Utility-Based Data-Gathering in Wireless Sensor Networks with Unstable Links

Mingming Lu and Jie Wu [⋆]

Department of Computer Science and Engineering
Florida Atlantic University
Boca Raton, FL 33431

Abstract. Traditional utility-based data-gathering models consider the maximization of the gathered information and the minimization of energy consumption in WSNs with reliable channels. In this paper, we extend the model to include retransmissions caused by link failure to improve network utility. The challenge lies in balancing two competing factors: energy loss (and hence utility) through retransmissions and increased reliability (and hence utility) through retransmissions. We adopt a utility-based metric proposed in our previous work [9] and show the NP-hardness of the problem, regardless of the number of source sensors. We design several approximation heuristics for either case and compare their performances through simulation. We also study the impact of retransmissions on the maximization of network utility. Extensive simulations through a customized simulator are conducted to verify our results.

Keywords: Data-gathering, heuristic solution, network utility, routing, stability, wireless sensor networks (WSNs).

1 Introduction

A typical data-gathering wireless sensor network (WSN) consists of one or more sinks which subscribe specific data by expressing interests. Many sensors act as data sources that detect environmental events and push the relevant data to the subscriber sinks. We consider a general many-to-one (one sink and many sensors) WSN with unstable wireless links, where the sink assigns different weights (*benefit*) to different types of events according to their importance. Sensors periodically sense the subscribed events and send data through a data-gathering tree to the sink in each round of communication.

Since the wireless channels are unstable, the reliability of data delivery from sensors to the sink cannot be guaranteed. This unreliability causes data loss and energy waste, and in turn decreases the amount of information collected by the sink and increases the total energy consumption by the sensors. To address the inefficiency caused by the unreliability, we integrate the energy consumption, the instability of wireless channels, and the benefit of sensed data (to the sink) into a single metric-*network utility*-which

[⋆] This work was supported in part by NSF grants ANI 0073736, EIA 0130806, CCR 0329741, CNS 0422762, CNS 0434533, CNS 0531410, and CNS 0626240. Email: {mlu2@, jie@cse.}fau.edu is the same as social welfare [11], which studies the efficient allocation of limited resources in a society to optimize the resource utilization. It is well known that a system is efficient if and only if the system's social welfare is maximized. The social welfare of a system $=$ the system benefit $-$ the system cost. Because the systems we study are the data-gathering WSNs and the purpose of the data-gathering WSNs is to collect sensed data, the system benefit (called *network benefit*) is the total amount of weighted (non-redundant) information gathered by the sink in a round, and the system cost (called *network energy consumption*) is the the total energy consumed by all sensors in a round.

The challenges in maximizing network utility in data-gathering WSNs are as follows. First, the selection of the path from any sensor to the sink depends not only on the network topology (including the energy consumption and the instability of wireless channels), but also the benefit value for each operation (collection of a particular type of data). Second, data from different sensors can share the same path (to the sink) in order to save energy, but this also introduces additional problems as it is vulnerable to link failure since multiple data share the same path/link. Third, there is a question as to whether the number of sensors that have data to send affects the complexity of the problem. Lastly, retransmission can increase the delivery ratio for a path/link, but can also increase transmission delay and energy consumption.

To assess the complex trade-offs one at a time, we assume the availability of a sufficient bandwidth for each channel so that contention for the channel is not an issue. Moreover, we assume that sensors are static and the benefit values for different data are predetermined. Under these assumptions, we can focus on the determination of the optimal reverse broadcast/multicast trees (in terms of maximum network utility).

2 Preliminaries

We first consider the path selection problem, i.e., choosing a path for any sensor to the sink according to the network topology and the benefit value. A network is modeled as an undirected disk graph. Each link (i,j) has two properties: link cost $c_{i,j}$ and link stability $p_{i,j}$. Link cost $c_{i,j}$ is node i's minimal transmission cost to send a packet to node j in a single transmission attempt. Link stability $p_{i,j}$ is the ratio of received packets by node j to transmitted packets by node i in a single transmission attempt. The costs and stabilities of all links compose the topology information of the network.

To illustrate the basic idea of the expected utility, we first consider a single-link route (i,j), where j is the sink. We assume that i has data with benefit v to send. Since the data will be delivered to j with probability $p_{i,j}$, the expected benefit is $v \times p_{i,j}$. Because sensor i will consume energy cost $c_{i,j}$ regardless whether j receives the data or not, the expected utility of this data delivery is:

$$v \times p_{i,j} - c_{i,j}. \tag{1}$$

We observe that the above calculation of the expected utility can extend to the case of a multi-hop route. For example, consider a route $R =< 1, \cdots, i, i+1, \cdots, r >$, where node 1 is the source sensor (the sensor with data to send), and node r is the sink. We can pretend node $r-1$ is also a source sensor; thus, according to Formula (1), the expected utility from node $r-1$ to the sink r is $v \times p_{r-1,r} - c_{r-1,r}$. For simpler presentation, we denote it as the residual expected utility u_{r-1}. Similarly, the expected utility from node $r-2$ to the sink r is $u_{r-2} = u_{r-1} \times p_{r-2,r-1} - c_{r-2,r-1}$. In general, we have

$$u_i = u_{i+1} \times p_{i,i+1} - c_{i,i+1}. \tag{2}$$

By applying Formula (2) recursively, we obtain the expected utility $U = u_1 = u_2 \times p_{1,2} - c_{1,2}$.

We observe that the value of $u_i - u_j$ can be regarded as the *distance* between node i and node j. Therefore, the distance between each two neighboring nodes can be regarded as the weight of the link connecting the two nodes, and hence, the weight information composes the topology information of WSNs with unstable links. Based on this topology information, it is straightforward to apply a Dijkstra-based algorithm to select the best path in terms of the shortest distance. However, the tricky part is that this topology information changes with the change of the benefit value, and therefore, different benefit values cause different topologies. Moreover, the weights of different links are interdependent, which complicates the construction of the data-gathering tree.

3 The Model

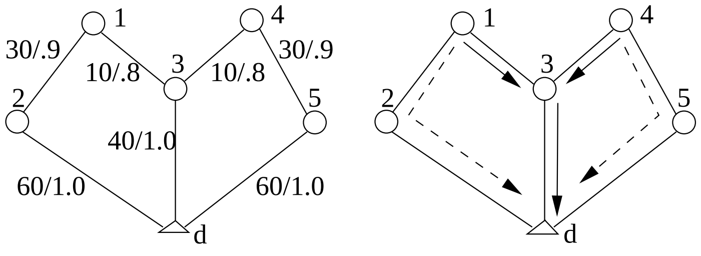
In this work, our main consideration is WSNs, where sensors periodically sense the environment and have data to send in each round (period) of communication. The problem lies in finding a routing scheme to deliver collected data from the designated sensors to the sink so that the expected network utility (in a round) is maximized. We assume that each sensor has only one unit of data to send in each round.

We consider path-sharing to save energy because each packet has a minimum fixed overhead provided by the sequence number, the radio header and CRC, etc. This cost is fixed and independent of the size of the packet payload. Path-sharing can improve transmission efficiency by having proportionally less overhead per useful bit transmitted in the payload. Without loss of generality, we assume that the size of the fixed overhead is 1 and the size of one unit of data is α. Hence, the packet size of transmitting k units of aggregated data is $1 + k\alpha$.

Formally, in our model, a WSN is modeled as an undirected disk graph $(\mathcal{N} \cup \{d\}, E)$, where $\mathcal{N} = \{1, 2, \cdots, N\}$ is the set of sensor nodes, d is the sink, and E is the set of links connecting the sensors. A subset $S \subseteq \mathcal{N}$ consists of all source sensors, each of which has 1 unit of data to send in each round. Let p_i be the delivery ratio from source sensor i to the sink d along the path in a spanning tree T, and c_i be the expected cost of node i in T. $\sum_{i \in S} v \times p_i$ and $\sum_{i \in T} c_i$ are the expected network benefit and the expected network consumption, respectively. Thus, our data-gathering problem can be defined as follows: find a spanning tree T rooted as the sink d that maximizes the expected network utility,

$$\sum_{i \in S} v \times p_i - \sum_{i \in T} c_i, \tag{3}$$

with the constraint that the cost of k units of data transmitted through link $c_{i,j}$ is $(1 + k\alpha)c_{i,j}$.



(a) The topology (b) The data-gathering trees

Fig. 1. An example of utility-based data-gathering tree.

Through a simple example shown in Fig. 1(a), we can see that the optimal data-gathering tree depends not only on the topology information but also on the benefit value and the value of α. There is one sink and two sources (nodes 1 and 4) in Fig. 1(a). In the gathering tree described by the flows in solid lines in Fig. 1(b), the expected utility is $0.8 \times 2v - 10 \times (1+\alpha) \times 2 - 0.8 \times 40 \times (1+2\alpha) = 1.6v - 52 - 84\alpha$ because path $< 1, 3, d >$ and path $< 4, 3, d >$ share link $(3, d)$. In the gathering tree described by the flows in dashed lines in Fig. 1(b), the expected utility of path $< 1, 2, d >$ is $0.9v - 84(1 + \alpha)$. Because path $< 1, 2, d >$ and path $< 4, 5, d >$ are symmetrical and do not share a path, the expected utility is $[0.9v - 84(1+\alpha)] \times 2 = 1.8v - 168(1+\alpha)$. Comparing the expected utilities from the two data-gathering trees, their difference is $0.2v - 116 - 84\alpha$. Note that $\alpha \geq 0$. If $v = 100$, the optimal data-gathering scheme is path-sharing. If $\alpha = 0.1$ and $v = 630$, the optimal scheme is to not share a path.

Both the reverse broadcast tree problem and the reverse multicast tree problem are NP-hard. If all of the links' stabilities are 1, the reverse broadcast tree problem can be reduced to the correlated data gathering problem [5], which has been proven to be NP-hard. Similarly, if all links' stabilities are 1 and the data cost is excluded, i.e., $\alpha = 0$, the reverse multicast tree problem can be reduced to the geometric spanning tree problem, which is also NP-hard. Therefore, our reverse broadcast/multicast tree problem is NP-hard. If we restrict the overhead cost, the data cost α, the link stability, the benefit v, and the source sensor set S, the problem can be reduced to different well-known or solved subproblems. For example, if the overhead cost is not counted, the problem is reduced to the maximum expected utility path tree problem, whose special case that $|S| = 1$ (only one source sensor) has been studied in our prior work [9] and an optimal algorithm with complexity of $O((|E| + |\mathcal{N}|)log|\mathcal{N}|)$ was designed to solve the problem. Furthermore, if all links are reliable, the problem is reduced to the shortest-path tree problem. If link stability is not 1 but the benefit $v \rightarrow \infty$ and $|S| = 1$, it is equal to the most reliable path problem, i.e. find the path with the highest delivery ratio from s to d. On the other hand, if only the overhead cost is considered ($\alpha = 0$), all links' stabilities are 1, and the source set $S = \mathcal{N}$, the problem is the standard broadcast tree problem, which can be solved via the Prime algorithm to construct a minimum spanning tree.

4 The Construction of The Data-Gathering Tree

4.1 Build The Reverse Broadcast Tree

Maximum Expected Utility Path Tree A nave method of constructing the reverse broadcast tree is to build the maximum expected utility path for each sensor, i.e. build the maximum expected utility (MEU) path tree. This heuristic is similar to the shortest-path tree. The difference is that in a MEU path tree, a sensor's distance to the sink depends not only on the link cost, but also on the link stability and the data's benefit to the sink. Different benefit values usually cause different MEU path trees.

Algorithm 1 MEUPT$(\mathcal{N}, d, v)$

```
1: Initialize;
2: while $\mathcal{N} \neq \emptyset$ do
3:     Find the maximum EU sensor $i$ from $\mathcal{N}$;
4:     Remove $i$ from $\mathcal{N}$ to $T$;
5:     For each $i$'s neighbor $j$ not in $T$, Relax$(i, j)$;
```

Relax(i, j)

```
1: if $i$ can increase $j$'s utility then
2:     $u_j \leftarrow u_i \cdot p_{j,i} - \delta \cdot c_{j,i}$;
```

The formal description of this heuristic is given in Algorithm MEUPT. The input of this algorithm is the sensor set $\mathcal{N}$, the sink d, and the benefit v. The link cost $c_{i,j}$ and link stability $p_{i,j}$ for each link (i, j) are also given. Initially, the sink's expected utility is v, if a sensor j can directly communicate with the sink, its expected utility is $v \cdot p_{j,d} - \delta \cdot c_{j,d}$, and all the other sensor's expected utilities are $-\infty$. The reverse broadcast tree T first contains only the sink. In each iteration of the construction phase, the algorithm chooses a link that connects a frontier node (node in T) with a node not in T and has the maximum expected utility, and removes the node from the sensor set. Then, the sensor relaxes its neighbors that are still in the sensor set.

The relaxation consists of two steps. First, the chosen node calculates the expected utility of each neighbor according to the recursive definition of the expected utility (Formula (2)) with a small modification because of the consideration of the data cost and overhead cost. Second, the node compares each neighbor's calculated expected utility with its original expected utility and saves the larger value as the neighbor's new expected utility. This procedure repeats until all sensor nodes are included in T.

Note that in line 2 of the Relax(i, j) function, the coefficient of the cost δ can be either $1+\alpha$ or α. If $\delta = \alpha$, it means the overhead cost is excluded from the energy cost, and hence, there is no need for path-sharing. If $\delta = 1 + \alpha$, it means that data flows do not share paths. Without path-sharing, the MEU path tree is the optimal data gathering tree. Thus, the expected network utilities of the MEU path tree with $\delta = \alpha$ and the MEU path tree with $\delta = 1 + \alpha$ can be used as an upper bound and a lower bound of the optimal data-gathering tree, respectively.

To illustrate the algorithm, we describe the execution of the algorithm on the simple example given in Fig. 1(a). Assume that the benefit is 200 and $\delta = \alpha = 1$. Among d's three neighbors 2, 3, and 5, node 3 has the maximum expected utility 160 while both node 2 and node 5's expected utilities are 140. Thus, link $(3, d)$ is first added into T, and then node 1 and 4's expected utilities are both relaxed through node 3 from $-\infty$ to $160 \times 0.8 - 10 = 118$. Since both node 2 and node 5's expected utilities are larger than node 1 and node 4's expected utilities, node 2 and node 5 are selected earlier than node 1 and 4. Node 2 (5) will try to relax node 1 (4) but fails to improve nodes 1 and 4's expected utilities. Finally, node 1 and 4 will be selected, and the MEU path tree consists of link $(1, 3)$, $(4, 3)$, $(3, d)$, $(2, d)$ and $(5, d)$. If the overhead is not counted, the expected network utility is $160 + 140 \times 2 + 118 \times 2 = 676$. The actual expected network utility of the MEU path tree is $676 - 180 = 496$, where 180 is the overhead of the entire MEU path tree.

Maximum Incremental ENU Link First Although the nave method is simple, it does not take advantage of path-sharing. Therefore, we propose a greedy-based heuristic, MIENULF, to utilize path-sharing. In each iteration of the construction phase, the MIENULF heuristic selects a link that can increase the expected network utility the most. The only difference between the MIENULF heuristic and the MEUPT heuristic is the relaxation part.

Algorithm 2 MIENULF($\mathcal{N}, d, v$)

```
1: The main part is the same as Algorithm MEUPT;

Relax($i, j$)
1: if $i$ can increase $j$'s utility then
2:     $p_j \leftarrow p_i \cdot p_{j,i}$;
3:     $c_j \leftarrow c_i \cdot p_{j,i} + c_{j,i}$;
4:     $u_j \leftarrow v \cdot p_j - \alpha \cdot c_i \cdot p_{j,i} - (1 + \alpha)c_{j,i}$;
```

Besides the expected utility from each sensor, algorithm MIENULF has to memorize the delivery ratio and the expected cost from each sensor to the sink. Therefore, the algorithm maintains two additional variables for each node: p_i and c_i - the current delivery ratio and the current expected cost from node i to the sink along the current path, respectively. Initially, $p_i = 0$ and $c_i = 0$ if $i \neq d$; otherwise, $p_d = 1$ and $c_d = 0$. The reason to maintain p_i and c_i for each node is that the expected cost consists of two parts: the first part is the cost of the new relaxed link, i.e., $(1 + \alpha)c_{j,i}$, which consists of the data cost and the overhead; the second part is the cost shared with other sensors, i.e., $\alpha \cdot c_i \cdot p_{j,i}$, in which the overhead has been included in other nodes's expected cost.

We illustrate algorithm MIENULF by running the example given in Fig. 1(a) and still set $v = 200$ and $\alpha = 1$. After d's relaxation, node 2, 3, and 5's expected utilities are 80, 120, and 80, respectively. Then link $(3, d)$ is selected and node 3 will relax node 1 and node 4, whose expected utilities will change to $200 \times 0.8 - 40 \times 0.8 - 2 \times 10 = 108$.

Therefore, in MIENULF, nodes 1 and 4 are selected before nodes 2 and 5, whose orders are different from the execution of MEUPT, although the final trees are the same.

Spanning Tree-Based Heuristic The spanning tree-based method builds the data-gathering tree by applying the Prime algorithm to construct the minimum spanning tree. The reason is that if we omit the data cost, and hence the utility, the weigh of each link (i, j) is just $-c_{i,j}$, i.e., the negative value of the overhead. Therefore, finding a data-gathering tree that maximizes the weight of the tree is equal to finding the minimal spanning tree. For the example in Fig. 1(a), the minimum spanning tree, which is different from the MEU path tree, consists of links $(3, d)$, $(1, 3)$, $(4, 3)$, $(1, 2)$, and $(4, 5)$. The cost of the overhead is 120, and the expected network utility is $160 + 118 \times 2 + 76.2 \times 2 - 120 = 428.4$.

SLT-Based Approximation Algorithm The SLT-based heuristic utilizes the property that the expected utility of a tree rooted at the sink can be separated into two parts: the expected utility excluding the overhead energy cost and the overhead energy cost. Each part alone can be optimized by a polynomial algorithm. This heuristic is inspired by the shallow light tree (SLT) [5, 8]. The SLT is a spanning tree that has two properties: the cost of the SLT is no more than $1 + \frac{\sqrt{2}}{\gamma}$ times the cost of the minimal spanning tree, and the cost of the path from any node to the sink in the SLT is no more than $1 + \sqrt{2}\gamma$ times the cost of the shortest path, where γ can be any positive constant. But our SLT-based heuristic cannot have the approximation ratio because the metrics for the overhead cost and the modified expected utility are different, unlike those for the MST and the shortest path.

The construction of the SLT-based approximation algorithm is as follows. First, a minimum spanning tree is constructed. Starting from the sink, a depth-first-search of the tree is made. When a node is visited the first time, its expected utility is compared to its maximum expected utility (along the maximum expected utility path). If its expected utility is less than θ ($\theta < 1$) times its maximum expected utility, the link connecting its parent node will be removed, and the maximum expected utility path from the node to the sink is added.

For the example in Fig. 1(a), assume that $v = 200$, $\alpha = 1$, and $\theta = 0.75$. First, the MST is constructed. When searching the MST from the sink in the depth first order, since the expected utilities for nodes 3, 1, and 4 are equal to their maximum expected utilities, links $(3, d)$, $(1, 3)$, and $(3, 4)$ remain the same. However, when nodes 2 and 5 are visited, their expected utilities are $76.2 < 0.75 \times 140$, where 140 is the maximum expected utility. Thus, links $(1, 2)$ and $(4, 5)$ are removed, and links $(2, d)$ and $(5, d)$ are inserted. In this example, the data-gathering tree produced by the SLT-based algorithm is the same as the MEU path tree.

4.2 Build The Reverse Multicast Tree

All the algorithms used in building the reverse broadcast tree can be used to build the reverse multicast tree by pruning the redundant, useless branches in order to connect source sensors to the sink. Besides these algorithms, we propose an algorithm that builds the reverse multicast tree directly.

Maximum Incremental ENU Path First The maximum incremental ENU path first (MIENUPF) approach is similar to the MIENULF heuristic. The difference is that instead of adding one link at each iterative step, the MIENUPF inserts a path that connects an unconnected source to the current T. After the selection of the new branch, each node in the new branch will relax the remaining source sensors. This procedure repeats until all required source sensors are included in T.

The MIENUPF heuristic uses a modified MIEUIF heuristic as a building block. In the modified MIEUIF heuristic, line 2 (the loop termination condition) changes from $\mathcal{N} \neq \emptyset$ to $S \bigcap \mathcal{N} \neq \emptyset$ because we intend to build a reverse multicast tree instead of the reverse broadcast tree. After the selection of the maximum expected utility node i, besides removing i from $\mathcal{N}$, node i should also removed from S if $i \in S$.

Initially, only the expected utility of the sink is set to v, and the expected utilities of all the other nodes are set to $-\infty$. At each iterative step, after the execution of the modified MIEUIF heuristic, the MIENUPF heuristic will select an unconnected source sensor with the maximum expected utility and add the branch that connects the source sensor to T. Although a lot of nodes' expected utilities were updated in the execution of the modified MIEUIF heuristic, only the expected utilities of the nodes on the new branch will be kept.

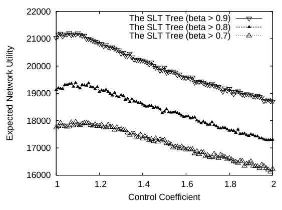
Algorithm 3 MIENUPF($\mathcal{N}, S, d, v$)

```
1: Initialize;
2: while $S \bigcap T \neq \emptyset$ do
3:     MIENULF($\mathcal{N}, S, d, v$);
4:     Find the maximum EU sensor $i$ from $S$;
5:     Remove $i$ from $S$;
6:     Insert into $T$ the branch connecting $i$ to $T$;
7:     Keep the $u_j$ of each node $j$ on the new branch;
```

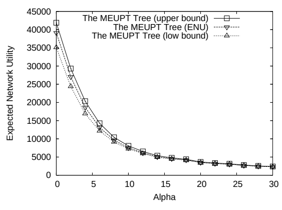
For the example in Fig. 1(a), assume that $v = 200$, $\alpha = 1$. After the first round of the MIENULF, all five nodes have been relaxed, and path $< 1, 3, d >$ (or $< 4, 3, d >$, depending on the tie-breaking rule; here we adopt the smallest node ID) is inserted into T. The expected utilities of nodes 2, 4, and 5 change back to $-\infty$ at the end of this round. In the next round, T starts with the path $< 1, 3, d >$ and link $(3, 4)$ will be inserted in the end.

5 Simulation

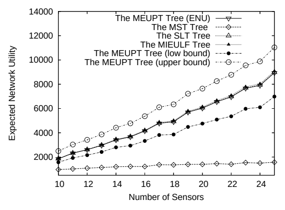
All approaches are simulated on our customized simulator. We empirically study the performance of different heuristics for the reverse broadcast/multicast tree and the effect of various network parameters on the performance of the proposed heuristics. The



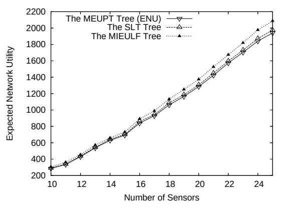
(a) The effect of θ on the STL tree.



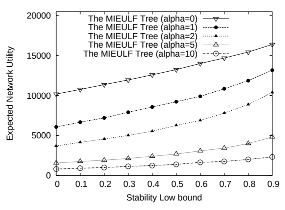
(b) The effect of α, the size of unit data.



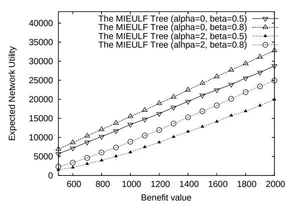
(c) The comparison of the heuristics (RBT).



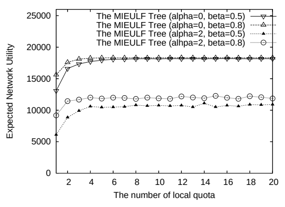
(d) The further comparison of three heuristics.



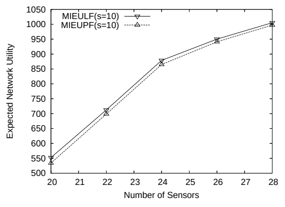
(e) The effect of stability range.



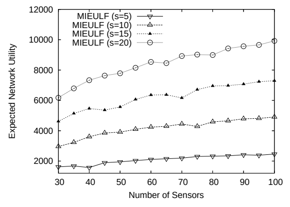
(f) The effect of the benefit value.



(g) The effect of local quota.



(h) The comparison of the heuristics (RMT).



(i) The effect of source sensor number (RMT).

Fig. 2. Simulation results of the heuristics in building the reverse broadcast/multicast tree.

network parameters include the network density n (i.e., node population), the size of a unit data (α), link stability, the value of the benefit v, the source sensor set S, the local quota, and the parameter θ in SLT-based heuristic. The simulation is set up in a $100m \times 100m$ area, where all sensors are homogeneous and can be deployed in this area arbitrarily. The energy cost between any two nodes is proportional to their distance. The stability of each link is randomly generated (uniform distribution) in the range $[\beta, \gamma]$, where $0 \leq \beta \leq \gamma \leq 1$.

5.1 Simulation Results

First, we study the effect of θ on the performance of the STL-based algorithm. θ is the control coefficient in the SLT-based heuristic, in which the value of θ controls whether a node's maximum expected utility path should be inserted into the existing spanning tree or not. As shown in Fig. 2 (a), when $\theta \approx 1.1$, the SLT-based heuristic has the best performance. Because the link stability has direct impact on the SLT-based heuristic, we adopt three different stability low bounds $\beta = 0.7, 0.8,$ and 0.9 as comparisons. The three settings show a similar curve, which means the selection of θ is independent of the link stability.

In section 4, we argue that the expected network utilities of the MEU path tree with $\delta = \alpha$ and the MEU path tree with $\delta = 1 + \alpha$ can be used as the upper bound and the low bound for the optimal data-gathering tree, respectively. Our claim can be verified by the simulation results shown in Fig. 2 (b) and (c). In Fig. 2 (b), we study the effect of α, the size of a unit of data on the performance of the MEUPT algorithm and verify the lower bound and upper bound. The value of α ranges from 0 to 30 with the increment being 2. The simulation results show that as the value of α increases, the expected utility of the MEU path tree, the upper bound, and the lower bound converge. The reason is that as the value of α grows, the effect of the overhead decreases. In the extreme case, as $\alpha \to \infty$, the size of the overhead can be omitted, i.e., $1 + \alpha \to \infty$.

In Fig. 2 (c), we compare the four algorithms in building the reverse broadcast tree (RBT) in the test dimension of network density, and range it from 10 to 25 with the increment being 1. As expected, the network density increases the expected network utility for all the heuristics except the MST-based heuristic, which has the worst performance and is even below the lower bound. The reason is that the MST does not take into account the link stability and benefit issues. The links selected by the MST are close in geometry but may have a low delivery ratio, and hence, cause a lot data losses. The expected network utilities of the other three heuristics are close to each other and hard to compare in Fig. 2 (c). Therefore, we use the lower bound as the base and the expected network utilities of all the three heuristics are subtracted by the base. The result is shown in Fig. 2 (d). From Fig. 2 (d), we can conclude that the MIEULF algorithm has the best performance and the MEUPT algorithm has the worst performance because it does not take into account path-sharing. Since the MIEULF heuristic has the best performance, we will use MIEULF in the following simulations.

In Fig. 2 (e), we simulate the effect of the range of link stability on the performance of the MIEULF heuristic. We increase β from 0 to 0.9 with an incremental step of 0.1. As β increases, the links become stable. We compare the MIEULF trees with different values of α, the size of unit data. The simulation results show that the more stable the links, the higher the expected network utility, and the increment of data size decreases the expected network utility. The simulation results reflect the fact that the expected network utility can be affected from two causes. On one hand, the expected network utility increases with the increment of the link stability. On the other hand, the expected network utility decreases with the increment of the transmission cost.

Fig. 2 (f) shows the results of the simulation on the effect of benefit value. We use the MIEULF heuristic to simulate and adopt four combinations of stability range and the value of α ($\alpha = 0, \beta = 0.5$, $\alpha = 0, \beta = 0.8$, $\alpha = 2, \beta = 0.5$, and $\alpha = 0, \beta = 0.8$). The benefit value varies from 500 to 2000 with an increment of 100. As expected, the increment of the benefit improves the expected network utility, the increment of the data size decreases the expected network utility, and the increment of the stability increases the expected network utility.

We also study the effect of the local quota on the performance. We use the same setting as the previous experiment. The local quota increases from 1 to 20 with an increment of 1. According to the simulation results shown in Fig. 2 (g), the increment of the local quota can increase the expected network utility, but as the number of the local quota reaches 6, the impact of the continuous increment of the local quota becomes less essential. The reason for this is that retransmissions increase the delivery ratio the most in the first several retry attempts.

Finally, we compare the proposed heuristics for constructing the reverse multicast tree and study the effect of the size of the source sensor set on the performance. The simulation results are plotted in Fig. 2 (h) and (i). Because the MIEULF heuristic has the best performance in building the reverse broadcast tree, we adopt the pruning heuristic based on the MIEULF heuristic as the representative pruning method. We set the benefit value to 1000, $\beta = 0.9$, and $\alpha = 2$. Because the expected network utilities of the MIEULF-based heuristic and the MIEUPF heuristic are close, we subtract 5000 from both utilities. Fig. 2 (h) shows that the MIEULF-based pruning heuristic has better performance. Since the MIEULF-based pruning heuristic has better performance, we use it as the method to study the effect of the size of the source sensor set. The other settings are the same. Fig. 2 (i) illustrates that the increment of the number of the source sensors can increase the expected utilities.

6 Related Work

Many existing data-gathering models [5, 10, 12] assume that wireless channels are reliable, or the channels are unstable but the reliability can be achieved through retransmissions. However, wireless communication is unreliable in practice, and 100% reliability is not achievable due to practical issues such as the constraint on the maximum number of retransmissions in link layer technologies. Therefore, we proposed a new-data gathering model that takes this unreliability into account.

Existing link reliability models [1, 3, 7] usually adopted the packet-delivery ratio to define the link reliability, and defined the expected link cost as the link cost divided by the link reliability. This definition is based on the assumption of unbounded retransmissions. Our previous work [9] proposed a more reasonable definition of the expected link costs, which does not allow unlimited retransmissions and involves the interdependence of the stabilities among different links.

Many energy-efficient data-gathering models adopt the reverse broadcast/multicast tree models [12], which utilizes in-network aggregation and fusion to reduce energy consumption. To reduce the complexity, the reverse broadcast tree model [12] assumed the energy consumption of the aggregated data flow is equal to that of a single data flow. A more reasonable model [4, 5, 6] assumed the existence of data correlation so that the energy consumption of an aggregated flow from two flows is less than two single flows and more than one single flow. Our model admits the existence of data correlation, and adopts a utility metric to balance the reliability and energy cost. Chen and Sha [2] also adopted the utility-based model in data-gathering WSNs. They assumed that different data have different levels of importance, and the sink would assign different weights to different types of data according to their importance.

7 Conclusion

In this paper, we study the data-gathering problem in wireless sensor networks from the maximization of the expected network utility point of view by considering resource scarcity and the unstable nature of wireless channels. We model the data-gathering problem as an optimization problem, prove its NP-hardness, propose several heuristics for both the reverse broadcast tree and the reverse multicast tree problems, and use simulation to study the effects of different parameters and to compare the performance of various heuristics. In the future, we will explore the effect of the data redundance on the evaluation of the network benefit and the effect compression technique on reducing the energy consumptions, as well as the effect of signal strength on stability.

References

[1] S. Banerjee and A. Misra. Minimum energy paths for reliable communication in multi-hop wireless networks. In *MobiHoc '02: Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, pages 146–156, 2002.

[2] W. Chen and L. Sha. An energy-aware data-centric generic utility based approach in wireless sensor networks. In *IPSN '04: Proceedings of the third international symposium on Information processing in sensor networks*, pages 215–224, 2004.

[3] D. S. J. D. Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proceedings of ACM MOBICOM'03*, 2003.

[4] R. Cristescu, B. Beferull-Lozano, and M. Vetterli. On network correlated data gathering. In *INFOCOM*, 2004.

[5] R. Cristescu, B. Beferull-Lozano, M. Vetterli, and R. Wattenhofer. Network correlated data gathering with explicit communication: Np-completeness and algorithms. *IEEE/ACM Trans. Netw.*, pages 41–54, 2006.

[6] R. Cristescu and M. Vetterli. Power efficient gathering of correlated data: optimization, np-completeness and heuristics. *SIGMOBILE Mob. Comput. Commun. Rev.*, 7(3):31–32, 2003.

[7] Q. Dong, S. Banerjee, M. Adler, and A. Misra. Minimum energy reliable paths using unreliable wireless links. In *MobiHoc'05: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, pages 449–459, 2005.

[8] S. Khuller, B. Raghavachari, and N. Young. Balancing minimum spanning and shortest path trees. In *SODA '93: Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms*, pages 243–250, 1993.

[9] M. Lu and J. Wu. Social welfare based routing in ad hoc networks. Accepted to appear in Proceedings of ICPP'06, 2006.

[10] H. Luo, J. Luo, and Y. Liu. Energy efficient routing with adaptive data fusion in sensor networks. In *DIALM-POMC '05: Proceedings of the 2005 joint workshop on Foundations of mobile computing*, pages 80–88, New York, NY, USA, 2005. ACM Press.

[11] A. Mas-Collel, W. Whinston, and J. Green. *Microeconomic Theory*. Oxford University Press, 1995.

[12] H.O. Tan and I. Korpeoglu. Power efficient data gathering and aggregation in wireless sensor networks. In *SIGMOD'03*, 2003.