbor only if its expected number of transmissions toward that node is above a given threshold. We find the number of neighbors to be at most 10 in our simulations and testbed experiments. By using more sophisticated schemes similar to provenance encoding methods [26], the bit requirements for embedding 1-hop neighbor identifier can be reduced.

After sending 1-hop neighbor information once, a node sends it to the base station only when the ordering (in terms of expected number of transmissions) of its current parent changes with respect to other neighbors. This will trigger a topological change event in the underlying provenance transmission scheme [26] which is designed to embed the ID of the new parent into the provenance field of the subsequent packet. We again exploit the provenance field of the packet to embed the changed 1-hop neighbor information. As long as the process of sending 1-hop neighbor information is executed infrequently (which is the case for most practical networks), the associated transmission costs are amortized over time.

To send updated path information to particular nodes, the base station employs a data dissemination protocol. We configure the transmission power of base station to the highest level to quickly send updates to the network.

VII. Simulation Results

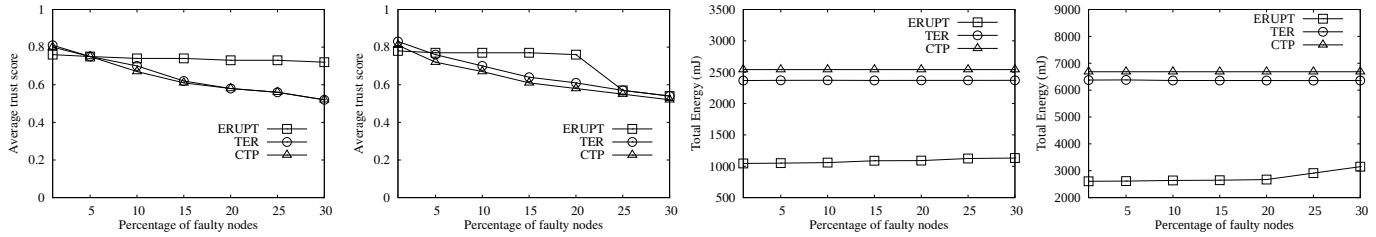
We implement our solution, ERUPT, with a provenance-based trust framework using Java and we interact with the BS via a serial interface. We conduct simulations using TOSSIM for networks of sizes between 49 and 225. For networks with more than 50 nodes, we only collect 1-hop neighbor information from TOSSIM simulations and then apply our solution offline due to the scalability limits of TOSSIM. We compare our solution to the following two approaches:

(1) CTP: CTP lets all nodes sense actively, and does not apply any route optimization based on trust or cost of paths other than what the default routing protocol CTP provides (i.e., builds and maintains a minimum cost tree rooted at the base station).

(2) TER: TER is a variant of the trust and energy-aware routing protocol proposed in [15]. The routing metric used in TER is a weighted sum of the trust, residual energy, and routing cost of a neighboring node. When calculating this metric for each neighbor, a node uses weights 0.1, 0.2, 0.3, and 0.4 for ETX between itself and a neighbor, ETX of the path from that neighbor to the base station, distrust of the neighbor, and energy consumed so far by the neighbor, respectively. To make the comparison fair, we use the same provenance based-trust framework to compute the distrust of nodes in TER.

The base station applications for CTP and TER compute the shortest path (using their routing metric) from each node to the base station based on Dijkstra's algorithm. Finding shortest paths from all nodes to the base station takes $O(|N|^3)$ time for both CTP and TER, where N is the set of all nodes in the network. In contrast, ERUPT uses simulated annealing which starts with a random routing tree and iteratively applies a random change to the current configuration to find a new tree configuration. The method of finding a new configuration can take $O(|N|^3)$ time in the worst case. Assuming a fixed number of iterations, c_{sa} for simulated annealing [31], ERUPT has a run time complexity of $O(c_{sa} \cdot |N|^3)$, i.e., also $O(|N|^3)$.

In our simulations, all active sensor nodes sense and send data every 2.5 s. The sensor data collected at the base station during a single period pertains to the same event. Each event



(a) Average trust in a 7 × 7 grid (b) Average trust in a 10× 10 grid (c) Total energy usage in a 7 × 7 grid (d) Total energy usage in a 10× 10 grid

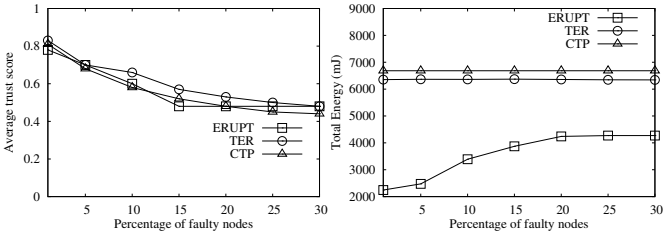
Fig. 6. Average trust score of data items and total energy consumption with varying number of faulty nodes for different grid dimensions

triggers a re-evaluation of the trust scores of all relevant nodes. The weight factors for trust estimate calculation are: $c_1 = 0.6$ and $c_2 = 0.4$. We vary the trustworthiness threshold from 0.2 to 0.6. The results are averaged over 50 runs. Unless otherwise stated, we use the above default values in our simulations.

To introduce faulty nodes, we first allow the trust scores of all the nodes to converge, and then make a different node (that is already part of the active network configuration) faulty every 20 events. Once a node becomes faulty, it remains faulty. The faulty nodes generate false data according to the unusual temperature data values generated by the faulty nodes in the Intel Berkeley testbed.

We consider the following performance metrics: *Average trust score:* The average of trust scores of all the data items per event generated from the nodes that are part of the active network configuration, *Total Energy Consumption:* The amount of energy consumed by all the active nodes due to sensing, transmission, and reception.

To measure total energy consumption, we adopt the energy model proposed by Polastre et al. [32]. The energy consmuption due to the sensing, transmission and reception operations are calculated by multiplying the voltage of the mote with the current consumption and the time spent for the respective operation. We use the value of voltage and current consumption based on the specifications of Tmote sky [33].



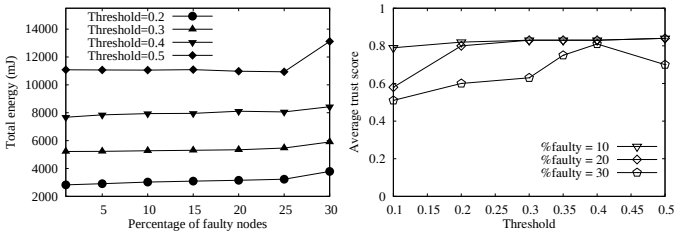
(a) Average trust in a 10× 10 grid (b) Total energy usage in a 10× 10 grid

Fig. 7. Average trust score of data items and total energy consumption with varying numbers of (random) faulty nodes

A. Grid Network Topology

We vary the grid dimensions from 7×7 to 15×15, while keeping the nodes spaced 2 meters apart and the base station in the center of the grid. We use 0.3 as the trustworthiness threshold for this experiment.

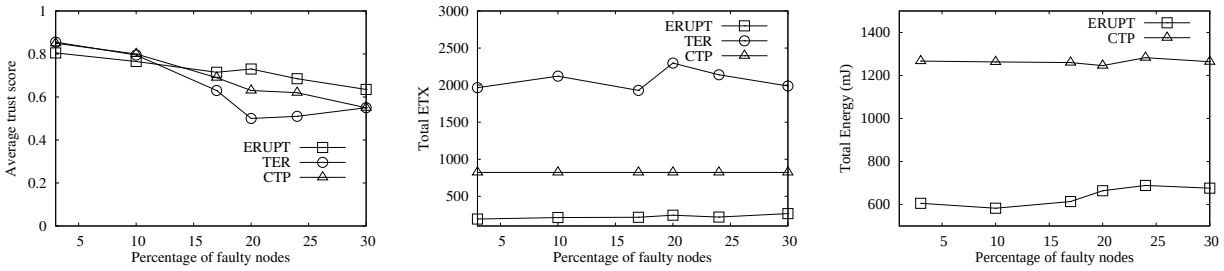
Since sensor nodes located far from the base station (BS) are typically left unattended, and nodes closer to the BS may be easy to manually inspect, we start with an experiment in which distant nodes are more likely to be faulty than closer ones. Fig. 6(a) and 6(b) show the average trust score of the data items per event under varying percentages of faulty nodes. For both grid dimensions, ERUPT achieves higher trustworthiness than TER and CTP. As the percentage of faulty nodes increases, the trustworthiness for both these approaches decreases. Since they keep all the nodes active in the network, unusual data from the faulty nodes lowers the trust scores of all other data items, which eventually degrades the trustworthiness of related nodes. However, TER has a slight edge over CTP since it takes a forwarder node's trustworthiness into consideration. In contrast, ERUPT adjusts the routing tree whenever the overall trustworthiness of the network drops below the given threshold, thereby maintaining higher trustworthiness. We find that there is a number of bottleneck nodes, especially in the larger grid, which are used on the shortest paths of a large number of nodes. When the percentage of faulty nodes is high, these bottleneck nodes become faulty and affect a large number of paths. This limits the availability of alternate good paths, and is why ERUPT shows a slight decrease in the average trust score when the percentage of faulty nodes is around 25 in a 10×10 network.



(a) Total energy usage in a 15× 15 grid (b) Average trust in a 15× 15 grid

Fig. 8. Effect of different threshold and percentage of faulty nodes on average trust score of data items and total energy consumption

Fig. 6(c) and 6(d) show the aggregate energey consumption of the network. Both CTP and TER have a high and almost constant total energy consumption since all the nodes are actively sensing and forwarding data to the BS. TER has lower energy consumption than CTP because it takes residual energy into consideration while choosing the next hop. In contrast, ERUPT can reduce the total energy consumption to a great extent by choosing the appropriate trustworthiness threshold. In case of a higher percentage of faulty nodes, ERUPT shows a slight increase in total energy consumption, since it may require a few more sensing nodes or dissimilar forwarding paths of higher transmission costs to achieve the trustworthiness threshold. Nevertheless, for both grid dimensions, ERUPT reduces the total energy consumption by at



(a) Average trust (b) Total ETX (c) Total energy consumption

Fig. 9. Average trust score of data items, total ETX and energy consumption in a testbed of 30 nodes

least 50% compared with CTP and TER. Similar results are observed for 15 $\times$ 15 grid network and are not included here due to space limit.

In order to understand the effect of faulty "top-level" bottleneck nodes (closest to the BS) more clearly, we perform another set of experiments in which a number of nodes are randomly selected as faulty. We present results for the 10$\times$ 10 grid in Fig. 7. Since a single top-level node affects the trustworthiness of a large number of paths, the availability of alternative trustworthy paths becomes limited, which affects the trustworthiness reported by all the three approaches equally. However, by activating a select set of sensors, ERUPT still achieves lower energy consmuption than CTP and TER by at least 32% and 36%, respectively.

B. Choice of Trustworthiness Threshold

The trustworthiness threshold can be interpreted as the percentage of nodes in a network that are trustworthy, with their forwarding paths dissimilar to some extent. To understand its effect, we measure the average trust score and total energy consumption by varying both the threshold and percentage of faulty nodes in a 15$\times$15 grid network. The results are averaged over 10 runs. Fig. 8(a) shows that the total energy consumption increases with an increasing threshold. The reason is that more sensor nodes must be active to achieve a higher trustworthiness threshold. Furthermore, their forwarding paths are required to be dissimilar which causes additional energy consumption due to transmission overhead.

Fig. 8(b) depicts the average trust score for varying thresholds and percentages of faulty nodes. Here, the threshold is on the x-axis, which gives us interesting insights into the choice of the trustworthiness threshold. We find that when the percentage of faulty nodes is low (5$\sim$15), the average trust score increases as the trustworthiness threshold increases, which is obvious. However, when the percentage of faulty nodes is high, increasing the threshold helps up to a certain point, but after that, the average trustworthiness decreases. Since a large number of nodes are faulty and we are optimizing the transmission cost of the forwarding paths, ERUPT may include some faulty nodes on forwarding paths of low transmission cost in achieving the trustworthiness. This lowers the average trust score in the case of a large number of faulty nodes when the threshold is low. As the threshold increases, ERUPT looks for more nodes with trustworthy forwarding paths, which improves the overall trust score of the network. However, in order to achieve a high trustworthiness threshold, ERUPT has to add more and more nodes and there are no options but to include some faulty nodes. The presence of these faulty nodes affects the trust score of other good nodes and eventually the average trust score of the network starts decreasing. Thus we recommend selecting the trustworthiness threshold considering the reliability requirements of applications and the probability of node failures in the network.

VIII. Testbed Evaluation

We construct a 7.5 m $\times$ 6 m topology consisting of 30 battery-powered TelosB sensors deployed in 6$\times$5 grid fashion in a Purdue University classroom. We select transmission power level 2 to ensure multi-hop communication. The base station node is placed in the middle of the testbed and connected to a laptop via a serial interface that runs the base station application integrated with the provenance based trust framework. To make comparisons fair, we implement ERUPT and the routing algorithm of TER and CTP as part of the base station application. All three protocols use the TinyOS dissemination protocol Drip to propagate updates about forwarding paths to the appropraite nodes.

In our experiments, all active sensor nodes sense and send data every 3.5 s. The trustworthiness threshold is set to 0.3. We use the same fault introduction strategy and performance metrics as in Sec. VII.

Fig. 9(a) shows the average trust score of the data items per event in the presence of varying percentages of faulty nodes. ERUPT achieves higher trustworthiness than TER and CTP in the same way that simulation results show. However, unlike simulation results, TER has lower trustowrthiness than CTP in this case. The reason is that TER manages reception of packets at the base station from only 17 to 24% nodes out of the whole network. With such a small number of reports per event, trustowrthiness of data items becomes low and decreases faster than CTP as the percentage of faulty nodes increases. To investigate this case further, we measure total ETX of the active network configuration chosen by the three aproaches. Fig. 9(b) shows that TER has a very high total ETX with comparison to CTP and ERUPT, indicating highly probable unreachable paths from nodes to the base station.

We measure total energy consumption of the network using the same energy model used in our simulations. Since the packets from a large number of nodes fail to reach the base station, we omit TER results. Fig. 9(c) shows that ERUPT shows a slight increase in total energy consumption in case of a higher percentage of faulty nodes as it includes a few more sensing nodes to achieve the desired trustworthiness threshold.

Nonetheless, ERUPT reduces the total energy consumption by at least 50% compared to CTP.

IX. Conclusions

We have explored the trade-off between the trustworthiness of a wireless sensor network and its associated energy consumption. We perform simulations to reveal that by keeping more sensor nodes active and the forwarding paths (i.e., provenance) of data items generated from these nodes trustworthy and dissimilar, the trustworthiness of the data items can be increased. However, this trustworthiness comes at the expense of more redundant sensors and higher numbers of expected transmissions along the forwarding paths that are chosen to make provenance dissimilar.

We define the overall trustworthiness of the network as the weighted sum of the trustworthiness of all paths and the dissimilarity among the paths. Then, we formulate an optimization problem to find a set of nodes and their corresponding paths that minimize the number of expected transmissions while achieving a certain threshold for the network trustworthiness. Administrators of a particular sensor network are free to choose an appropriate threshold for trustworthiness based on the reliability requirements of their applications. We prove the NP-hardness of our problem and provide a solution based on simulated annealing. Our solution, ERUPT, determines a routing tree rooted at the base station that achieves a required trustworthiness level with low expected number of transmissions. We implement the proposed solution as part of the base station and integrate it with sensor nodes on TOSSIM. We also port our implementation to TelosB motes and conduct experiments on an indoor testbed consisting of 30 motes. The testbed and simulation results show that ERUPT increases the average trustworthiness of the network while reducing the total energy consumption. In our future work, we plan to theoretically analyze the convergence and performance of ERUPT, and demonstrate its effectiveness under intricate adversarial models.

References

[1] "Shell to use CeNSE for clearer picture of oil and gas reservoirs," 2009, http://www.hpl.hp.com/news/2009/oct-dec/cense.html.

[2] A. Doboli and et al., "Cities of the future: Employing wireless sensor networks for efficient decision making in complex environments," SUNYSB, Tech. Rep., April 2008, cEAS Technical Report Nr 831.

[3] "Sensor Andrew at Pennsylvania Smart Infrastructure Incubator," http://www.ices.cmu.edu/psii/sensor-andrew.html.

[4] L. Mo, Y. He, Y. Liu, J. Zhao, S.-J. Tang, X.-Y. Li, and G. Dai, "Canopy closure estimates with greenorbs: Sustainable sensing in the forest," in *Proc. of ACM Sensys*, 2009.

[5] M. S. Lab, "Real-time wireless sensor network platform." http://www.ece.cmu.edu/firefly/projects.html.

[6] X. Liu, "Quality of optical channels in wireless SCADA for offshore wind farms," *IEEE Transactions on Smart Grid*, vol. 3, no. 1, pp. 225–232, 2012.

[7] S. Ganeriwal, "Trustworthy sensor networks," Ph.D. dissertation, University of California, Los Angeles, 2006.

[8] X. Wang, K. Govindan, and P. Mohapatra, "Provenance based information trustworthiness evaluation in multi-hop networks," in *Proc. of IEEE GLOBECOM*, 2010.

[9] H.-S. Lim, Y.-S. Moon, and E. Bertino, "Provenance-based trustworthiness assessment in sensor networks," in *Proc. of International Workshop on Data Management for Sensor Networks*, 2010.

[10] G. Marias, V. Tsetsos, O. Sekkas, and P. Georgiadis, "Performance evaluation of a self-evolving trust building framework," in *Security and Privacy for Emerging Areas in Communication Networks. Workshop of the 1st International Conference*, 2005.

[11] G. Zhan, W. Shi, and J. Deng, "Design and implementation of TARF: A trust-aware routing framework for WSNs," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, pp. 184–197, 2012.

[12] M. Krasniewski, P. Varadharajan, and R. S. Bagchi, "Tibfit: Trust index based fault tolerance for arbitrary data faults," in *Proc. of International Conference on Dependable Systems and Networks (DSN*, 2005, pp. 672–681.

[13] J. Hur, Y. Lee, H. Yoon, D. Choi, and S. Jin, "Trust evaluation model for wireless sensor networks," in *Advanced Communication Technology*, 2005, pp. 491–496.

[14] H. C. Leligou, P. Trakadas, S. Maniatis, P. Karkazis, and T. Zahariadis, "Combining trust with location information for routing in wireless sensor networks," *Wireless Communications and Mobile Computing*, vol. 12, no. 12, pp. 1091–1103, 2012.

[15] L.Gheorgh, R.Rughinis, and N.Tapus, "Trust and energy-aware routing protocol for wireless sensor networks," in *Proc. of the The International Conference on Wireless and Mobile Communications*, 2012.

[16] T. Zahariadis, H. C. Leligou, S. Voliotis, S. Maniatis, P. Trakadas, and P. Karkazis, "An energy and trust-aware routing protocol for large wireless sensor networks," in *Proc. of WSEAS International Conference on Applied Informatics and Communications*, ser. AIC, 2009.

[17] W. Wang, V. Srinivasan, K.-C. Chua, and B. Wang, "Energy-efficient coverage for target detection in wireless sensor networks," in *Proc. of IPSN*, 2007.

[18] Q. Zhao and M. Gurusamy, "Lifetime maximization for connected target coverage in wireless sensor networks," *IEEE/ACM Trans. Netw.*, vol. 16, no. 6, Dec. 2008.

[19] M. Cardei, M. T. Thai, Y. Li, and W. Wu, "Energy-efficient target coverage in wireless sensor networks," in *Proc. of IEEE INFOCOM*, 2005, pp. 1976–1984.

[20] O. Younis and S. Fahmy, "Distributed clustering in ad-hoc sensor networks: A hybrid, energy-efficient approach," in *Proc. of IEEE INFOCOM*, 2004.

[21] "Sensor network deployment at Intel Research," htttp://www.intel-research.edu.

[22] F. Ye, H. Luo, S. Lu, and L. Zhang, "Statistical en-route filtering of injected false data in sensor networks," *Selected Areas in Communications, IEEE Journal on*, vol. 23, no. 4, pp. 839–850, April 2005.

[23] "James san jacinto mountains reserve," http://www.jamesreserve.edu.

[24] "Soil pylon sensor array design and validation," http://research.cens.ucla.edu/projects/2006/Contaminant/Soil_Pylon/default.htm.

[25] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection tree protocol," in *Proc. of ACM Sensys*, 2009.

[26] S. M. I. Alam and S. Fahmy, "A practical approach for provenance transmission in wireless sensor networks," *Ad Hoc Networks*, vol. 16, pp. 28 – 45, 2014.

[27] S. Sultana, G. Ghinita, E. Bertino, and M. Shehab, "A lightweight secure provenance scheme for wireless sensor networks," in *Proc. of IEEE Parallel and Distributed Systems (ICPADS)*, 2012.

[28] H. Kellerer, U. Pferschy, and D. Pisinger, *Knapsack Problems*. Springer, 2004.

[29] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing," *Science, Number 4598, 13 May 1983*, vol. 220, 4598, pp. 671–680, 1983.

[30] D. Eppstein, "Finding the k shortest paths," in *Proc. Symp. Foundations of Computer Science*. IEEE, November 1994, pp. 154–165.

[31] D. Schwab, J. Goulian, and A. Tchechmedjiev, "Worst-case complexity and empirical evaluation of artificial intelligence methods for unsupervised word sense disambiguation," *Int. J. Web Eng. Technol.*, vol. 8, no. 2, pp. 124–153, 2013.

[32] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *Proc. of ACM Sensys*, 2004.

[33] S. Croce, F. Marcelloni, and M. Vecchio, "Reducing power consumption in wireless sensor networks using a novel approach to data aggregation," *Comput. J.*, vol. 51, no. 2, pp. 227–239, Mar. 2008.