Fault-tolerant Relay Node Placement in Heterogeneous Wireless Sensor Networks

Xiaofeng Han*◇ Xiang Cao*◇ Errol L. Lloyd*◇ Chien-Chung Shen*◇

***Abstract*— Existing work on placing additional relay nodes in wireless sensor networks to improve network connectivity typically assumes homogeneous wireless sensor nodes with an identical transmission radius. In contrast, this paper addresses the problem of deploying relay nodes to provide fault-tolerance with higher network connectivity in *heterogeneous* wireless sensor networks, where sensor nodes possess different transmission radii. Depending on the level of desired fault-tolerance, such problems can be categorized as: (1) *full* fault-tolerance relay node placement, which aims to deploy a minimum number of relay nodes to establish k $(k \geq 1)$ vertex-disjoint paths between every pair of sensor and/or relay nodes; (2) *partial* fault-tolerance relay node placement, which aims to deploy a minimum number of relay nodes to establish k $(k \geq 1)$ vertex-disjoint paths only between every pair of sensor nodes. Due to the different transmission radii of sensor nodes, these problems are further complicated by the existence of two different kinds of communication paths in heterogeneous wireless sensor networks, namely *two-way* paths, along which wireless communications exist in both directions; and *one-way* paths, along which wireless communications exist in only one direction. Assuming that sensor nodes have different transmission radii, while relay nodes use the same transmission radius, this paper comprehensively analyzes the range of problems introduced by the different levels of fault-tolerance (full or partial) coupled with the different types of path (one-way or two-way). Since each of these problems is NP-hard, we develop $O(\sigma k^2)$-approximation algorithms for both one-way and two-way partial fault-tolerance relay node placement, as well as $O(\sigma k^3)$-approximation algorithms for both one-way and two-way full fault-tolerance relay node placement (σ is the best performance ratio of existing approximation algorithms for finding a minimum k-vertex connected spanning graph). To facilitate the applications in higher dimensions, we also extend these algorithms and derive their performance ratios in d-dimensional heterogeneous wireless sensor networks $(d \geq 3)$. Finally, heuristic implementations of these algorithms are evaluated via simulations.**

I. Introduction

Heterogeneous wireless sensor networks (H-WSNs) are composed of a large number of wireless devices equipped with different communication and computing capabilities. In comparison with *homogeneous* wireless sensor networks, where all of the devices possess the same communication and computing capability, H-WSNs allow for a variety of operating environments, and hence are useful for many practical applications.

However, in real applications, unpredictable events, such as battery depletion and environmental impairment, may cause these wireless devices to fail, partitioning the network and disrupting normal network functions. Therefore, fault tolerance becomes a critical factor for the successful deployment of wireless sensor networks. One approach to achieve fault tolerance in wireless sensor networks is to deploy a small number of additional *relay nodes* to provide k $(k \geq 1)$ *vertex-disjoint* paths between every pair of functioning devices (including sensors, data sinks, and other wireless equipments, all termed *target nodes* in this paper) so that the network can survive the failure of fewer than k nodes. This problem is known as *relay node placement* in the literature [1][2][3][4][5][6][7].

Most of the existing work considers relay node placement in the context of homogeneous wireless sensor networks where both target nodes and relay nodes use an identical transmission radius. Before reviewing these results, we first introduce a key definition of approximation algorithm that will be used in this paper.

Definition 1.1: [*Approximation Algorithm*][8]: An algorithm solving a minimization problem is a p-approximation algorithm (or has a performance ratio p), if the solution provided by the algorithm is no more than p times the optimal solution.

Relay node placement has been well studied for the case of $k = 1$, *i.e.,* using a minimum number of relay nodes to bridge a partitioned network. For instance, Lin and Xue [1] proved this problem to be NP-hard, and proposed a minimum spanning tree (MST) based 5-approximation algorithm. Chen *et al.* [2] showed that the performance ratio of the algorithm described in [1] is actually 4, and they also proposed a 3-approximation algorithm for this problem. In [3], Cheng *et al.* proposed a faster 3-approximation algorithm and a randomized algorithm with a performance ratio of 2.5.

Recently, work on relay node placement has also been done for the general case of $k \geq 2$. For example, Bredin *et al.* [5] studied the *full fault-tolerance relay node placement (FFRP)*, which aims to deploy a minimum number of relay nodes to create a full k-vertex connected network such that the resulting network contains k vertex-disjoint paths between every pair of *target and/or relay* nodes. Figure 1(a) gives an example of a full 2-vertex connected network. The authors of [5] presented a $\sigma(9k^4 + 36(k^3 + k^2))$-approximation algorithm for FFRP. Here σ is the best performance ratio of existing approximation

*The research of these authors is supported in part by the National Science Foundation under grant CNS-0347460.

*Prepared through collaborative participation in the Communications and Networks Consortium sponsored by the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011. The U. S. Government is authorized to reproduce and distribute reprints for Government purposes not withstanding any copyright notation thereon.

◇Department of Computer and Information Sciences, University of Delaware, Newark, DE 19716, U.S.A. Email: {han, cao, elloyd, cshen}@cis.udel.edu.

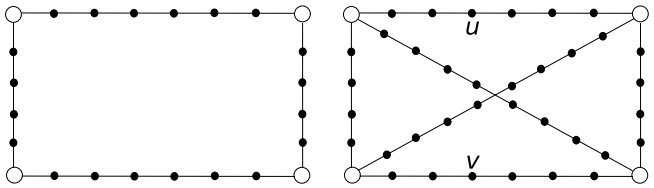


Fig. 1. Hollow circles stand for target nodes, and solid circles stand for relay nodes. (a) A full 2-vertex connected network which contains 2 vertex-disjoint paths between every pair of target and/or relay nodes. (b) A partial 3-vertex connected network. In this network, there exist 3 vertex-disjoint paths between every pair of target nodes, while for some pairs of relay nodes, like node u and node v, there exist only 2 vertex-disjoint paths between them.

algorithms for finding a *minimum k-vertex connected spanning graph* (see Section II-D for the definition). In [6], Kashyap *et al.* proposed approximation algorithms for deploying relay nodes to create partial k-edge (vertex) connected networks for the cluster heads in wireless sensor networks. The problem defined in [6] is termed *partial fault-tolerance relay node placement (PFRP)* where the k edge (vertex)-disjoint paths are only guaranteed between every pair of target nodes. Figure 1(b) shows an example of a partial 3-vertex connected network. The authors of [6] proved that the performance ratios of their algorithms are 10 when $k = 2$. However, the performance ratios of their algorithms for the case of $k > 2$ remain open.

In addition to the above work assuming that both target nodes and relay nodes use an identical transmission radius, relay node placement has also been studied in *two-tiered* homogeneous wireless sensor networks under the assumption that all of the relay nodes use transmission radius R and can communicate with either relay or target nodes, while all of the target nodes use transmission radius r and only communicate with relay nodes. Relay node placement in two-tiered homogeneous wireless sensor networks aims to create a backbone containing only relay nodes to provide k vertex-disjoint paths between every pair of target nodes. For this particular problem, Tang *et al.* [7] proposed 4.5-approximation algorithms for the cases of $k = 1$ and $k = 2$, provided $R \geq 4r$. For the more general situation of $R \geq r$, Lloyd and Xue [4][1] recently gave a $(5+\varepsilon)$-approximation algorithm for the case of $k = 1$.

To the best of our knowledge, this paper is the first effort to address relay node placement in the context of $heterogeneous$ wireless sensor networks (H-WSNs). In H-WSNs, target nodes may have different transmission radii, while all of the relay nodes use an identical transmission radius. The different transmission radii of target nodes introduce *asymmetric communication links* between neighboring nodes, which raise two nontrivial issues. First, asymmetric links result in the existence of two kinds of paths in H-WSNs, namely *one-way* paths and *two-way* paths. For a two-way path, wireless communications exist in both directions, while for a one-way path, wireless communications exist only in one direction. Second, since no constraints are imposed on the relation between the transmission radius of relay nodes and the transmission radii of target nodes, as discussed further in Section II-B, placing relay nodes in the network to connect two particular target nodes becomes much more complicated.

Given a set of target nodes V in the context of H-WSNs, and a desired connectivity level k $(k \geq 1)$ (assume that the cardinality of V is larger than k), this paper systematically addresses the following problems.

• *One-way / Two-way partial fault-tolerance relay node placement (One-way / Two-way PFRP)*. We seek to deploy a minimum number of relay nodes to form a one-way / two-way partial k-vertex connected network for V, such that the resulting network contains k vertex-disjoint one-way / two-way paths from any target node to any other target node.

• *One-way / Two-way full fault-tolerance relay node placement (One-way / Two-way FFRP)*. We seek to deploy a minimum number of relay nodes to form a one-way / two-way full k-vertex connected network for V, such that the resulting network contains k vertex-disjoint one-way / two-way paths from any node to any other node.

The contributions of this paper are summarized as follows: (1) we give $O(\sigma k^2)$-approximation algorithms for both One-way PFRP and Two-way PFRP; (2) we give $O(\sigma k^3)$-approximation algorithms for both One-way FFRP and Two-way FFRP; (3) we extend each of these algorithms to networks in d-dimensional $(d \geq 3)$ metric space, and generalize the approximation ratios of the extended algorithms; (4) we evaluate heuristic implementations of the proposed algorithms with realistic sensor network scenarios, and show that their performance is much better than the proven performance ratios suggest.

The remainder of this paper is organized as follows. Section II presents the network model and preliminaries. Section III first describes the algorithm for One-way PFRP, then proves its approximation ratio. Section IV gives the algorithm for Two-way PFRP, as well as the proof of its performance ratio. Section V provides approximation algorithms for both One-way and Two-way FFRP, and analyzes their approximation ratios. Section VI extends the approximation algorithms to higher dimensional networks and derives the corresponding performance ratios. Section VII discusses heuristics for the practical implementations of the presented approximation algorithms. These heuristic implementations are simulated in QualNet 3.8 [9] and the results are evaluated in Section VIII. Finally, Section IX concludes the paper with future research directions.

II. Network Model And Preliminaries

This section describes the model of heterogeneous wireless sensor networks, as well as some basic operations and preliminary knowledge that will be used in this paper.

A. Model of Heterogeneous Wireless Sensor Networks

We consider stationary heterogeneous wireless sensor networks with omni-directional transceivers. Each target node x possesses a (possibly different) transmission radius $T(x)$, and

[1]The authors of [4] also presented a MST-based 7-approximation algorithm provided that target nodes are also capable of communicating with each other.

$\|N\|$, $\|E\|$	The cardinality of sets N and E
$\overrightarrow{uv}$	Directed edge from u to v
$\widehat{uv}$	Undirected edge between u and v
$\|uv\|$	Euclidean distance between u and v
in-neighbor	x is an in-neighbor of u if $\overrightarrow{xu} \in E$
out-neighbor	x is an out-neighbor of u if $\overrightarrow{ux} \in E$
neighbor	x is a neighbor of u if $\widehat{ux} \in E$
$P_G(\overrightarrow{u,v})$	A directed path from u to v in G
$P^i_G(\overrightarrow{u,v})$	i-th directed path from u to v in G
$P_G(\widehat{u,v})$	An undirected path between u and v in G

TABLE I
TERMS, SYMBOLS AND THEIR SEMANTICS

all of the relay nodes use the same transmission radius *T(relay)*. *T(min)* and *T(max)* represent the minimum and the maximum transmission radius among all of the relay nodes and target nodes. Correspondingly, we define constants $\alpha = \lceil \frac{T(max)}{T(min)} \rceil$, $\beta = \lceil \frac{T(max)}{T(relay)} \rceil$, and $\gamma = \lceil \frac{T(relay)}{T(min)} \rceil$. Furthermore, each wireless node x has a transmission range, which is a circle in a 2D plane (or a sphere in a 3D space) centered at x with radius $T(x)$. Given these terms and notations, we model a H-WSN as a directed graph $G = (V \cup R, E)$, where V, R, and E are the set of target nodes, the set of additionally deployed relay nodes, and the set of directed edges, respectively. For any two nodes u and v in $V \cup R$, there is a directed edge $\overrightarrow{uv}$ from u to v in E if and only if v is in u's transmission range.

Relative to an arbitrary graph $G = (N, E)$, either directed or undirected, where N is the set of nodes and E is the set of edges, and two nodes u and v in N, Table I lists the terms, notations, and their semantics used in this paper. Note that the term *neighbor* is defined and used for undirected graphs, and the terms *in-neighbor* and *out-neighbor* are defined and used for directed graphs.

B. Steinerization of Edges

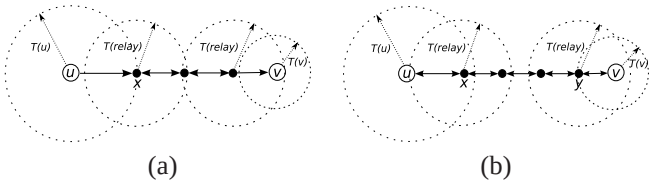
For two target nodes u and v, one common scenario in this paper is that we want to create a one-way path from u to v, or a two-way path between u and v, using as few relay nodes as possible, while ignoring all the other target nodes and any previously deployed relay nodes. To facilitate creating such paths, we define the following two operations.

(1) *One-way Steinerization*. We create an edge $\overrightarrow{uv}$ and *One-way Steinerize* $\overrightarrow{uv}$ as follows. Compute the weight of $\overrightarrow{uv}$ using Equation 1.

$$weight(\overrightarrow{uv}) = \begin{cases} 0 & \text{if } T(u) \geq |uv| \\ \lceil \frac{|uv| - T(u)}{T(relay)} \rceil & \text{if } T(u) < |uv| \end{cases} \quad (1)$$

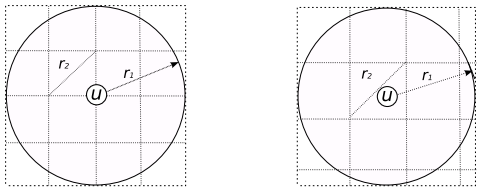
If $weight(\overrightarrow{uv}) \geq 1$, then place one relay node x on the straight line between u and v such that $|ux| = T(u)$, and then evenly place $weight(\overrightarrow{uv}) - 1$ relay nodes along the straight line between x and v. In this way, we create a one-way path from u to v. Figure 2(a) depicts this operation.

(2) *Two-way Steinerization*: We define $\delta = min\{T(u), T(v)\}$, $\lambda = min\{T(u), T(relay)\}$ and $\omega = min\{T(v), T(relay)\}$.



(a) (b)

Fig. 2. (a) One-way Steinerize the directed edge $\overrightarrow{uv}$ and create a one-way path from u to v. (b) Two-way Steinerize the undirected edge $\widehat{uv}$ and create a two-way path between u and v.



(a) (b)

Fig. 3. Segmentation of the neighborhood of node u with (r_1, r_2). (a) The encompassing square can be evenly segmented by cells. (a) The encompassing square cannot be evenly segmented by cells.

We create an edge $\widehat{uv}$ and *Two-way Steinerize* $\widehat{uv}$ as follows. Compute the weight of $\widehat{uv}$ using Equation 2.

$$weight(\widehat{uv}) = \begin{cases} 0 & \text{if } |uv| \leq \delta \\ \lceil \frac{|uv| - \lambda - \omega}{T(relay)} \rceil + 1 & \text{if } |uv| > \delta \end{cases} \quad (2)$$

If $weight(\widehat{uv}) = 1$, then place one relay node x on the straight line between u and v such that $|ux| = \lambda$. If $weight(\widehat{uv}) \geq 2$, then place two relay nodes x and y on the straight line between u and v such that $|ux| = \lambda$ and $|vy| = \omega$, and then evenly distribute $weight(\widehat{uv}) - 2$ relay nodes along the straight line between x and y. In this way, we create a two-way path between u and v. Figure 2(b) depicts this operation.

C. Segmentation of Neighborhood

Another operation frequently used in this paper is to divide a certain neighborhood area of a particular node into small regions such that the nodes within the same region are connected by directed or undirected one-hop communication links. Specifically, for a node u, we *segment* its neighborhood with a pair of positive values (r_1, r_2) in a 2D plane as follows. We first create a circle centered at u with radius r_1, along with a square just large enough to encompass the circle. Then, we evenly segment this square (from top to bottom, and from left to right) into small square *cells* with the length of the diagonal of each cell being equal to r_2 (or shorter than r_2 for some *cells* if the encompassing square cannot be evenly segmented). As depicted in Figures 3(a) and 3(b), this segmentation operation guarantees that the Euclidean distance between every pair of nodes in the same *cell* is no more than r_2.

D. Minimum k-Vertex Connected Spanning Graph

An important problem related to the relay node placement is to find a minimum k-vertex connected spanning graph (MKCSG). This concept depends on the following definition.

Definition 2.1: [*k-Vertex Connected Graph*]: Consider a graph $G = (N, E)$, where N and E are the node set and the edge set, respectively. Then G is a k-vertex connected graph (for short, G is k-vertex connected), if for any two nodes u and v in N, there exist k vertex-disjoint paths between u and v in G (or there exist k vertex-disjoint directed paths from u to v in G, if G is a directed graph).

Then, the MKCSG problem is to compute a k-vertex connected spanning graph of weighted complete graph with minimum total weight. For undirected graphs, when $k = 1$, this problem is exactly that of finding the minimum spanning tree; when $k \geq 2$, this problem is NP-hard, and the following results are known. Ravi and Williamson [10] claimed the first constant approximation algorithm with performance ratio $(2\sum_{i=1}^{k}\frac{1}{i})$. However, their proof was later found [11] to contain errors. In [12], Kortsarz and Nutov presented a k-approximation algorithm, and a $(\frac{k+1}{2})$-approximation algorithm for the case of $k \leq 7$. Most recently, Cheriyan *et al*. [13] developed an $O(lgk)$-approximation algorithm, provided the complete graph contains at least $6k^2$ nodes. For directed graphs, this problem is NP-hard when $k \geq 1$. Kortsarz and Nutov [12] presented a $(1+k)$-approximation algorithm, along with a $(2+\frac{k}{\|N\|})$-approximation algorithm (N is the node set) if the edge weights satisfy the triangle inequality.

Aside from the above algorithms with provable performance guarantees, Li and Hou [14] proposed an easily implemented heuristic algorithm, which works by iteratively adding edges in increasing order of edge weight until the resulting graph is k-vertex connected. Bredin *et. al* [5] improved the algorithm of [14] by imposing an optimization step, which tests each added edge in decreasing order of edge weight and removes any edge whose removal does not destroy the k-vertex connectivity. The simulation results in [5] show that the performance of the heuristic algorithms are typically close to the optimal.

III. One-way Partial Fault-Tolerance Relay Node Placement

This section presents an approximation algorithm for One-way PFRP in H-WSNs, and analyzes the quality of the result produced by the algorithm with respect to the optimal solution.

A. Algorithm for One-way PFRP

In brief, given a set of target nodes V, the algorithm first finds a directed MKCSG M of a complete graph over V, then One-way Steinerizes each edge in M. We leave the choice of an approximation algorithm for computing MKCSG as an open option, and assume that the approximation ratio of the selected algorithm is σ in our analysis. The complete algorithm is stated in Algorithm 1, and Theorem 3.1 states the performance ratio of Algorithm 1.

Theorem 3.1: Let V be a set of target nodes. Algorithm 1 is an $O(\sigma k^2)$-approximation algorithm in terms of the number of relay nodes required to form a one-way partial k-vertex connected network for V.

Algorithm 1 Algorithm for One-way PFRP

1: **INPUT: Integer** k **and a set of target nodes** V**.**
2: **OUTPUT: A set of relay nodes** R**.**
3: $R \leftarrow \phi$ (empty set); $W \leftarrow \{\overrightarrow{uv} \mid u,v\ (u \neq v) \in V\}$;
4: Define the weight of each edge $\overrightarrow{uv} \in W$ according to Equation 1;
5: $C \leftarrow (V,\ W)$;
6: Compute an approximate directed MKCSG M of C using a σ-approximation algorithm;
7: *One-way Steinerize* each edge $\overrightarrow{uv} \in M$ and place the relay nodes into R;
8: Output R;

B. Proof of the Performance Ratio

Although Algorithm 1 is relatively straightforward, the analysis of its performance ratio is complicated. We first give two definitions.

Definition 3.2: [*One-way Partial k-Vertex Connected Graph*]: Let V be a set of target nodes, and $G = (V \cup R, E)$ be a directed graph, where R is the set of additionally deployed relay nodes, and E is the set of directed edges. Then, G is a one-way partial k-vertex connected graph for V, if for every pair of target nodes u and v in V, there exist k vertex-disjoint one-way paths from u to v in G ($i.e.$, there exists at least one one-way path in G from u to v after the removal of fewer than k arbitrary nodes other than u and v).

Definition 3.3: [*Super Path*][4]: Let V be a set of target nodes, and $G = (V \cup R, E)$ be a one-way partial k-vertex connected graph for V. Then, a one-way path $P_G(\overrightarrow{u,v})$ in G is a super path, if u and v are target nodes and every interior node (if any) of $P_G(\overrightarrow{u,v})$ is a relay node.

From the description of Algorithm 1, we have an important observation that: *every relay node placed by Algorithm 1 is on exactly one super path*. We use this fact to help establish the quantitative relationship between the result produced by Algorithm 1 and the optimal solution. The entire analysis consists of three steps.

B.1 Step One

In this step, we prove that if there exists a one-way partial k-vertex connected graph $G = (V \cup R, E)$ for V where each relay node in R is on exactly one super path, then the number of relay nodes computed by Algorithm 1 on V is at most $\sigma\|R\|$. The analysis proceeds by first establishing Lemma 3.4 and Lemma 3.5, which show that duplicate super paths from the same starting node to the same ending node are unnecessary, if each relay node is on exactly one super path. Finally, Lemma 3.6 proves the result of Step One.

Lemma 3.4: Let $G = (N, E)$ ($\|N\| > k$) be a directed k-vertex connected graph. For two nodes u and v in N, if there are multiple directed edges from u to v in G, then if we keep one of these edges and remove the others, then the resulting graph $G' = (N, E')$ remains a directed k-vertex connected graph. Due to the space limitation, the proof of this lemma is omitted.

Lemma 3.5: Let V be a set of target nodes, and G = ($V \cup R$, E) be a one-way partial k-vertex connected graph for V where each relay node in R is on exactly one super path. For any pair of target nodes u and v in R, if in G there exist multiple super paths from u to v, then if we keep one of these super paths and remove the others, and denote the resulting graph as G' = ($V \cup R'$, E'), then G' remains a one-way partial k-vertex connected graph for V.

Proof: Since each relay node in R is on exactly one super path, by treating each super path as an edge, all of the target nodes and super paths in G form a directed k-vertex connected graph. From Lemma 3.4, it follows that after removing the redundant super paths, the target nodes and the remaining super paths in G' still form a directed k-vertex connected graph. This means that, in G', there exist k vertex-disjoint one-way paths from any target node to any other target node. Therefore, G' is a one-way partial k-vertex connected graph for V. ■

Lemma 3.6: Let V be a set of target nodes, and G = ($V \cup R$, E) be a one-way partial k-vertex connected graph for V where each relay node in R is on exactly one super path in G. Then, the number of relay nodes computed by Algorithm 1 on V is at most $\sigma\|R\|$.

Proof: For any pair of target nodes u and v in R, if in G there exist multiple super paths from u to v, we keep one of these super paths and remove the others, and denote the resulting graph as G' = ($V \cup R'$, E'). By Lemma 3.5, G' remains a one-way partial k-vertex connected graph for V, and we have $\|R'\| \leq \|R\|$.

Now consider the result produced by Algorithm 1. For two arbitrary target nodes u and v in V, Equation 1 defines the minimum number of relay nodes required to create a one-way path from u to v. Furthermore, Algorithm 1 uses a σ-approximation algorithm to compute an approximate directed MKCSG M. Therefore, the number of relay nodes computed by Algorithm 1 is at most $\sigma\|R'\| \leq \sigma\|R\|$. ■

B.2 Step Two

Let V be a set of target nodes, and G_o = ($V \cup R_o$, E_o) be an optimal one-way partial k-vertex connected graph for V. In this step, motivated by the analysis in [5], we perform a sequence of transformations on G_o, and create a new one-way partial k-vertex connected graph G_f = ($V \cup R_f$, E_f) for V where each relay node in R_f is on exactly one super path, and $\|R_f\| \leq ((32\beta\alpha^2 + \frac{1}{2})k^2 + 3k + 4)\|R_o\|$ (Note that α and β are defined in section II-A). Our analysis uses two transformation rules, which are defined in the following paragraphs.

- *Transformation Rule 1* : For relay node r, we segment its neighborhood with ($T(max)$, $T(min)$), and get at most $8\alpha^2$ cells. In each cell, node r randomly selects k target in-neighbors and k target out-neighbors (or select all of the target in-neighbors or out-neighbors if there are fewer than k of them) as its new target in-neighbors and out-neighbors, as well as all of the edges associated with the selected target nodes.

Denote graph G'_o = ($V \cup R_o$, E'_o) as the graph that results by performing Transformation Rule 1 on G_o. We have the following lemma.

Lemma 3.7: The graph G'_o = ($V \cup R_o$, E'_o) is a one-way partial k-vertex connected graph for V, and each relay node in R_o has at most $8\alpha^2 k$ target in-neighbors and $8\alpha^2 k$ target out-neighbors.

Proof: Let G = (N, E) be a graph where N and E are the sets of nodes and edges, respectively, and let X be a set of nodes (or edges). We define $G \backslash X$ as the graph that results by removing the nodes in $N \cap X$ plus their incident edges from G (or by removing the edges in $E \cap X$ from G, if X is an edge set).

For two target nodes u and v in V, and $k-1$ arbitrary nodes $A = \{n_1, n_2, .., n_{k-1}\}$ other than u and v in G'_o, we prove that there is a path from u to v in $G'_o \setminus A$.

Obviously, there is a one-way path $P_{G_o}(\overrightarrow{u,v})$ from u to v in $G_o \setminus A$. For an arbitrary hop on $P_{G_o}(\overrightarrow{u,v})$, say $\overrightarrow{xy}$, if both x and y are target nodes or relay nodes, then $\overrightarrow{xy}$ still exists in $G'_o \setminus A$. Now, consider the situation where x is a relay node and y is a target node: (1) if x chooses y as its new out-neighbor in G'_o, then $\overrightarrow{xy}$ exists in $G'_o \setminus A$; (2) if y is not chosen, then in the cell of x where y resides, x must select k other target out-neighbors, and at least one of these target out-neighbors z is not in A. Since $|zy| \leq T(min)$, $\overrightarrow{zy}$ exists in $G'_o \setminus A$. Therefore, we replace $\overrightarrow{xy}$ with $\overrightarrow{xz}$ plus $\overrightarrow{zy}$ in $G'_o \setminus A$. We can perform a similar replacement for the situation where x is a target node and y is a relay node.

Because each hop on $P_{G_o}(\overrightarrow{u,v})$ is valid in $G'_o \setminus A$, there is a one-way path from u to v in $G'_o \setminus A$. Therefore, G'_o is a one-way partial k-vertex connected graph for V. Moreover, since there are at most k target in-neighbors and k target out-neighbors in each cell, each relay node in G'_o has at most $8\alpha^2 k$ target in-neighbors and $8\alpha^2 k$ target out-neighbors. ■

Before stating the second transformation rule, we first provide three additional definitions.

Definition 3.8: [*Relay Component*][5]: Let V be a set of target nodes, and G = ($V \cup R$, E) be a one-way partial k-vertex connected graph for V. For a relay node $r \in R$, the relay component of r can be derived as follows. We start at r, travel along each edge incident to r in G (when we travel, we omit the direction of each edge, and traverse in either direction). If we meet a relay node, we repeat the process; if we meet a target node, we stop. Finally, all of the nodes (target or relay) and edges visited in this recursive process form a relay component of r. Intuitively, all of the boundary nodes in the relay component are target nodes.

Definition 3.9: [*Undirected Spanning Tree of Relay Component*]: Let C_i = ($V_i \cup R_i$, E_i) be a relay component, where V_i, R_i, and E_i are the sets of target nodes, relay nodes and directed edges in C_i, respectively. We first create a new undirected graph C'_i from C_i by omitting the direction of each edge in E_i. Then, an undirected spanning tree of C_i, denoted as $Tree(C_i)$, is a spanning tree of C'_i rooted at an arbitrary relay node and having all of the target nodes in V_i as leaves.

Definition 3.10: [*Harary Graph*]: Let V be a set of nodes. We can construct a Harary graph of V as follows. We place all of the nodes in V into a circular doubly-linked list L. For each node x in L, we add undirected edges between node x and k

nearest nodes of x in L. A Harary graph is a k-vertex connected graph. If we replace each undirected edge in a Harary graph with a pair of opposite directed edges, the resulting graph is a directed k-vertex connected graph.

Assume that G'_o = ($V \cup R_o$, E'_o) has m relay components, denoted as C_i=($V_i \cup R_i$, E_i) ($1 \leq i \leq m$). Now we define the second transformation rule, which is performed on each relay component.

• *Transformation Rule 2* : For each relay component C_i = ($V_i \cup R_i$, E_i): (1) if $\|V_i\| > k$, we first make a clockwise Eulerian tour of $Tree(C_i)$, and place the target nodes in a circular doubly-linked list L in the order in which they are visited in this Eulerian tour. Then, we remove all of the relay nodes in C_i, and create a Harary Graph in C_i by connecting each target node with k nearest target nodes in L; (2) if $\|V_i\| \leq k$, we first remove all of the relay nodes in C_i, and create a complete undirected graph in C_i by connecting each target node with all of the other target nodes. Then, we add $k - \|V_i\| + 1$ duplicate edges for each edge in this complete graph. Finally, after creating a Harary graph or a complete graph with duplicate edges, we replace each edge with two opposite directed edges, and One-way Steinerize all of the directed edges.

Let C'_i = ($V_i \cup R'_i$, E'_i) ($1 \leq i \leq m$) be the graph that results by performing Transformation Rule 2 on relay component C_i, and let graph G''_o = ($V \cup R''_o$, E''_o) be the graph that results by performing Transformation Rule 2 on each relay component in G'_o. We have the following lemma.

Lemma 3.11: The graph G''_o is a one-way partial k-vertex connected graph for V, and each relay node in G''_o is on exactly one super path.

Proof: For two target nodes u and v in V, and $k-1$ arbitrary nodes $A = \{n_1, n_2, .., n_{k-1}\}$ other than u and v in G''_o, we prove that there is a path in $G''_o \setminus A$ from u to v. By Lemma 3.7, there is a one-way path $P_{G'_o}(\overrightarrow{u,v})$ from u to v in $G'_o \setminus A$. We partition $P_{G'_o}(\overrightarrow{u,v})$ into multiple sub-paths, where each sub-path is a super path. For an arbitrary sub-path $P_{G'_o}(\overrightarrow{x,y})$: (1) if $P_{G'_o}(\overrightarrow{x,y})$ does not contain any relay nodes, then $P_{G'_o}(\overrightarrow{x,y})$ exists in $G''_o \setminus A$; (2) if $P_{G'_o}(\overrightarrow{x,y})$ contains some relay nodes, x and y must be in the same relay component C_j in G'_o, and hence x and y are in C'_j. Transformation Rule 2 guarantees that there is at least one one-way path from x to y in $C'_j \setminus A$, which can be used to replace the sub-path $P_{G'_o}(\overrightarrow{x,y})$ in G''_o. Since each sub-path of $P_{G'_o}(\overrightarrow{u,v})$ is still valid in G''_o, there is a one-way path from u to v in $G''_o \setminus A$, which means G''_o is a one-way partial k-vertex connected graph for V. Moreover, each relay node in G''_o is on exactly one super path. ■

Finally, we generate the graph G_f=($V \cup R_f$, E_f) as follows. For two arbitrary target nodes u and v in G''_o = ($V \cup R''_o$, E''_o), if there are multiple super paths from u to v in G''_o, we keep one of those super paths and remove the others. By Lemma 3.5, G_f remains a one-way partial k-vertex connected graph for V. Furthermore, each relay node in G_f is on exactly one super path. Lemma 3.12 presents the result of Step Two.

Lemma 3.12: $\|R_f\| \leq ((32\beta\alpha^2 + \frac{1}{2})k^2 + 3k + 4)\|R_o\|$.

Proof: For any relay component C_i = ($V_i \cup R_i$, E_i) in G'_o, by Lemma 3.7, we have $\|V_i\| \leq 16\alpha^2 k\|R_i\|$. After applying Transformation Rule 2 on C_i, we have C'_i=($V_i \cup R'_i$, E'_i).

When $\|V_i\| > k$, G_f contains C'_i. We use $Tree(C_i)$ to count the number of relay nodes in R'_i. For an arbitrary super path from u to v added by Transformation Rule 2 in C'_i, we spread the weight of $\widehat{uv}$ on the path $P_{Tree(C_i)}(\widehat{u,v})$ between u and v in $Tree(C_i)$ by charging β on the first and the last hop, and charging one on each interior hop. For each edge $\widehat{xy}$ in $Tree(C_i)$: (1) if $\widehat{xy}$ is incident to a target node, then $\widehat{xy}$ is totally charged $2\beta k$; (2) otherwise, we denote the target descendants of x in $Tree(C_i)$ from left to right as $\{t_1, t_2, \cdots, t_n\}$. For the nodes from t_1 to $t_{\lceil \frac{k}{2} \rceil}$, $\widehat{xy}$ is charged $2\lceil \frac{k}{2} \rceil$, $2(\lceil \frac{k}{2} \rceil - 1)$, $\cdots$, 1, respectively. And for the nodes from t_n to $t_{n-\lceil \frac{k}{2} \rceil + 1}$, $\widehat{xy}$ is also charged $2\lceil \frac{k}{2} \rceil$, $2(\lceil \frac{k}{2} \rceil - 1)$, $\cdots$, 1, respectively. Therefore, $\widehat{xy}$ is charged at most $(\frac{1}{2}k^2 + 3k + 4)$. As a result, all of the edges in $Tree(C_i)$ are charged a total of at most $2\beta k\|V_i\| + (\frac{1}{2}k^2 + 3k + 4)\|R_i\|$, which is no more than $((32\beta\alpha^2 + \frac{1}{2})k^2 + 3k + 4)\|R_i\|$.

When $\|V_i\| \leq k$, G_f keeps a one-way path from any target node to any other target node. In this case, all of the edges in $Tree(C_i)$ are charged a total of no more than $((32\beta\alpha^2 + \frac{1}{2})k^2 + 3k + 4)\|R_i\|$.

Summing up all of the relay components, we have $\|R_f\| \leq ((32\beta\alpha^2 + \frac{1}{2})k^2 + 3k + 4)\|R_o\|$. ■

B.3 Step Three

Proof of Theorem 3.1: By Lemma 3.6 and Lemma 3.12, the number of relay nodes added in Algorithm 1 is at most $\sigma((32\beta\alpha^2 + \frac{1}{2})k^2 + 3k + 4)$ times the optimal solution. Therefore, Algorithm 1 is an $O(\sigma k^2)$-approximation algorithm. ■

IV. Two-way Partial Fault-Tolerance Relay Node Placement

This section provides an approximation algorithm for Two-way PFRP, and derives the performance ratio of the algorithm by following the framework used for analyzing Algorithm 1. The complete algorithm is presented in Algorithm 2, and Theorem 4.1 describes its performance ratio.

Algorithm 2 Algorithm for Two-way PFRP

1: **INPUT: Integer k and a set of target nodes V.**
2: **OUTPUT: A set of relay nodes R.**
3: $R \leftarrow \phi$; $W \leftarrow \{\widehat{uv} \mid u, v\ (u \neq v) \in V\}$;
4: Define the weight of each edge $\widehat{uv} \in W$ according to Equation 2;
5: $C \leftarrow (V,\ W)$;
6: Compute an approximate undirected MKCSG M of C using a σ-approximation algorithm;
7: *Two-way Steinerize* each edge $\widehat{uv} \in M$ and place the relay nodes in R;
8: Output R;

Theorem 4.1: Let V be a set of target nodes. Algorithm 2 is an $O(\sigma k^2)$-approximation algorithm in terms of the number of relay nodes required to form a two-way partial k-vertex connected network for V.

Note that Algorithm 2 assumes the use of an undirected graph. We can analogously define *two-way partial k-vertex connected graph* and *super path* for undirected graph, and Lemma 4.2 follows directly.

Lemma 4.2: Let V be a set of target nodes, and let G = ($V \cup R$, E) be a two-way partial k-vertex connected graph for V where each relay node in R is on exactly one super path. Then, the number of relay nodes computed by Algorithm 2 on V is at most $\sigma\|R\|$.

Further, we modify the transformation rules in previous sections as follows: (1) in Transformation Rule 1, for each relay node r, we segment its neighborhood with $(T(relay), T(min))$, and let r select k target neighbors in each cell; (2) in Transformation Rule 2, for each relay component C_i, after creating a Harary graph or a complete graph with duplicate edges, we directly two-way Steinerize each edge. Moreover, when we count the number of relay nodes for Two-way Steinerizing each added edge $\widehat{uv}$ with $Tree(C_i)$, we charge *one* on every hop in the path between u and v in $Tree(C_i)$.

Then, it immediately follows:

Lemma 4.3: Let V be a set of target nodes, and G_o = ($V \cup R_o$, E_o) be an optimal two-way partial k-vertex connected graph for V. There exists a two-way partial k-vertex connected graph G_f = ($V \cup R_f$, E_f) for V where each relay node in R_f is on exactly one super-path, and $\|R_f\| \leq ((8\gamma^2+\frac{1}{4})k^2+\frac{3}{2}k+2)\|R_o\|$.

Proof of Theorem 4.1: By Lemma 4.2 and Lemma 4.3, the number of relay nodes added in Algorithm 1 is at most $\sigma((8\gamma^2+\frac{1}{4})k^2+\frac{3}{2}k+2)$ times the optimal solution. Therefore, Algorithm 2 is an $O(\sigma k^2)$-approximation algorithm. ■

V. One-way And Two-way Full Fault-Tolerance Relay Node Placement

In this section, based on the work for One-way PFRP and Two-way PFRP, we propose approximation algorithms for both One-way FFRP and Two-way FFRP, and present the analysis of their performance ratios. The algorithms are described in Algorithm 3.

We first analyze the performance of Algorithm 3 for One-way FFRP. Algorithm 3 executes the One-way PFRP algorithm on V and produces a resulting network G = ($V \cup R$, E). Then, for each super path $P_G(\overrightarrow{u,v})$ in G, Algorithm 3 replicates each relay node on $P_G(\overrightarrow{u,v})$ with $k-1$ additional relay nodes. Furthermore, in Step 6, Algorithm 3 connects the relay nodes on $P_G(\overrightarrow{u,v})$ with u and v as well as their in-neighbors and out-neighbors. We term the operation in Step 6 *full-connection*. Although *full-connection* is costly, it guarantees that there are k vertex-disjoint one-way paths from any node to any other node in the resulting network. Finally, in Step 7, Algorithm 3 tests each cluster of relay nodes deployed in Step 6, and removes the cluster if the graph of the resulting network remains a directed k-vertex connected graph. Simulation results show that Step 7 on the average removes 83.6% of the relay nodes deployed in Step 6. Theorem 5.1 presents the approximation ratio of Algorithm 3 for One-way FFRP.

Algorithm 3 Algorithm for One-way (Two-way) FFRP

1: **INPUT: Integer** k **and a set of target nodes** V
2: **OUTPUT: A set of relay nodes** F
3: $F \leftarrow \phi$;
4: Execute the One-way (Two-way) PFRP algorithm on V and obtain a set of relay nodes R, as well as the resulting network $G = (V \cup R, E)$;
5: For each super path $P_G(\overrightarrow{u,v})$ ($P_G(\widehat{u,v})$ for Two-way FFRP) in G, at the position of every relay node in $P_G(\overrightarrow{u,v})$ ($P_G(\widehat{u,v})$), place $k-1$ additional relay nodes, and add the new relay nodes into F;
6: For each target node u in V, if u is the starting or the ending node of a super path containing relay nodes in G, then segment u's neighborhood with $(T(u), T(relay))$, and place a cluster of $k-1$ relay nodes at the position of u, and a cluster of k relay nodes at the $center$ of each cell. Add every cluster of relay nodes into F;
7: **For** each cluster of relay nodes deployed in Step 6
- Remove all of the relay nodes in the cluster from F;
- If the resulting network is not k-vertex connected, restore all of the relay nodes in the cluster;

End For
8: Output F=$F \cup R$;

Theorem 5.1: Let V be a set of target nodes. Algorithm 3 is an $O(\sigma k^3)$-approximation algorithm in terms of the number of relay nodes required to form a one-way full k-vertex connected network for V.

Proof: Denote R and F as the relay nodes sets that result by respectively running the One-way PFRP algorithm and the One-way FFRP algorithm on V, and denote G = ($V \cup R$, E) as the network that results by deploying R in V.

Now consider the optimal set F_o of relay nodes for One-way FFRP. Since F_o is also a solution to One-way PFRP, we have $\|R\| < \sigma((32\beta\alpha^2+\frac{1}{2})k^2+3k+4)\|F_o\|$. For an arbitrary super path $P_G(\overrightarrow{u,v})$ in G, Algorithm 3 adds at most $(8\beta^2+1)k + weight(\overrightarrow{uv})k + (8\beta^2+1)k$ relay nodes. Therefore, for all of the super paths in G, Algorithm 3 totally adds at most $(16\beta^2+2)k\|R\| + \|R\|k$ relay nodes. As a result, we have $\|F\| \leq \sigma k(16\beta^2+3)((32\beta\alpha^2+\frac{1}{2})k^2+3k+4)\|F_o\|$. Therefore, Algorithm 3 is an $O(\sigma k^3)$-approximation algorithm. ■

The analysis of Algorithm 3 for Two-way FFRP can be conducted in a similar manner, and Theorem 5.2 states the result.

Theorem 5.2: Let V be a set of target nodes. Algorithm 3 is an $O(\sigma k^3)$-approximation algorithm in terms of the number of relay nodes required to form a two-way full k-vertex connected network for V.

VI. Extensions to Higher Dimensions

This section discusses extending the approximation algorithms presented in previous sections to higher dimensional H-WSNs, and derives the approximation ratio of each of these extended algorithms.

Algorithms	Performance Ratio in d-dimensional H-WSNs
One-way PFRP	$O(\sigma(2\sqrt{d}\alpha)^d k^2)$
Two-way PFRP	$O(\sigma(2\sqrt{d}\gamma)^d k^2)$
One-way FFRP	$O(\sigma(4d\alpha\beta)^d k^3)$
Two-way FFRP	$O(\sigma(4d\gamma\beta)^d k^3)$

TABLE II
PERFORMANCE RATIOS IN HIGHER DIMENSIONS

We first modify the method of segmenting the neighborhood of a node x with (r_1, r_2) in d-dimensional ($d \geq 3$) metric space as follows. We create a d-dimensional sphere centered at x with radius r_1, along with a d-dimensional cube just large enough to encompass this sphere. Then, we evenly segment this d-dimensional cube into small d-dimensional cube *cells* where the length of the diagonal of each *cell* equals r_2.

With this extended segmentation method, we can apply the algorithms and the corresponding analysis presented in previous sections to tackle the relay node placement in the situation where heterogeneous target nodes are deployed in high-dimensional metric space. The performance ratios of these algorithms in d-dimensional H-WSNs are listed in Table II.

VII. HEURISTIC IMPLEMENTATIONS

All of the approximation algorithms discussed in the previous sections are based on an approximation algorithm for finding a MKCSG. However, most existing approximation algorithms for MKCSG problem uses the Frank and Tardos algorithm [15] or its variations as the subroutine, and are quite complicated and difficult to implement in computation- and communication-constrained sensor networks [5]. Thus, motivated by the heuristic algorithms described in [14] and [5], we present a greedy heuristic algorithm in this section as a practical alternative for the MKCSG problem.

The basic idea of this greedy algorithm is to repeatedly add the edges that can best help to improve the graph connectivity until the graph becomes k-vertex connected. To quantitatively measure the improvement of each edge on the graph connectivity, we define the concept of the *contribution* of the edges.

Definition 7.1: [*Contribution*]: In a graph that is *not* k-vertex connected, there must exist some node pairs which can be partitioned by the removal of fewer than k nodes, which means the connectivity between such node pairs is lower than k. We term these node pairs *unsaturated node pairs*. Consequently, the *contribution* of an edge $\overrightarrow{uv}$ (or $\widehat{uv}$) is defined as the number of *unsaturated node pairs* whose connectivity can be improved by the deployment of edge $\overrightarrow{uv}$ (or $\widehat{uv}$) in this graph.

The greedy algorithm is stated in Algorithm 4. One key step in Algorithm 4 is to the check the connectivity between node pairs and the connectivity of the entire graph. This can be achieved by using the well-known maximum network flow based checking algorithm presented in [10].

Algorithm 4 Greedy Algorithm for MKCSG

1: **INPUT: Integer k, an undirected (or directed) weighted complete graph G = (V, E).**
2: **OUTPUT: An undirected (or directed) k-vertex connected spanning graph M of G.**
3: $S \leftarrow \phi$;
4: $M \leftarrow (V, S)$;
5: Place all of the edges in G with weight zero into S;
6: **While** M is not k-vertex connected
- Add the edge in $G \backslash S$ with highest contribution to S (If multiple edges have the same highest contribution, add an edge with the lowest weight);

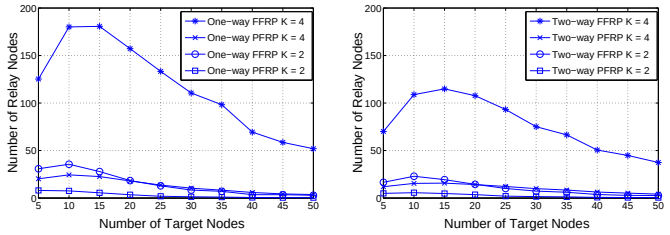
End While
7: Test each edge $\widehat{uv}$ (or $\overrightarrow{uv}$) $\in M$ in decreasing order of weight, and delete $\widehat{uv}$ (or $\overrightarrow{uv}$) from M if $M \backslash \widehat{uv}$ (or $\overrightarrow{uv}$) is k-vertex connected;
8: Output M

VIII. EXPERIMENTAL RESULTS

This section evaluates the performance of our algorithms using heuristic implementations described in Section VII. We use Qualnet 3.8[9] as the simulation platform. In each of these simulations, we randomly place target nodes in a $1000m \times 1000m$ 2D terrain. To model a H-WSN, we set $T(min) = 200m$ and $T(max) = 500m$, and let every target node use a random transmission radius between $T(min)$ and $T(max)$ in each simulation. Each of the results presented in this section is the average of 50 runs.

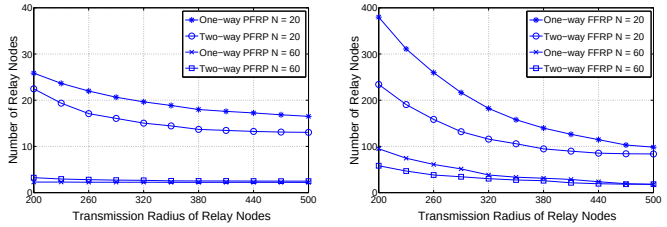
In the first simulation, we use $T(relay) = 350m$, and gradually increase the number of target nodes in the network from 5 to 50. Figures 4(a) and 4(b) depict the performance of each of the described algorithms. We have follow observations: (1) The number of relay nodes computed by all of the algorithms first increases, then decreases when the network size goes beyond a threshold. (2) On the average, the One-way FFRP algorithm requires 5.9 times and 10.2 times more relay nodes than the One-way PFRP algorithm for the cases of $k = 2$ and $k = 4$, respectively. (3) Likewise, on the average, the Two-way FFRP algorithm requires 4.6 times and 7.7 times more relay nodes than the Two-way PFRP algorithm for the cases of $k = 2$ and $k = 4$, respectively. We further continue the first simulation by increasing the number of target nodes in the network from 50 to 100, and the number of relay nodes computed by all of the algorithms converges to small values fewer than 9.

In the second simulation, we gradually increase $T(relay)$ from $T(min)$ to $T(max)$, and evaluate the algorithms for the case of $k = 4$ in two networks containing 20 and 60 target nodes, respectively. As depicted in Figures 5(a) and 5(b), the increase of $T(relay)$ can effectively improve the performance of all of the algorithms. Specifically, we have the following observations: (1) In a sparse network with 20 target nodes, when T(relay) increases from $T(min)$ to $T(max)$, the edge weight drops steadily, and the number of relay nodes computed by the One-way and the Two-way PFRP algorithms drops 36% and 42%, respectively. (2) In a dense network with 60 target nodes, most edges selected by the One-way and the Two-way PFRP algorithms carry weight 1, as a result, when $T(relay)$



(a) (b)

Fig. 4. (a) Results of the One-way PFRP and the One-way FFRP Algorithms (b) Results of the Two-way PFRP and the Two-way FFRP Algorithms



(a) (b)

Fig. 5. The case of $k = 4$. N stands for the network size. (a) Results of the One-way and the Two-way PFRP Algorithms. (b) Results of the One-way and the Two-way FFRP Algorithms.

increases, the performance of the One-way and the Two-way PFRP algorithms remain stable. (3) For full fault-tolerance, the increase of $T(relay)$ exponentially reduces the cost of the *full-connection* operation in Step 6 of Algorithm 3. Therefore, in a sparse network, the number of relay nodes computed by the One-way FFRP and the Two-way FFRP algorithms drops 74% and 64%, respectively; and in a dense network, the number of relay nodes computed by the One-way and the Two-way FFRP algorithms drops 82% and 67%, respectively.

Overall, the simulation results show that the expected behaviors of the described algorithms are much better than the performance ratios suggest, which indicates that these algorithms work well in real sensor networking applications where the target nodes are usually densely deployed.

IX. Conclusion And Future Work

This paper systematically addresses the problem of deploying a minimum number of relay nodes to achieve diverse levels of fault-tolerance in the context of heterogeneous wireless sensor networks, where target nodes have different transmission radii. The different transmission radii of the target nodes introduce *asymmetric communication links* between neighboring nodes, resulting in one-way and two-way paths. The problem is further complicated by the need to facilitate the desired fault-tolerance levels between every pair of (target and/or relay) nodes, or every pair of target nodes. Specifically, we develop $O(\sigma k^2)$-approximation algorithms for one-way and two-way partial fault-tolerance relay node placement, and $O(\sigma k^3)$-approximation algorithms for one-way and two-way full fault-tolerance relay node placement. Furthermore, we extend these algorithms to d-dimensional networks ($d \geq 3$), and generalize the approximation ratios of the extended algorithms. To support real applications, we also provide heuristic implementations of these algorithms, and evaluate their performance via simulations. The results show that the performance of the proposed algorithms is much better than the performance ratios derived, suggesting that these algorithms work well in real sensor networking scenarios. In ongoing work, we are pursuing tighter performance ratios of the approximation algorithms, as well as better heuristic implementations.

ACKNOWLEDGMENT: We would like to thank the authors of [5] [13] for useful discussions on the heuristic implementations of our algorithms. We also would thank the anonymous reviewers for their time and valuable feedback. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government.

References

[1] G. Lin and G. Xue, "Steiner tree problem with minimum number of steiner points and bounded edge-length," *Information Processing Letters*, vol. 69, pp. 53–57, 1999.

[2] D. Chen, D. Du, X. Hu, G. Lin, L. Wang, and G. Xue, "Approximations for steiner trees with minimum number of steiner points," *Journal of Global Optimization*, vol. 18, pp. 17–33, 2000.

[3] X. Cheng, D. Du, L. Wang, and B. Xu, "Relay sensor placement in wireless sensor networks," *Accepted by ACM/Springer WINET*. [Online]. Available: http://www.seas.gwu.edu/∼cheng/Publication/relay.pdf.

[4] E. L. Lloyd and G. Xue, "Relay node placement in wireless sensor networks," *IEEE Transactions on Computers*, vol. 56, pp. 134–138, 2007.

[5] J. L. Bredin, E. D. Demaine, M. Hajiaghayi, and D. Rus, "Deploying sensor networks with guaranteed capacity and fault tolerance," in *ACM MobiHoc*, 2005.

[6] A. Kashyap, S. Khuller, and M. Shayman, "Relay Placement for Higher Order Connectivity in Wireless Sensor Networks," in *IEEE INFOCOM*, 2006.

[7] J. Tang, B. Hao, and A. Sen, "Relay node placement in large scale wireless sensor networks," *Computer Communications*, vol. 29, pp. 490–501, 2006.

[8] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, 1979.

[9] "Qualnet simulator," Scalable Network Technologies, Inc., http://www.-scalable-networks.com.

[10] R. Ravi and D. P. Williamson, "An approximation algorithm for minimum-cost vertex-connectivity problems," in *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 1995.

[11] ——, "An approximation algorithm for minimum-cost vertex-connectivity problems," *Algorithmica*, vol. 34, pp. 98–107, 2002.

[12] G. Kortsarz1 and Z. Nutov, "Approximating node connectivity problems via set covers," *Algorithmica*, vol. 37, pp. 75–92, 2003.

[13] J. Cheriyan, S. Vempala, and A. Vetta, "Approximation algorithms for minimum-cost k-vertex connected subgraphs," in *34th Annual ACM Symposium on the Theory of Computing*, 2002, pp. 306–312.

[14] N. Li and J. C. Hou, "Flss: a fault-tolerant topology control algorithm for wireless networks," in *ACM MobiCom*, 2004.

[15] A. Frank and E. Tardos, "An application of submodular flows," *Linear Algebra Application*, vol. 243(1-2), pp. 329–348, 1989.