Fault-Tolerant Topology for Energy-Harvesting Heterogeneous Wireless Sensor Networks

Zhiyuan Yin* Fan Li* Meng Shen* Yu Wang†
* Beijing Engineering Research Center of High Volume Language Information Processing and Cloud Computing Applications
School of Computer Science, Beijing Institute of Technology, Beijing, 100081, China.
† Department of Computer Science, University of North Carolina at Charlotte, Charlotte, NC 28223, USA.

***Abstract*—Recent advances in ambient energy-harvesting wireless sensor networks (WSNs) technologies have made it possible to power the network by energy generated from the environment and thereby increase its lifetime. Various energy sources including light, vibration and heat can be harvested by sensor nodes. However, time-varying energy harvesting also bring new design challenging for WSNs. In this paper, we study a fault-tolerant topology design problem for an energy-harvesting heterogeneous WSN, where multiple supernodes with rich resources are used to improve the performance. We first model the network as a directed and weighted space-time graph in which both spacial and temporal information are preserved. We then define the fault-tolerant topology problem which aims to build a sparser time-varying structure from the original space-time graph while maintaining k-connectivity for the fault-tolerant purpose. Six different algorithms are proposed to solve the problem. Simulation results demonstrate that our proposed methods can save up to around 80% costs.**

I. Introduction

Wireless sensor networks (WSNs) have attracted much attention due to its great potential to be used in various applications. As sensor nodes are generally battery-powered devices, the critical aspects to face concern how to reduce the energy consumption of nodes, so that the network lifetime can be extended [1]. There are different ways to save energy or extend the lifetime. In this paper, we consider topology design for *energy-harvesting heterogeneous wireless sensor networks*.

Existing research shows that heterogeneity, when properly deployed, can improve the average delivery rate and extend the lifetime of a large battery-powered network of simple sensors [2], [3]. As shown in Fig. 1, heterogeneous WSNs consist of two types of wireless devices: a large number of randomly deployed wireless sensor nodes and a much smaller number of resource-rich supernodes placed at known locations. Each supernode has two transceivers: one connects to sensor nodes, and the other one connects to other supernodes. Sensor nodes transmit and relay data packets on multiple paths toward multiple potential supernodes. Once a data packet encounters a supernode, it is forwarded using fast supernode-to-supernode communications toward the sink [4].

The work of F. Li is partially supported by the National Natural Science Foundation of China under Grant No. 61370192 and 61432015. The work of Y. Wang is supported in part by the US National Science Foundation under Grant No. CNS-1319915 and CNS-1343355, and the National Natural Science Foundation of China under Grant No. 61428203. F. Li is the corresponding author (Email: fli@bit.edu.cn).

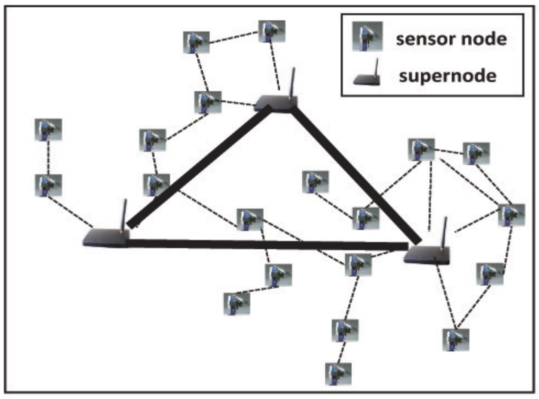
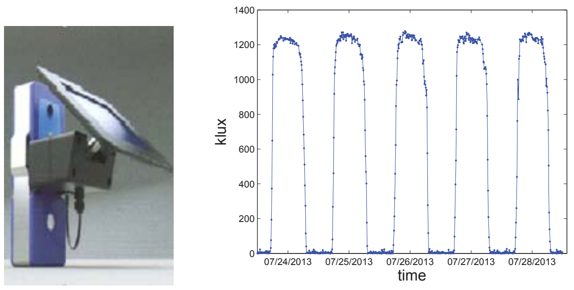


Fig. 1. An example of heterogeneous wireless sensor networks.

Recent advances in ambient energy-harvesting technologies show the possibilities to power WSNs by energy generated from the environment [5]–[7]. Various energy sources including light, vibration or heat can be harvested by sensors. For example, the sensors used in the CitySee [8] are powered by solar cells, as shown in Fig. 2(a). Even though energy-harvesting technology can power the network more perpetually than non-renewable energy sources like batteries, the harvested energy is fundamentally different from battery energy. Usually, it has a limit on the maximum rate at which the energy can be used. Furthermore, the harvested energy availability and the maximum supported rate typically vary with time and space. Such temporal and spatial variations of ambient energy sources and consumption impose a great challenge in protocol design for energy-harvesting WSNs [9]. The change of energy resources not only affects the communication cost, but also results in an evolving network topology. Such dynamic topologies over time domain in energy-harvesting WSNs are often ignored in protocol design [10], [11].

Fortunately, for certain types of energy-harvesting heterogeneous WSNs, we can capture the temporal characteristics of energy resources and dynamic evolution of topologies from historical tracing data. For instance, it is easy to discover



(a) (b)

Fig. 2. (a) A solar energy-harvesting sensor node from CitySee testbed [8] in WuXi, China; (b) The perceived illuminance of one solar energy-harvesting sensor in CitySee testbed from July 24 to 28, 2013.

temporal patterns of energy resources in a solar energy-harvesting system, since the change of solar irradiance over a place follows regular patterns. Fig. 2(b) plots the perceived illuminance of a solar energy-harvesting sensor in CitySee testbed [8] over five days. This implies that it is easy to predict the solar energy resources in an energy-harvesting WSN. Actually, several energy prediction methods have been proposed for different energy-harvesting WSNs [12]–[14].

In this paper, we study the fault tolerant topology design problem for an Energy-harvesting Heterogeneous WSN (EHWSN). EHWSN contains a large number of energy-harvesting sensor nodes and a few resource-rich supernodes. The topology design problem aims to build a sparse time-evolving topology which not only maintains the connectivity from sensor nodes to supernodes but also can survive with $k-1$ node failures. This time-evolving structure then can be used for deciding which subset of sensors need to be awake and assigning their transmission ranges. By doing so, the total energy consumption is minimized. Our major contributions are summarized as follows

- We first define the *fault-tolerant topology problem* in an EHWSN modeled by a space-time graph $\mathcal{G}$, which aims to find a k-connected space-time graph $\mathcal{H}$ (a subgraph of $\mathcal{G}$), such that $\mathcal{H}$'s cost is minimized.
- We propose six heuristics which can significantly reduce the total cost of network topology while preserving the k-connectivity over time.
- Extensive simulations demonstrate the proposed methods can save up to around 80% costs.

The rest of this paper is organized as follows. We first summarize the related work on fault-tolerant topology design in Section II. Then we introduce our space-time graph model and formally define the fault-tolerant topology problem for EHWSN in Section III. Six algorithms are devised to solve the problem in Section IV. The simulation results are exhibited in Section V, finally we conclude this paper in Section VI.

II. Related Work

A great number of studies [4], [15]–[21] have been done on fault-tolerant topology control problem in wireless ad hoc or sensor networks. For example, [15]–[17] study power control algorithms with the objective of minimizing the total power consumption while providing k-vertex connectivity between any two nodes. Most of these proposed algorithms for such optimization problems are centralized. On the other hand, several distributed topology control algorithms [18], [19] are proposed to build sparse topologies which can provide fault tolerance and extend the network lifetime. A survey on fault tolerant topology control can be found at [22].

Fault tolerance design in heterogeneous wireless sensor networks has been studied recently too. Han et al. [20] study two problems of deploying supernodes (or called relay nodes) to provide fault tolerance with higher network connectivity in heterogeneous WSNs. They categorize such problems into two NP-hard placement problems: full fault-tolerant placement problem and partial fault-tolerant placement problem, and then propose a set of approximation algorithms. Cardei et al. [4] consider fault tolerant topology problem with a fixed set of supernodes in heterogeneous WSNs, with the objective of minimizing the total energy consumption while providing k-vertex independent paths from each sensor node to one or more supernodes. Such a topology provides the infrastructure for fault-tolerant data-gathering applications robustness against failures of up to $k-1$ sensors. They propose three algorithms, namely a k-approximation algorithm, a centralized greedy algorithm that minimizes the maximum transmission range of sensor nodes, and a distributed algorithm that incrementally adjusts sensor's transmission range. Recently, there is a more efficient distributed fault-tolerant topology control algorithm [21] proposed for solving the same problem. Our work differs from [4], [21] by considering a different network setting with different objectives, for instance, we focus on the topology design for an energy-harvesting heterogeneous WSN (EHWSN) by considering time-varying nature of energy replenishment and dynamic evolution of the topology. Note that our recent work [9] does consider topology design in EHWSN, however, without fault tolerance guarantee.

III. Fault-tolerant Topology Problem in EHWSN

In this section, we first introduce a weighted space-time graph model to model time-evolving EHWSNs and then formally define the fault-tolerant topology problem.

A. Models and Assumptions

Inspired by [23] [24], we leverage the space-time graph to model the EHWSN, which captures the evolving characteristics in both spacial and temporal aspects. Assume that $V_{sensor} = \{v_1, \cdots, v_n\}$ and $V_{supernode} = \{v_{n+1}, \cdots, v_{n+m}\}$ are the set of all energy-harvesting sensor nodes and the set of all supernodes in the network over a period of time T. Here time is divided into discrete and equal time slots. Let $V = V_{sensor} \bigcup V_{supernode}$ be the whole node set. We assume that supernodes have sufficient power sources, while energy-harvesting sensor nodes may die out at certain time. With the assumptions that co-evolving information of links between individual node and supernodes (*i.e.,* the topology

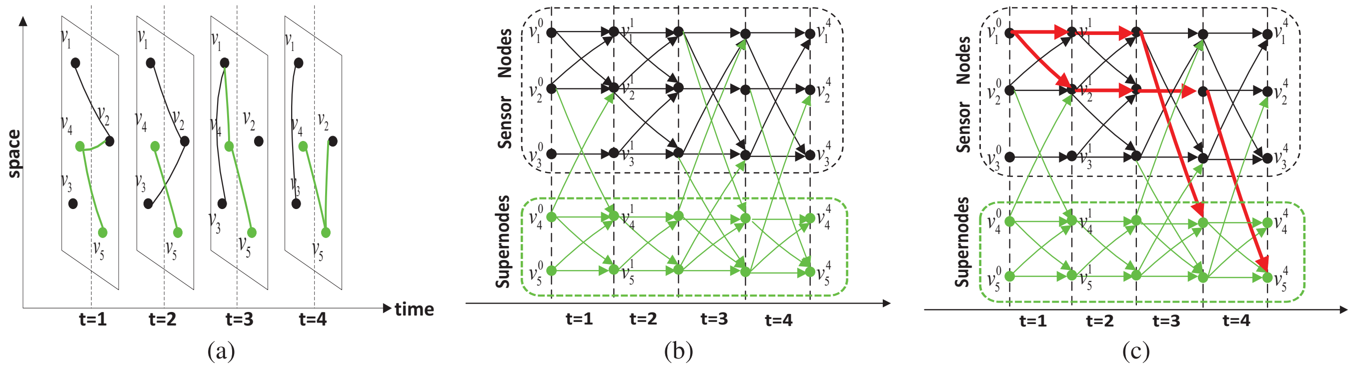


Fig. 3. (a) An example with three energy-harvesting sensors (in black) and two supernodes (in green). (b) The corresponding space-time graph. (c) Two disjoint paths from v_1^0 to supernodes set during time period T.

over time) are known in advance (from predictions of energy resources), a sequence of static graphs can be defined over V to model the interactions among nodes in the EHWSN. Fig. 3(a) illustrates such an example with three energy-harvesting sensors (in black) and two supernodes (in green).

We can then convert this sequence of static graphs into space-time graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, which is a directed graph defined in both spacial and temporal spaces. Fig. 3(b) illustrates the corresponding space-time graph of the same network. In $\mathcal{G}$, T+1 layers of nodes are defined and each layer has $n + m$ nodes, thus the whole vertex set $\mathcal{V} = \{v_j^t | j = 1, \cdots, n+m \ and \ t = 0, \cdots, T\}$ and there are $(n+m)(T+1)$ nodes in total. Two kinds of links (spacial links and temporal links) are added between consecutive layers in the edge set $\mathcal{E}$. A temporal link $\overrightarrow{v_j^{t-1} v_j^t}$ (horizontal links in Fig. 3(b)) connects the same node v_j across consecutive $(t-1)$th and tth layers, which represents the node carrying the message in the tth time slot. A spacial link $\overrightarrow{v_j^{t-1} v_k^t}$ represents forwarding a message from one node v_j to its neighbor v_k in the tth time slot (*i.e.*, v_j and v_k are within each others transmission range in the tth time slot). In Fig. 3, black links are communication links among wireless sensor nodes, while green links are communication links among supernodes. By defining the space-time graph $\mathcal{G}$, any communication operation in the time-evolving network can be simulated on this directed graph. For example, there are two disjoint paths from v_1^0 to the supernode set during the period of T: $v_1^0 \rightarrow v_1^1 \rightarrow v_1^2 \rightarrow v_4^3 \rightarrow v_4^4$ and $v_1^0 \rightarrow v_2^1 \rightarrow v_2^2 \rightarrow v_2^3 \rightarrow v_5^4$, marked in red in Fig. 3(c).

As [25], we assume that the transmission power needs to support a transmission between v_i and v_j (where v_i is a sensor node) is $w(v_i, v_j) = |v_i, v_j|^\alpha$, where $|v_i, v_j|$ is the Euclidean distance between v_i and v_j and α is the path loss exponent (between 2 and 4). Therefore, the link weight in the space-time graph of the link from v_i to v_j is defined as $w(v_i, v_j)$. Note that the supernode always has enough power, thus the link started at any supernode has a weight of zero. Then for any given space-time graph $\mathcal{G}$, we can define its cost as $c(\mathcal{G}) = \sum_{v_k \in \mathcal{G}} \max_{\overrightarrow{v_k v_j} \in \mathcal{G}} w(v_k, v_j)$. Note that $\max_{\overrightarrow{v_k v_j} \in \mathcal{G}} w(v_k, v_j)$ is the cost of a node v_k in $\mathcal{G}$, *i.e.,* its transmission power which can reach all its neighbors in $\mathcal{G}$.

We can also define the k-connectivity of a space-time graph over time as follows:

Definition 1: A space-time graph $\mathcal{G}$ is k-connected over time period of T if and only if for any sensor node $v_j \in \{v_i^0 \mid 1 \leq i \leq n\}$, there are k pairwise vertex disjoint paths from v_j to one or more supernodes in supernodes set $\{v_i^t \mid n < i \leq n+m, 0 < t \leq T\}$ during the period of T.

B. The Problem

Given the weighted space-time graph, we can now define the fault-tolerant topology design problem for EHWSNs with a goal of minimizing the total cost while maintaining k-connectivity over time.

Definition 2: Given a k-connected and weighted space-time graph $\mathcal{G}$ (with n energy-harvesting sensor nodes and m supernodes) and a constant k, the **fault tolerant topology problem** is to find a k-connected space-time graph $\mathcal{H}$, which is a subgraph of $\mathcal{G}$, such that $\mathcal{H}$'s cost $c(\mathcal{H})$ is minimized.

Notice that we assume that all supernodes have enough power resources and can connect to each other via a dedicated transceiver. In addition, for each sensor that we only need to connect it to any one or more of the supernodes. Therefore, we can simply merge all supernodes in the space-time graph as a virtual *root* node. Then, $m = 1$ and $\mathcal{V} = \{v_1, \cdots v_n, root\}$. Links between a sensor node and multiple supernodes can be replaced with a single link between the sensor node and the *root*. The weight of such a link takes the minimal ones from the original graph. Fig. 4(a) shows an example of the merged space-time graph from the original graph in Fig. 3(b). Fig. 4(b) illustrates two disjoint paths from v_0^1 to *root* during the period of T. Now the k-connected space-time graph means for any sensor node there are k-disjoint paths from v_i to the *root*.

IV. Topology Design Algorithms

In this section, we propose a set of heuristics to solve the fault tolerant topology problem in EHWSNs. For all algorithms, we assume that the space-time graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, including n energy-harvesting sensors and the root over time period of T, are given as the input. Let $\tilde{\mathcal{V}}$ and $\tilde{\mathcal{E}}$ denote the

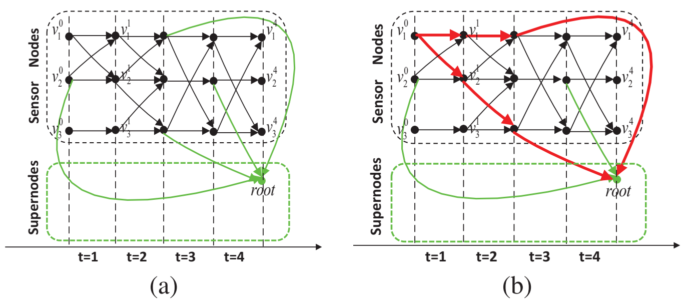


Fig. 4. (a) Example of the merged graph from the space-time graph in Fig. 3(b). (b) Two disjoint paths (marked in red) from v_i^0 to the *root* in the merged graph.

total number of nodes and links in graph $\mathcal{G}$, respectively. Note that $\tilde{\mathcal{V}} = O(nT)$ and $\tilde{\mathcal{E}} = O(n^2T)$. All algorithms will output a new space-time graph $\mathcal{H}$. Based on the output $\mathcal{H}$, we can set the transmission power of each node v_i to $\max_{\overrightarrow{v_iv_j}\in\mathcal{H}} w(v_i, v_j)$.

The first algorithm (Algorithm 1) basically picks the k least cost disjoint paths (*KPS*) between each node in $\mathcal{V}$ and the *root*. We leverage the Suurballe's algorithm [26] (a modified Dijkstra's algorithm) to find the k least-cost path. Note that any k least cost disjoint paths can be used, thus, we refer it as KPS algorithm in Line 3. Assume that KPS is the output from the algorithm for one node k. Then KPS for different nodes might involve completely distinctive links and therefore result in high overall power cost of the entire output structure $\mathcal{H}$. One possible improvement is to reuse as much links as possible in different rounds. For such a purpose, we associate each edge e with a *minset* and a *minsetvalue*, which are denoted by *minset(e)* and *minsetvalue(e)*, respectively, and they are defined as follows.

Definition 3: The *minset* of (v_i, v_j) is the edge set $minset(v_i, v_j) = \{(v_i, v_k) \mid k \neq j, w(v_i, v_k) < w(v_i, v_j)\}$. The *minsetvalue* of this edge is $minsetvalue(v_i, v_j) = \sum_{(v_i,v_k)\in minset(v_i,v_j)} w(v_i, v_k)$.
Similarly, we can define the *maxset* and *maxsetvalue*.

Definition 4: The *maxset* of (v_i, v_j) is the edge set $maxset(v_i, v_j) = \{(v_i, v_k) \mid k \neq j, w(v_i, v_k) > w(v_i, v_j)\}$. The *maxsetvalue* of this edge is $maxSetValue(v_i, v_j) = \sum_{(v_i,v_k)\in maxset(v_i,v_j)} w(v_i, v_k)$.
Algorithm 1 (denoted by UKPS) then updates the weights of the links in the *minset* and the *maxset* of the link belongs to *KPS* (Lines 6-10). Such operations of updating the link weight in *minset* and *maxset* would make it prefer to choose the links already selected in previous rounds. The time complexity of UKPS is $O(nk(\tilde{\mathcal{E}} + \tilde{\mathcal{V}} log\tilde{\mathcal{V}}))$ since the algorithm ends within n rounds and the modified Dijkstra's algorithm runs k times in each round. A simplified version of UKPS can be derived by removing the updates of link weights (Lines 6-10), which is referred to as UKPS-S.

The next two algorithms are greedy algorithms which either delete edges from the original graph or add edges from an empty graph. GrdDelEdge (Algorithm 2) greedily deletes an

Algorithm 1 Union of k-Disjoint Paths (UKPS)

```
Input: The original space-time graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.
Output: the new space-time graph $\mathcal{H}$.
 1: $V = \{v_i^0 \mid 1 \leq i \leq n\}$;
 2: for each $v \in V$ do
 3:   Apply KPS algorithm to find $KPS$ from $v$ to root.
 4:   for each edge $e \in KPS$ do
 5:     $\mathcal{H} \leftarrow \mathcal{H} \cup e.minset$
 6:     for each edge $e' \in minset(e)$ do
 7:       $w(e') \leftarrow 0$;
 8:     for each edge $e' \in maxset(e)$ do
 9:       $w(e') \leftarrow w(e') - w(e)$;
10:     $w(e) \leftarrow 0$;
11: return $\mathcal{H}$
```

Algorithm 2 Greedy Algorithm Delete Edge (GrdDelEdge)

```
Input: the original space-time graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.
Output: the new space-time graph $\mathcal{H}$.
 1: $\mathcal{H} \leftarrow \mathcal{G}$;
 2: Sort all remaining edges in $\mathcal{E}$ in descending order of its
    $maxsetvalue$;
 3: for each edge $e \in \mathcal{E}$ do
 4:   if $(\mathcal{H} - maxset(e))$ is $k$-connected then
 5:     $\mathcal{H} \leftarrow \mathcal{H} - maxset(e) - \{e\}$;
 6: return $\mathcal{H}$
```

Algorithm 3 Greedy Algorithm Add Edge (GrdAddEdge)

```
Input: the original space-time graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.
Output: the new space-time graph $\mathcal{H}$.
 1: $\mathcal{H} \leftarrow NULL$;
 2: Sort all remaining edges in $\mathcal{E}$ in ascending order of its
    $minsetvalue$;
 3: for each edge $e \in \mathcal{E}$ do
 4:   $\mathcal{H} \leftarrow \mathcal{H} \cup minset(e) \cup \{e\}$;
 5:   if $\mathcal{H}$ is $k$-connected then
 6:     break;
 7: return $\mathcal{H}$
```

edge and its *edge.maxset* with the largest *maxsetvalue* if such operation does not break the k-connectivity of the graph over time. On the other hand, GrdAddEdge starts with a graph with only nodes in the first layer and the root. Then it greedily adds in more edges and nodes until the k-connectivity $\mathcal{H}$ achieved. During the process, it selects the edge with the smallest *maxsetvalue* first. Detailed algorithm is given in Algorithms 3. For these two algorithms, we can also add or delete edges based on their weights, *i.e.*, replacing *minsetvalue* or *maxsetvalue* with the edge's weight. Such solutions are referred as GrdDelEdge-W and GrdAddEdge-W, respectively. By using network flow techniques [27], a query on whether two vertices are k-connected in the graph $\mathcal{G} = \{\tilde{\mathcal{V}}, \tilde{\mathcal{E}}\}$ can be solved in $O(\tilde{\mathcal{V}} + \tilde{\mathcal{E}})$ for any fixed k. Therefore, the time complexity of GrdDelEdge/GrdAddEdge is $O(\tilde{\mathcal{E}}(\tilde{\mathcal{V}} + \tilde{\mathcal{E}}))$.

V. Simulation Results

To evaluate our proposed algorithms for EHWSNs, we conduct extensive simulations over randomly generated time-evolving networks. The algorithms used for comparison are listed as follows:

- **UKPS (or -S):** Algorithm based on union of k-disjoint paths (or without the link updates);
- **GrdDelEdge (or -W):** Greedy algorithm deleting edge based on value of its *maxset* (or weight);
- **GrdAddEdge (or -W):** Greedy algorithm adding edge based on value of its *minset* (or weight).

In all simulations, we take the following two metrics as the performance measurements.

- **Total cost:** the summation of total power consumption of each node in $\mathcal{H}$ over time, denoted as $c(\mathcal{H})$.
- **Total number of nodes:** the total number of nodes in $\mathcal{H}$, *i.e.*, the total number of nodes with the remaining edges in $\mathcal{H}$, denoted as $|\mathcal{H}|$.

For both metrics, we report the ratios between those of the constructed topology $\mathcal{H}$ and those of the original network $\mathcal{G}$. These ratios illustrate how much saving is achieved by the proposed algorithm, compared with the original network without topology design.

We test all algorithms on randomly generated networks. A sequence of static random graphs with $n + m$ nodes (*i.e.*, n sensor nodes and m supernodes) over $T = 10$ time slots deployed in $100m \times 100m$ area. For each static snapshot, the transmission range is randomly set within the range from $23m$ to $36m$ (different at each time slot). A directed link from v_i to v_j is added if v_j is within v_i's transmission range. The path loss exponent α is set to 2, as [21]. The network is discarded if it is not connected with its initial setting. For each setting, we generate 100 random EHWSNs and report average performances of our proposed algorithms.

A. Results with Different Values of k

In the first set of simulations, our algorithms are performed with different fault tolerant requirements (values of k ranging from 1 to 3) for random networks with $n = 20$ nodes and $m = 3$ supernodes. Fig. 5 shows the ratios of total cost and node numbers between $\mathcal{H}$ and those of $\mathcal{G}$. The ratio indicates how much saving is achieved by the proposed algorithms compared with the original network without topology design. Obviously, for each algorithm, the lower k's value is, the fewer total cost and fewer number of nodes are needed for k-connectivity. Overall, UKPS performs the best among all algorithms, because it updates edge weights with *minset* and *maxset* to encourage the reuse of edges already selected by least cost disjoint paths. As the growth of the k, more nodes and total cost are needed. In addition, GrdAddEdge and GrdAddEdge-W perform poorly in terms of cost rate and node number rate. This is mainly due to that these algorithms both select edges purely by *minsetvalues* and their weights naively, without considering their contribution in constructing the k-connected graph. On the other hand, GrdDelEdge and GrdAddEdge perform better than GrdDelEdge-W and GrdAddEdge-W. This is mainly due to that GrdDelEdge/GrdAddEdge deletes/adds a edge set based on its *maxsetvalue/minsetvalue* while GrdDelEdge-W/GrdAddEdge-W only considers a single edge weight.

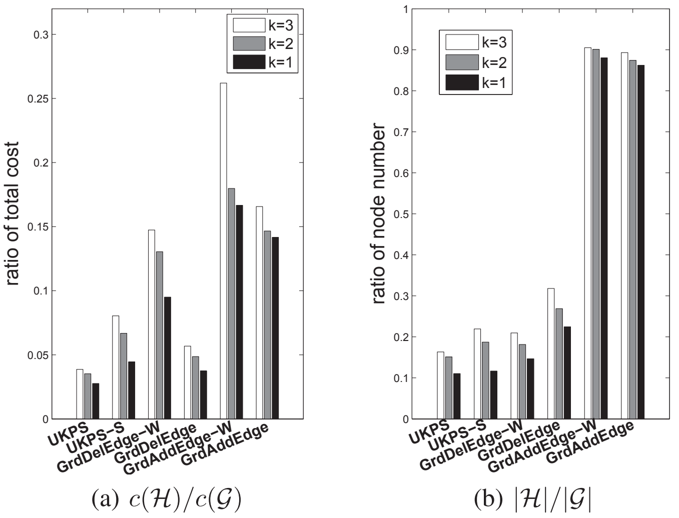


Fig. 5. Results on random networks with different values of k ($n = 20$, $m = 3$).

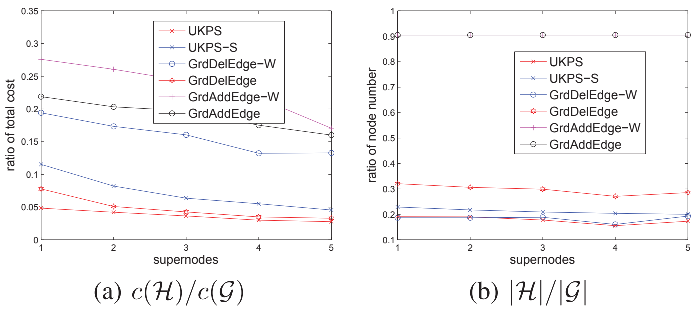
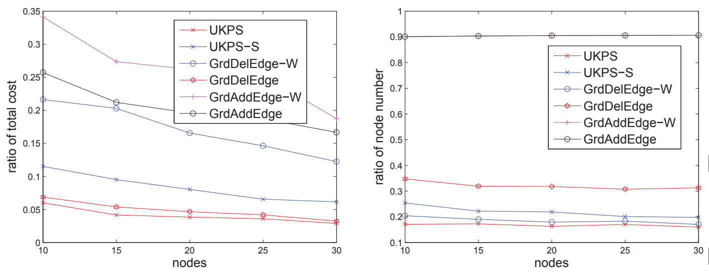


Fig. 6. Results on random networks with different numbers of supernodes m ($n = 20$, $k = 3$).

B. Results with Different Numbers of Supernodes

In the second set of simulations, we fixed the fault tolerant requirement $k = 3$ and the number of sensors to 20, but the number of supernodes, *i.e., m*, varies from 1 to 5. From the results illustrated in Fig. 6, as the number of supernodes increases, the cost rate and the node number rate decrease slightly. More supernodes can indeed help with connectivity. Note that how to place supernodes is outside the scope of this paper, however it is an interesting problem to study. UKPS also performs the best among all the algorithms, and the conclusion are in consistent with the prior. In most cases, UKPS can save more than 80% number of nodes involved in maintaining connectivity over time and around 95% total costs.



(a) $c(\mathcal{H})/c(\mathcal{G})$ (b) $|\mathcal{H}|/|\mathcal{G}|$

Fig. 7. Results on random networks with different numbers of nodes n ($m = 3$, $k = 3$).

C. Results with Different Network Sizes

In the third set of simulations, we use random networks with different numbers of sensor nodes (n ranging from 10 to 30) to test the scalability of the proposed algorithms, while $m = 3$ and $k = 3$. Fig. 7 plots the cost rate and node number rate generated by each algorithm with different network sizes. It is clear that the larger the network is, the lower the cost rate and node number rate are. Other conclusions remain the same as those of previous sets.

VI. Conclusion

In this paper, we study the fault-tolerant topology design problem for EHWSNs. We first model such an EHWSN as a directed and weighted space-time graph, in which both spacial and temporal information are preserved. We then define the fault-tolerant topology problem which aims to build a sparser structure (also a space-time graph) from the original space-time graph, while maintaining k-connectivity over time. We propose six different algorithms to solve such problem. Simulation results demonstrate the efficiency of the proposed methods. As the future work, our methods can be further extended in a distributed manner to enhance their scalability for large-scale EHWSNs.

References

[1] G. Anastasi, M. Conti, M. Di Francesco, and A. Passarella, "Energy conservation in wireless sensor networks: A survey," *Ad Hoc Networks*, vol. 7, no. 3, pp. 537–568, 2009.

[2] M. Yarvis, N. Kushalnagar, H. Singh, A. Rangarajan, Y. Liu, and S. Singh, "Exploiting heterogeneity in sensor networks," in *Proc. of INFOCOM*, 2005.

[3] Y. Qu, K. Xu, J. Liu, W. Chen, "Towards a Practical Energy Conservation Mechanism with Assistance of Resourceful Mules," in *IEEE Internet of Things Journal*, DOI:10.1109/JIOT.2014.2371056, 2014.

[4] M. Cardei, S. Yang, and J. Wu, "Algorithms for fault-tolerant topology in heterogeneous wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 4, pp. 545–558, 2008.

[5] K. Lin, J. Yu, J. Hsu, S. Zahedi, D. Lee, J. Friedman, A. Kansal, V. Raghunathan, and M. Srivastava, "Heliomote: enabling long-lived sensor networks through solar energy harvesting," in *Proc. of ACM SenSys*, 2005.

[6] C. Park and P. H. Chou, "Ambimax: Autonomous energy harvesting platform for multi-supply wireless sensor nodes," in *Proc. of IEEE SECON*, 2006.

[7] G. K. Chen, M. Fojtik, D. Kim, D. Fick, J. Park, M. Seok, M.-T. Chen, Z. Foo, D. Sylvester, and D. Blaauw, "Millimeter-scale nearly perpetual sensor system with stacked battery and solar cells," in *Proc. of IEEE ISSCC*, 2010.

[8] Y. Liu, X. Mao, Y. He, K. Liu, W. Gong, and J. Wang, "CitySee: not only a wireless sensor network," *IEEE Network*, vol. 27, no. 5, pp. 42-47, 2013.

[9] F. Li, S. Chen, S. Tang, X. He, and Y. Wang, "Efficient topology design in time-evolving and energy-harvesting wireless sensor networks," in *Prof. of IEEE MASS*, 2013.

[10] F. Li, S. Chen, M. Huang, Y. Zhu, C. Zhang, and Y. Wang, "Reliable topology design in time-evolving delay-tolerant networks with unreliable links," *IEEE Transactions on Mobile Computing*, vol. PP, no. 99, DOI: 10.1109/TMC.2014.2345392, 2014.

[11] M. Huang, S. Chen, Y. Zhu, and Y. Wang, "Topology control for time-evolving and predictable delay-tolerant networks," *IEEE Transactions on Computers*, vol. 62, no. 11, pp. 2308 - 2321, 2013.

[12] J. R. Piorno, C. Bergonzini, D. Atienza, and T. S. Rosing, "Prediction and management in energy harvested wireless sensor nodes," in *Proc. of IEEE Wireless VITAE*, 2009.

[13] C. Bergonzini, D. Brunelli, and L. Benini, "Algorithms for harvested energy prediction in batteryless wireless sensor networks," in *Proc. of IEEE IWASI*, 2009.

[14] J. Lu, S. Liu, Q. Wu, and Q. Qiu, "Accurate modeling and prediction of energy availability in energy harvesting real-time embedded systems," in *Proc. of IEEE International Green Computing Conference*, 2010.

[15] M. Hajiaghayi, N. Immorlica, and V. S. Mirrokni, "Power optimization in fault-tolerant topology control algorithms for wireless multi-hop networks," in *Proc. of ACM Mobicom*, 2003.

[16] X. Jia, D. Kim, S. Makki, P.-J. Wan, and C.-W. Yi, "Power assignment for k-connectivity in wireless ad hoc networks," in *Proc. of IEEE INFOCOM*, 2005.

[17] E. L. Lloyd, R. Liu, M. V. Marathe, R. Ramanathan, and S. Ravi, "Algorithmic aspects of topology control problems for ad hoc networks," *Mobile Networks and Applications*, vol. 10, no. 1-2, pp. 19–34, 2005.

[18] Y. Wang, L. Cao, T. A. Dahlberg, F. Li, and X. Shi, "Self-organizing fault tolerant topology control in large-scale three-dimensional wireless networks," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 4, no. 3, pp. 19:1-19:21, 2009.

[19] X. Wang, M. Sheng, M. Liu, D. Zhai, and Y. Zhang, "RESP: A k-connected residual energy-aware topology control algorithm for ad hoc networks," in *Proc. of IEEE WCNC*, 2013.

[20] X. Han, X. Cao, E. L. Lloyd, and C.-C. Shen, "Fault-tolerant relay node placement in heterogeneous wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 9, no. 5, pp. 643–656, 2010.

[21] H. Bagci, I. Korpeoglu, and A. YAZICI, "A distributed fault-tolerant topology control algorithm for heterogeneous wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, to appear.

[22] Y. Wang, "Fault tolerant topology design for ad hoc and sensor networks," in *Handbook of Research on Wireless Security*, edited by Y. Zhang, et al, Idea Group Reference, March 2008.

[23] S. Merugu, M. H. Ammar, and E. W. Zegura, "Routing in space and time in networks with predictable mobility," GaTech, Tech. Rep., CC-04-07, 2004.

[24] C. Liu and J. Wu, "Routing in a cyclic mobispace," in *Proc. of ACM MobiHoc*, 2008.

[25] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, pp. 660–670, 2002.

[26] J.W. Suurballe, "Disjoint paths in a network," *N*etworks, vol. 4, no. 2, pp. 125–145, 1974.

[27] S. Even and R. E. Tarjan, "Network flow and testing graph connectivity," *SIAM journal on computing*, vol. 4, no. 4, pp. 507–518, 1975.