Mobile Relay Scheduling in Partitioned Wireless Sensor Networks

Zhong Shen, *Member, IEEE*, and Hai Jiang, *Senior Member, IEEE*

***Abstract*—In a wireless sensor network (WSN), sensors collect data that need to be delivered to a base station. This requires that the WSN should be a connected network. However, due to possible failures of sensors, a WSN might be partitioned into multiple isolated blocks. A solution to reconnect the WSN is to deploy mobile relays in the WSN, and schedule the mobile relays to positions that can reconnect the isolated blocks. In this paper, we target at a mobile relay scheduling algorithm with the minimal total movement cost such that the partitioned WSN becomes connected. We give a definition for boundary sensors of blocks, and show that, to connect two blocks, fewer mobile relays are used if we connect the two blocks through a pair of their boundary sensors. To connect a given pair of boundary sensors from two blocks, we derive the optimal new positions of mobile relays, which minimize the total energy consumption of mobile relays when relaying information between the two blocks. To connect a partitioned WSN with multiple isolated blocks, a mobile relay scheduling problem is formulated and shown to be NP-complete. Then a greedy mobile relay scheduling algorithm is proposed to select boundary sensor pairs and reconnect the WSN by scheduling mobile relays to optimal new positions of connecting the selected boundary sensor pairs. Since the selected boundary sensors are gateways of blocks, and the moved mobile relays serve as routers to connect the blocks, any failure of them may disconnect the WSN again. Accordingly, another greedy mobile relay scheduling algorithm is proposed to reconnect a partitioned WSN such that the reconnected WSN can tolerate failure of one gateway or one router. Extensive simulations demonstrate that the proposed algorithms have better performance over other existing methods in terms of higher success probability and less movement cost.**

***Index Terms*—Bi-connectivity, connectivity, mobile relays, network partitions, wireless sensor networks.**

I. Introduction

In a wireless sensor network (WSN) [1], sensors collect and forward data to a base station (BS). For this, one basic requirement is that the WSN should be connected such that data collected by each sensor can be successfully delivered (through one or multiple hops) to the BS. However, in some cases, WSNs may be disconnected due to node damage [2]–[6]. For example, for WSNs in battlefield surveillance, sensors in some areas may be destroyed by enemies or explosives; for WSNs in environmental monitoring, sensors may be damaged by natural disasters (fire, floods, earthquakes, etc.). Such large-scale damages may disconnect and partition a WSN into several isolated *blocks* [2], [3]. Each block is a connected sub-network, and sensors in different blocks cannot communicate with each other.

Although adding more sensors in the target area may make the WSN more likely to remain connected after some node failures, this method is costly, and may be inefficient and with high risk in hostile or hazardous environments. An alternative solution is to place *relays* between blocks such that the blocks are reconnected [5], [7], [8]. Generally, relays are more costly devices, and are more powerful (e.g., have more energy supply and a larger transmission range than regular sensors). Since the relays are more expensive, it is desired to use as few relays as possible to reconnect a partitioned WSN. Recently relay-assisted WSN reconnection has received much research attention [5], [7]–[12]. These existing research works assume that relays can be placed at any positions. However, this may not be true when the application environment is remote, hostile, or hazardous.

In remote, hostile, or hazardous environments, mobile nodes can be used to move to new locations to serve as relays for sensors [2], [13], [14]. The micro-electro-mechanical system and robotic techniques enable nodes to move, which significantly benefits the applications of WSNs. A variety of robotic platforms have been developed including car-like robots, robotic fish, and *micro aerial vehicles* (MAVs) [14]. Compared with car-like robots, MAVs such as *DelFly* [15], *DragonFly* [16], and *SensorFly* [17] can quickly fly to target locations. When these robotic platforms are equipped with sensing and communication units, they are often called *mobile sensors*. The applications of mobile sensors have attracted a lot of attention. In many applications, mobile sensors and regular sensors together form a *hybrid WSN*, and mobile sensors can be used to collect data from other sensors [18], to improve network coverage [19]–[22], to repair networks [6], [23], [24], and to enhance fault tolerance [25], [26].

In this paper, we investigate reconnecting a partitioned WSN by mobile sensors. Like [6], [19]–[22], we also consider a hybrid WSN, in which mobile sensors are deployed together with regular sensors. If the WSN is partitioned due to damages to regular sensors (some mobile sensors may also be damaged), remaining mobile sensors can be scheduled to move to new locations to act as routers to maintain the network connectivity.

The following three practical factors for mobile sensors in WSNs are taken into account in our research.

1). The moving capability of each mobile sensor is limited.

Manuscript received August 13, 2014; revised February 12, 2015 and June 4, 2015; accepted July 3, 2015. This work was supported in part by the National Natural Science Foundation of China under Grant 61172069, by the “111” Project under Grant B08038 with Xidian University, China, and by the Natural Sciences and Engineering Research Council of Canada. The associate editor coordinating the review of this paper and approving it for publication was Dr. A. Grieco. Z. Shen is with the State Key Laboratory of Integrated Services Networks, Xidian University, Xi’an 710071, China (e-mail: zhshen@mail.xidian.edu.cn). H. Jiang is with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, Alberta, Canada T6G 2V4 (e-mail: hai1@ualberta.ca).

For example, the maximal moving distance of a mobile sensor in the XYZ platform in [27] is 165 meters. In a self-healing minefield program, a mobile sensor is equipped with a fuel propeller and can make up to 100 leaps [21]. MAVs such as DelFly [15] and DragonFly [16] have about 15 minutes and 25 minutes flight time, respectively.

2). For a mobile sensor, power consumption in movement is much higher than that in data communications. For example, the energy consumption for a mobile sensor to move one meter is about 27.96 Joule, whereas the energy consumption to transmit one bit information is only about 1×10^{-7} Joule [28]. So, our main goal is to minimize the total movement cost (defined as movement distance) in rebuilding network connectivity.

3). It is desired that the rebuilt WSN can still be connected if subsequently one mobile sensor that connects the blocks fails.

Since mobile sensors will serve as relays, to be clear, we call them *mobile relays* in the sequel. With the above three factors taken into account, we give a definition for *boundary sensors* of blocks, and show that, to connect two blocks with as few mobile relays as possible, we should connect the two blocks through a pair of their boundary sensors. To connect a given pair of boundary sensors from two blocks, we derive the optimal new positions of mobile relays, which minimize the total energy consumption of mobile relays when relaying information between the two blocks. To connect a partitioned WSN including multiple isolated blocks, we formulate a mobile relay scheduling problem, and show that it is NP-complete. Thus, we focus on heuristic algorithms. A greedy mobile relay scheduling algorithm is proposed to reconnect a partitioned WSN by selecting boundary sensor pairs and scheduling mobile relays to optimal new positions of connecting the selected boundary sensor pairs. Since the selected boundary sensors are gateways of blocks, and the moved mobile relays serve as routers to connect the blocks, any failure of them may disconnect the WSN again. Accordingly, another greedy mobile relay scheduling algorithm is proposed to reconnect a partitioned WSN such that the reconnected WSN can tolerate failure of one gateway or one router.

The remainder of this paper is organized as follows. Related works are reviewed and discussed in Section II. The network model is presented in Section III. Section IV presents some results regarding connecting two isolated blocks and formulates a mobile relay scheduling problem that connects a partitioned WSN including multiple isolated blocks. The proposed mobile relay scheduling algorithms are detailed in Section V. Performance evaluation is given in Section VI. Conclusion is given in Section VII. Table I summarizes some important symbols used in this paper.

TABLE I
SUMMARY OF IMPORTANT SYMBOLS USED.

Symbols	Description
$\mathbb{B}_i$	the *i*th block (or partition)
$\mathbf{G}_b$	a bipartite graph
K	the number of blocks (or partitions)
$L(s_i, s_j)$	the minimum number of mobile relays needed to connect s_i and s_j
m_j	the *j*th mobile relay
$\mathbb{M}$	the set of mobile relays
$\mathcal{M}^*$	a maximum cardinality matching of $\mathbf{G}_b$ with minimum cost
n	the number of sensors
N	the number of mobile relays
$\mathbb{P}_{s_i s_j}$	the set of optimal positions of mobile relays to connect s_i and s_j
$\mathbb{P}$	the set of new mobile relay positions determined by a mobile relay scheduling algorithm
r	the transmission range of a sensor
R	the transmission range of a mobile relay
s_i	the *i*th sensor
$\|s_i s_j\|$	the distance between s_i and s_j
$\mathbb{S}$	the set of sensors
α	path loss exponent
λ	the maximum number of mobile relays needed to connect any boundary sensor pair from two different blocks
η	mobile relay percentage

II. Related Works

In this section, we briefly review related research efforts on reconnecting partitioned WSNs.

With assistance of relay nodes, the problem of reconnecting a partitioned WSN is formulated as a *Steiner tree problem with minimum Steiner points and bounded edge length (STP-MSPBEL)* in [29]. In specific, for a number of nodes to be connected in a two-dimensional Euclidean plane, some locations other than the nodes' positions are found, referred to as *Steiner points*. Then relays are placed on some Steiner points to connect all the nodes. An optimization problem is formulated to minimize the number of Steiner points needed to connect all nodes. The STP-MSPBEL problem is NP-complete. For reconnecting a partitioned WSN by using relay nodes, approximate algorithms are given in [4], [9]–[11], [29]. In [10], regular sensors and relay nodes have different transmission ranges. In [11], relay nodes are required to be placed at a subset of predetermined candidate positions. The works in [4], [11] focus on fault-tolerant networks with the help of relay nodes. In all these works, the main task is to connect a number of target sensors rather than a number of isolated blocks (each being a connected sub-network including multiple sensors).

A WSN may be partitioned into multiple blocks if large-scale damages happen. The partitioning of WSNs and restoration techniques are comprehensively surveyed in [2]. Here, we only review several related works using relays, and readers can refer to [2] for more information. The works in [5], [7], [8], [30] investigate how to repair a partitioned WSN through relay node placement. A block is modeled as a single node in [7], [8], [30], which may simplify the analysis for relay node placement. However, more relay nodes than necessary are likely to be placed [5]. On the other hand, in [5], a block is modeled as an entity with a particular shape and size, and different blocks have different shapes and sizes. Different from these existing works that *place* relay nodes and focus on minimizing the number of relay nodes placed, we consider *scheduling* mobile relays (since it may not be feasible to *place* relay nodes at any positions in hostile or hazardous environments) and focus on minimizing the total movement cost of mobile relays to reconnect a partitioned WSN.

Mobile sensors have been widely used in many applications of WSNs. The work in [14] surveys the hardware and dispatch software (or algorithms) of mobile sensors in detail. The problem of using node mobility to achieve connectivity restoration is investigated in [24], [31], [32]. In these works, wireless sensor and actor networks (WSANs) are considered, in which mobile sensors or actors are powerful devices with moving capabilities, and can perform tasks such as survivor search and fire extinguishing. If one or multiple actors fail, the WSAN may be disconnected, which can be repaired by subsequent movements of some actors. In [6], the problem of reconnecting a partitioned WSAN is modeled as a mixed integer linear program problem. In these research works, the focus is to rebuild connection *among mobile sensors (or actors)*. Different from these research efforts, we focus on maintaining the connectivity of sensor nodes by scheduling mobile relays.

The work in [23] investigates a class of movement problems. All nodes, viewed as vertices of a graph, have moving capabilities. It is required that the nodes move along the edges of the graph. In contrast, we do not have such a restriction in our problem. The works in [25], [26] focus on fault-tolerant networks (such as bi-connected networks) by exploiting node mobility, assuming that all nodes can move with unlimited moving capabilities. In our work, we also consider establishing a fault-tolerant network. But we assume that only a limited number of nodes (i.e., mobile relays) in our considered network can move and they have different and limited moving capabilities.

III. Network Model

Consider a WSN with regular static sensors and mobile relays. A sensor is to collect information in its neighborhood, send the data to its next-hop node to the BS, and also help other sensors to forward their data to the BS. The mobile relays are to maintain the connectivity of the WSN. Therefore, the mobile relays will keep inactive if the sensors themselves can form a connected network. When large-scale damages happen, some sensors and mobile relays are damaged, and the WSN is partitioned into multiple isolated blocks. Then some mobile relays (which are not damaged) will be activated and move to locations between the isolated blocks, to serve as routers (i.e., to help forward traffic between the blocks) such that the blocks are reconnected. Since mobile relays are more expensive, we assume the number of mobile relays is much smaller than that of regular sensors. The moving capabilities of mobile relays are limited and may be different from each other.

For a partitioned WSN, assume there are totally n sensors that function well, denoted as $s_1, s_2, ..., s_n$, and totally N mobile relays that function well, denoted as $m_1, m_2, ..., m_N$. The sets of sensors and mobile relays are thus $\mathbb{S} \overset{\triangle}{=} \{s_1, s_2, \cdots, s_n\}$ and $\mathbb{M} \overset{\triangle}{=} \{m_1, m_2 \cdots, m_N\}$ in the Euclidean plane. Note that s_i (or m_l) is also used to represent the position of sensor s_i (or mobile relay m_l). Denote r and R as the transmission ranges of a sensor and a mobile relay, respectively, with $r \leq R$. For two sensors (say sensors s_i and s_j), if their distance[1] denoted as $\|s_i s_j\|$ is not more than r, they can communicate directly with each other. For a mobile relay to communicate directly with a sensor or another mobile relay, their distance should be not more than r or R, respectively.

Consider that the set of sensors $\mathbb{S}$ is partitioned into $K (\geq 2)$ isolated blocks denoted as $\mathbb{B}_1$, $\mathbb{B}_2$, ..., and $\mathbb{B}_K$. The sensors in each block form a connected sub-network. The BS is assumed to be in one of the blocks. Therefore, it cannot receive sensed data from other blocks. Mobile relays are then used to reconnect the blocks. In specific, for two blocks, some mobile relays move to the area between the two blocks to form a new path such that any two consecutive nodes on the path can communicate directly. In other words, the new path established by the mobile relays forms a bridge between the two blocks.

Given $\mathbb{S}$, $\mathbb{M}$, $\mathbb{B}_1, \mathbb{B}_2, \cdots, \mathbb{B}_K$, our target is a mobile relay scheduling scheme that can reconnect all the blocks with the minimal total movement cost. The research problem is challenging due to the following reasons. First, for each mobile relay, the number of possible new positions is infinite. Second, the mobile relays have limited and different moving capabilities. Third, although it may be possible to move some mobile relays to simultaneously connect three or more blocks, the computational complexity is huge. Therefore, in this work, the purpose of any movement of any mobile relay is to connect two particular blocks.

IV. Results Regarding Connecting Two Isolated Blocks and Problem Formulation for Connecting a Partitioned WSN including Multiple Isolated Blocks

A. Boundary Sensors

First consider reconnecting two particular blocks, say blocks $\mathbb{B}_i$ and $\mathbb{B}_j$. One simple way to reconnect the two blocks is to select two sensors, each from a block, and move mobile relays to be along the line segment between the two sensors. Denote the two sensors as s_i and s_j from blocks $\mathbb{B}_i$ and $\mathbb{B}_j$, respectively, and denote their distance as $\|s_i s_j\|$. Since the transmission ranges of a sensor and a mobile relay are r and R, respectively, with $R \geq r$, the minimum number of mobile relays needed to connect s_i and s_j, denoted as $L(s_i, s_j)$, is given as

$$L(s_i, s_j) = \begin{cases} 1, & \text{if} \quad r < \|s_i s_j\| \leq 2r \\ 1 + \left\lceil \frac{\|s_i s_j\| - 2r}{R} \right\rceil, & \text{if} \quad \|s_i s_j\| > 2r \end{cases}, \quad (1)$$

in which $\lceil \cdot \rceil$ is the ceiling function. When $r < \|s_i s_j\| \leq 2r$, only one mobile relay is needed to connect s_i and s_j. When $\|s_i s_j\| > 2r$, one method to deploy the minimum number of mobile relays is[2]: first deploy two mobile relays to be along the line segment between s_i and s_j such that the two mobile relays' distances to s_i and s_j, respectively, are both r; then, to connect the two mobile relays, $\lceil (\|s_i s_j\| - 2r)/R \rceil - 1$ more mobile relays are needed.

[1]In this paper, a distance means an Euclidean distance.

[2]Note that the method to deploy the minimum number of mobile relays to connect s_i and s_j is not unique.

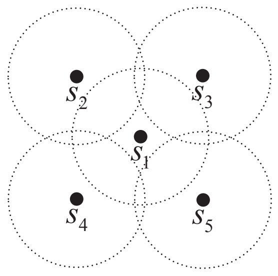


Fig. 1. Example of boundary sensors.

Intuitively, for connecting the two blocks $\mathbb{B}_i$ and $\mathbb{B}_j$, sensors on the *boundaries* of the two blocks should be selected as s_i and s_j, to use as few mobile relays as possible to connect s_i and s_j. In the literature, the works in [33]–[35] provide methods to approximately discover the boundary of a network, as well as some work [5] that places relays to connect boundary sensors such that a partitioned WSN is reconnected. However, an accurate definition of boundary nodes is not provided in these works.

Here we give a definition for boundary sensor as follows: s_i *is said to be a non-boundary sensor if and only if the perimeter of the communication area*[3] *of* s_i *is completely covered by communication areas of nearby sensors; otherwise,* s_i *is a boundary sensor.* Fig. 1 shows an example of five sensors, in which the five circles are perimeters of the communication areas of the five sensors. For s_1, its circle is completely covered by communication areas of other sensors. So it is a non-boundary sensor. For any of s_2, s_3, s_4, s_5, its circle is not completely covered by communication areas of other sensors. So the four sensors are boundary sensors.

As another example, Fig. 2 shows a partitioned WSN with eight blocks in a 2000m $\times$ 2000m square area. It is observed from the figure that the boundary sensors of a block can roughly depict the shape of the block.

[3]Communication area of a sensor is the area within a circle (the center of the circle is the target sensor, and the radius of the circle is r).

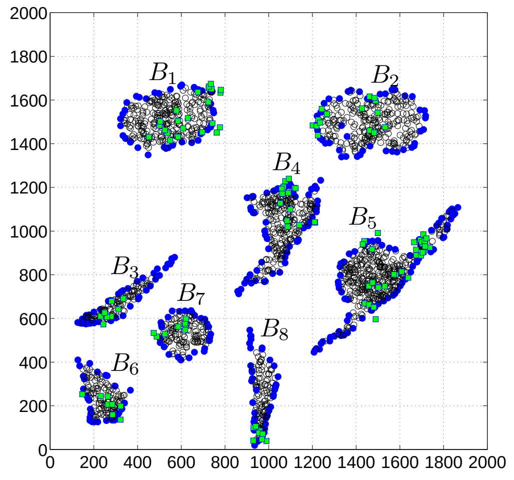


Fig. 2. Example of a partitioned WSN. Solid blue circles, empty black circles, and solid green squares are boundary sensors, non-boundary sensors, and mobile relays, respectively.

Next we show that, to use as few mobile relays as possible to connect two blocks, it is optimal to connect a pair of boundary sensors, each from one block. Recall that the minimum number of mobile relays needed to connect two sensors, say s_i and s_j from blocks $\mathbb{B}_i$ and $\mathbb{B}_j$, is denoted as $L(s_i, s_j)$ given in (1). The following theorem is in order.

Theorem 1: Consider two blocks $\mathbb{B}_i$ and $\mathbb{B}_j$ and two sensors $s_i \in \mathbb{B}_i$ and $s_j \in \mathbb{B}_j$. If one (or both) of s_i and s_j is (are) non-boundary sensor(s) of the corresponding block(s), then there exist a pair of boundary sensors from the two blocks, denoted as $s_i' \in \mathbb{B}_i$ and $s_j' \in \mathbb{B}_j$, such that $L(s_i', s_j') \leq L(s_i, s_j)$.

Proof: See the Appendix.

B. Optimal New Positions of Mobile Relays for Connecting Two Blocks

Theorem 1 shows that, to connect two blocks $\mathbb{B}_i$ and $\mathbb{B}_j$, we should only consider connecting two boundary sensors from them. Although the minimum number of mobile relays to connect two boundary sensors s_i and s_j is given in (1), there are still infinite new positions of those $L(s_i, s_j)$ mobile relays. Since the mobile relays serve as routers between the two blocks and the two boundary sensors s_i and s_j are gateways of the two blocks, it is desired to minimize the total energy consumption of the mobile relays and the gateways in forwarding traffic between the two blocks. Here we adopt the energy consumption model provided by [36], and denote

$$f(d) \triangleq cd^{\alpha} \tag{2}$$

as the energy consumption of sending one bit information, in which c is a constant value, d is the distance from the sender to the receiver, and α is path loss exponent with value varying from 2 to 4 (determined by the propagation environment). One justification for this model is that for a given wireless signal from the sender, the received power level at the receiver is proportional to $d^{-\alpha}$. Thus, in our system, to guarantee a given acceptable received power level, the transmission power level should be proportional to d^{α}.

Theorem 2: Consider two blocks $\mathbb{B}_i$ and $\mathbb{B}_j$, and two boundary sensors $s_i \in \mathbb{B}_i$ and $s_j \in \mathbb{B}_j$. Denote $\kappa \triangleq L(s_i, s_j)$ with $L(s_i, s_j)$ given in (1) as the minimum number of mobile relays needed to connect s_i and s_j. And denote $\mathbb{P}_{s_i s_j} = \{p_1, p_2, \cdots, p_\kappa\}$ as the set of the κ mobile relays' optimal new positions that, when connecting the two blocks $\mathbb{B}_i$ and $\mathbb{B}_j$, minimize the total transmission energy consumption of the mobile relays and gateways (s_i and s_j). Also denote s_i and s_j as p_0 and $p_{\kappa+1}$, respectively. We have: $p_1, p_2, \cdots, p_\kappa$ are on the line segment between s_i and s_j, and

- if $(\kappa+1)r \geq \|s_i s_j\| > r$, then $\|p_\ell p_{\ell+1}\| = \|s_i s_j\| / (\kappa + 1)$, $l = 0, 1, ..., \kappa$;
- if $(\kappa-1)R + 2r \geq \|s_i s_j\| > (\kappa+1)r$, then $\|p_0 p_1\| = \|p_\kappa p_{\kappa+1}\| = r$, $\|p_\ell p_{\ell+1}\| = (\|s_i s_j\| - 2r)/(\kappa - 1)$, $l = 1, 2, ..., \kappa - 1$.

Proof: See the Appendix.

Note that, given the positions of s_i and s_j, the minimum number of mobile relays needed to connect s_i and s_j, denoted as κ, $\kappa \geq 1$, can be calculated by (1). Theorem 2 determines the optimal positions of these κ mobile relays, i.e., $\{p_1, p_2, \cdots, p_\kappa\}$. From (1), it follows $(\kappa - 1)R + 2r \geq \|s_i s_j\| > r$. Since $R \geq r$, the range of $\|s_i s_j\|$ can be divided into two parts: $(\kappa + 1)r \geq \|s_i s_j\| > r$ and $(\kappa - 1)R + 2r \geq \|s_i s_j\| > (\kappa + 1)r$, which correspond to the two cases in Theorem 2, respectively.

C. Mobile Relay Scheduling Problem for Connecting a Partitioned WSN including Multiple Blocks

In Section IV-B, we have discussed how to connect two particular blocks in a partitioned WSN. Since our major target is to make the partitioned WSN become connected, we have the following formulation of mobile relay scheduling problem.

Given boundary sensors and mobile relays in a partitioned WSN, the mobile relay scheduling (MRS) problem is to select some pairs of boundary sensors between blocks, and move mobile relays to positions determined by Theorem 2 to connect those selected boundary sensor pairs such that the partitioned WSN is connected with minimal movement cost. Here movement cost is defined as movement distance. This problem is NP-complete as indicated by the following theorem.

Theorem 3: The MRS problem is NP-complete.

Proof: See the Appendix.

Since the MRS problem is NP-complete, next we focus on heuristic algorithms.

V. Mobile Relay Scheduling Algorithms to Reconnect a Partitioned WSN

A. One Possible Approach and its Limitations

One possible approach to reconnect a partitioned WSN is as follows: 1) compute the minimum distance between any two blocks (which equals the distance of the closest pair of boundary sensors from the two blocks); 2) establish a minimum spanning tree (MST) that connects all the blocks; 3) move mobile relays such that they are placed along the edges of the MST. The basic idea of this MST-based algorithm is to reconnect all the blocks by using as few mobile relays as possible, which is similar to [5], [9], [10], [29].

Fig. 3(a)&(b) show a simple example for the MST-based algorithm, to reconnect a partitioned WSN including four blocks $\mathbb{B}_1$, $\mathbb{B}_2$, $\mathbb{B}_3$, and $\mathbb{B}_4$ shown in Fig. 3(a). To be clear, only boundary sensors (solid blue circles) and mobile relays (solid green squares) are shown in the figure. The MST-based algorithm first constructs a complete graph $\mathbf{G}$ of all boundary sensors. In the complete graph, an edge (s_i, s_j) has weight zero if the two boundary sensors s_i and s_j are in the same block, or has weight equal to $\|s_i s_j\|$ otherwise. Then an MST is constructed as follows. Four subtrees spanning the boundary sensors of the four blocks are constructed. Note that since the weight of an edge between two boundary sensors in the same block is zero, the subtree spanning the boundary sensors in a block is not unique. Here, to be clear, the four subtrees are assumed to be four paths[4], as shown in Fig. 3(b). Since blocks $\mathbb{B}_1$ and $\mathbb{B}_2$ have the minimal distance of 4 units (two boundary sensors from them have distance of 4 units), the corresponding edge of the two boundary sensors is added to the MST, as shown in Fig. 3(b). Now blocks $\mathbb{B}_1$ and $\mathbb{B}_2$ are reconnected, to be viewed as a single block denoted as $\mathbb{B}_1\&\mathbb{B}_2$. Following similar procedure, we add an edge of weight 5 to the MST, which connects blocks $\mathbb{B}_1\&\mathbb{B}_2$ with block $\mathbb{B}_3$, forming a reconnected block $\mathbb{B}_1\&\mathbb{B}_2\&\mathbb{B}_3$; and then add an edge of weight 6 to the MST, which connects block $\mathbb{B}_4$ with block $\mathbb{B}_1\&\mathbb{B}_2\&\mathbb{B}_3$. Then, for each edge with nonzero weight in the MST (i.e., an edge connecting two blocks), the number of mobile relays needed to connect the two corresponding boundary sensors can be determined based on equation (1), and the new positions of mobile relays can be determined based on Theorem 2, as shown by the stars in Fig. 3(b).

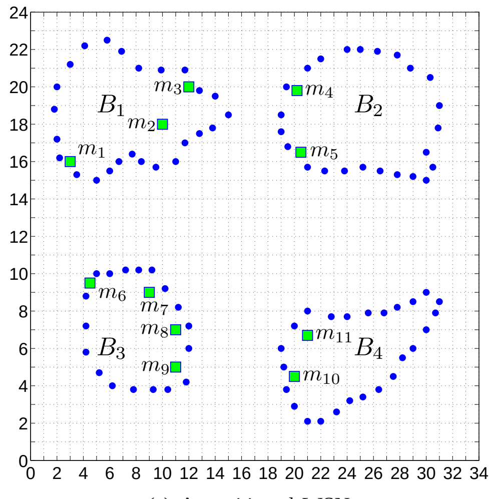
Given the new positions of mobile relays, to find a mobile relay scheduling scheme with minimal total movement cost, we can use a bipartite matching [37]. A bipartite graph $\mathbf{G}_b = (\mathbb{V}_b, \mathbb{E}_b)$ is constructed such that

- for the vertex set $\mathbb{V}_b$, we have: $\mathbb{V}_b = \mathbb{C} \cup \mathbb{P}$, in which $\mathbb{P}$ is the set of new relay positions, and $\mathbb{C}$ is the set of mobile relays that are capable of moving to a position in $\mathbb{P}$;
- for the edge set $\mathbb{E}_b$, we have: if a mobile relay $m_k \in \mathbb{C}$ is capable of moving to $p_l \in \mathbb{P}$, then there is an edge in $\mathbb{E}_b$ between m_k and p_l, and the cost of the edge is the movement distance of m_k to p_l.

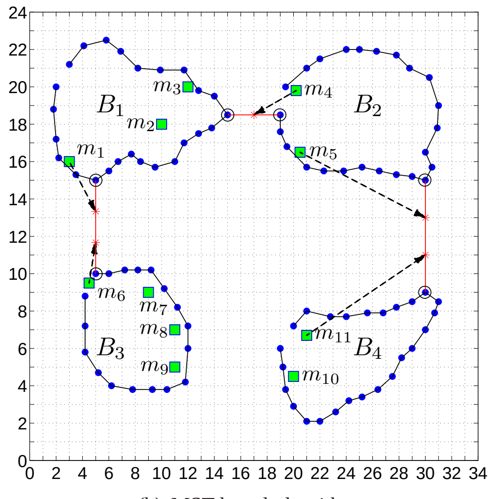
In the bipartite graph, we have two groups of vertices: the group of $\mathbb{C}$ and the group of $\mathbb{P}$. Each edge in the bipartite graph has two ends, each at one group. An edge represents that a mobile relay is capable of moving to one position in $\mathbb{P}$. We define a *matching* as a subset of edges such that each vertex in $\mathbb{P}$ has one and only one associated edge (which means that one and only one mobile relay is needed to move to each position in $\mathbb{P}$) and each vertex in $\mathbb{C}$ has up to one associated edge (which means that a mobile relay can move to at most one position). The cost of the matching is the sum of cost of all edges in the matching, which is actually the total movement distance of mobile relays. It can be seen that a matching represents a moving schedule of mobile relays to the target new positions in $\mathbb{P}$. The optimal movement scheme with the minimal cost can be obtained by solving the bipartite matching problem. As shown in Fig. 3(b), in the scheduling, mobile relays m_1, m_4, m_5, m_6 and m_{11} are moved to new positions.

The MST-based algorithm reconnects two blocks by using two nearest boundary sensors. In other words, it takes the blocks' shapes into account, and can achieve connectivity by using as few mobile relays as possible. However, it does not take into account the distribution of mobile relays. For the example shown in Fig. 3(b), to reconnect blocks $\mathbb{B}_2$ and $\mathbb{B}_4$ through the two selected boundary sensors, we need to move two mobile relays m_5 and m_{11}, which are far away from their new positions, and thus, the movement distance of the two mobile relays is large. In some extreme cases, it may be possible that no mobile relays are capable of moving to the new positions, because the moving capabilities of mobile relays are limited. Therefore, a good mobile relay scheduling should consider the shapes of blocks, the distribution of mobile

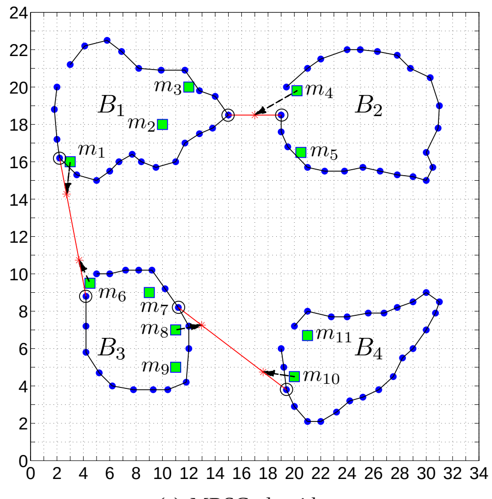
[4]Here a path is a sequence of nodes that are connected one after another.



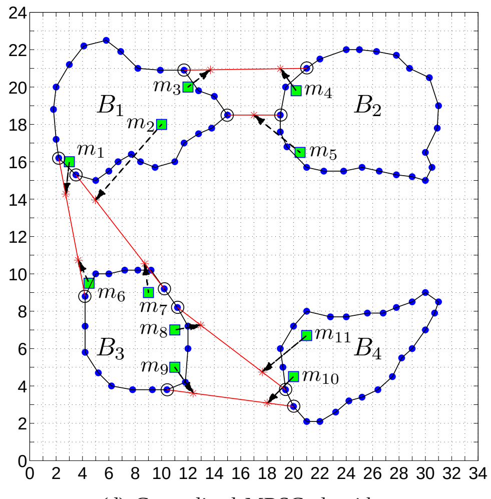
(a) A partitioned WSN



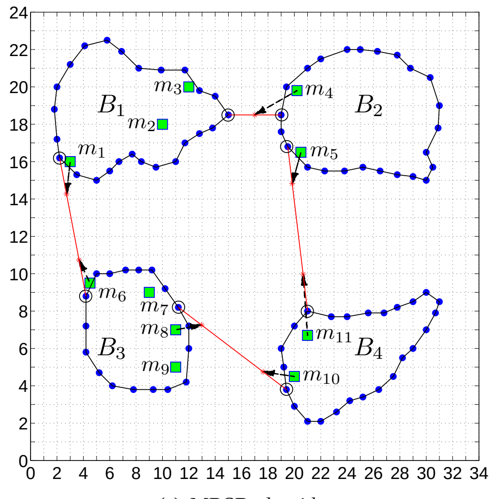
(b) MST-based algorithm



(c) MRSC algorithm



(d) Generalized MRSC algorithm



(e) MRSB algorithm

Fig. 3. A simple example of reconnecting a partitioned WSN by different algorithms ($r = 2$; $R = 6$, a star represents a new position for mobile relay, and an arrow from a mobile relay to a new position means the movement of the mobile relay).

relays, and moving capabilities of mobile relays.

B. Proposed Algorithm for Making the Network Connected

We propose an algorithm called MRSC (Mobile Relay Scheduling for making network Connected). The main idea is still to construct an MST of a graph. But different from the MST-based algorithm that selects pairs of boundary sensors that are close to each other, MRSC algorithm selects boundary sensor pairs that can be connected by mobile relays *with short movement distance*. We define the cost of reconnecting a pair of boundary sensors from two blocks as the movement cost of mobile relays. Thus, a method to calculate the movement cost in connecting a boundary sensor pair is needed.

1) Approximation of movement cost in connecting a boundary sensor pair: To connect two boundary sensors s_i and s_j from two blocks, the new positions of mobile relays, denoted as $\mathbb{P}_{s_is_j} = \{p_1, \cdots, p_\kappa\}$ with $\kappa = L(s_i, s_j)$ being the number of mobile relays needed to connect s_i and s_j, can be determined based on Theorem 2. We can use the bipartite matching in Section V-A to find a mobile relay scheduling scheme that connects s_i and s_j with minimal total movement cost. Then the minimal total movement cost is the movement cost of connecting boundary sensor pair (s_i, s_j). Denote the bipartite graph as $\mathbf{G}_b = (\mathbb{V}_b, \mathbb{E}_b)$ in which $\mathbb{V}_b = \mathbb{C} \cup \mathbb{P}_{s_is_j}$. The complexity of computing an optimal matching is $O(|\mathbb{V}_b|\cdot|\mathbb{E}_b|\cdot\log|\mathbb{V}_b|)$, in which $|\cdot|$ denotes cardinality of a set. Assume that the number of mobile relays needed to connect a boundary sensor pair in the partitioned WSN is at most λ. Recall that the number of all mobile relays is N. Then $|\mathbb{V}_b| = O(N + \lambda)$, $|\mathbb{E}_b| = O(N\lambda)$, and the complexity to get an optimal matching for boundary sensor pair (s_i, s_j) is $O((N+\lambda)\cdot N\lambda\cdot\log(N+\lambda))$. For the partitioned WSN, we have at most n^2 boundary sensor pairs. So the worst-case complexity in getting n^2 optimal matchings for all n^2 boundary sensor pairs is $O(n^2 \cdot (N + \lambda) \cdot N\lambda \cdot \log(N + \lambda))$, which is high when n and N are large. Therefore, we use a simple algorithm, as shown in Algorithm 1, to approximately compute the movement cost of connecting two boundary sensors s_i and s_j, instead of using bipartite matching.

In the algorithm, according to Theorem 2, we get the optimal new mobile relay positions to connect the boundary sensors s_i and s_j (Line 1). Then we sort the new positions in ascending order according to their distances to the midpoint of the two boundary sensors (Line 4). According to the sorted order, we check the new mobile relay positions one by one. For each checked position, we select the available mobile relay that has not been used yet and can reach the checked position with minimal distance (Lines 6–8). If we can find a mobile relay for each new mobile relay position in $\mathbb{P}_{s_is_j}$, the movement cost of connecting s_i and s_j is approximated as the total movement distance of moving those selected mobile relays to the new mobile relay positions in $\mathbb{P}_{s_is_j}$ (Line 10); otherwise, the movement cost of connecting s_i and s_j is ∞ (Line 9).

For the considered partitioned WSN, the worst-case complexity when computing the approximate movement cost of connecting a boundary sensor pair is $O(N\lambda + \lambda \cdot \log\lambda) = O(N\lambda)$. So the worst-case complexity to compute the approximate movement costs for all boundary sensor pairs is $O(n^2N\lambda)$.

Algorithm 1: Computing approximate movement cost – **Function** ApproximateMovementCost(s_i,s_j)

```
Input: a pair of boundary sensors $s_i$ and $s_j$ in different
       blocks, the set of mobile relays
       $\mathbb{M} = \{m_1, m_2, \cdots, m_N\}$ and their maximum
       moving capabilities
Output: the approximate movement cost for connecting
        $s_i$ and $s_j$
1  $\mathbb{P}_{s_is_j} \leftarrow \{p_1, p_2, \cdots, p_\kappa\}$;
2  $\mathbb{R} \leftarrow \emptyset$ (null set);
3  $C \leftarrow 0$;
4  Sort $p_1, p_2, \cdots, p_\kappa$ in ascending order of their distances
   to the midpoint between $s_i$ and $s_j$;
5  for each $p_l$ in sorted order do
6      if $\exists m_k$: $m_k$ can reach $p_l$ and is the nearest to $p_l$
       among mobile relays in $\mathbb{M}\backslash\mathbb{R}$ then
7          $C \leftarrow C + \|m_kp_l\|$;
8          $\mathbb{R} \leftarrow \mathbb{R} \cup \{m_k\}$;
9      else return $\infty$;
10 return $C$;
```

2) The MRSC Algorithm: Based on the simplified algorithm to approximate movement cost of connecting a boundary sensor pair, the proposed MRSC algorithm is shown in Algorithm 2.

In the algorithm, a graph $\mathbf{G} = (\mathbb{V}, \mathbb{E})$ is first constructed (Line 1), in which the vertex set $\mathbb{V}$ has all boundary sensors, and the edge set $\mathbb{E}$ has pairs of boundary sensors that are in different blocks. For each edge (u, v) (boundary sensor pair) in $\mathbb{E}$, a weight $w(u, v)$ is assigned (Lines 2–3), which is the approximate movement cost for connecting u and v determined by Algorithm 1. Then another graph $\mathbf{G}'$ is constructed with vertex set being $\mathbb{V}$ but without edges (Line 5). Since all boundary sensors in the same block are connected themselves, without loss of generality, a path is constructed to connect all boundary sensors in each block, and corresponding edges are added into $\mathbf{G}'$ (Line 6). After that, from all edges in $\mathbb{E}$, according to ascending order of their weights, we check them one by one: for an edge (u, v), if u and v are not connected in $\mathbf{G}'$ (which means that the two blocks that u and v belong to are not connected in $\mathbf{G}'$), we add edge (u, v) into $\mathbf{G}'$, and include into $\mathbb{P}$ the new mobile relay positions needed to connect (u, v) (based on Theorem 2).[5] We keep checking the edges in $\mathbb{E}$ until $K - 1$ edges have been added (which means that all K blocks are connected), or all edges in $\mathbb{E}$ have been checked and $\mathbf{G}'$ is still not connected (Lines 9–14). For the latter case, there is no feasible solution. For the former case, $\mathbf{G}'$ is connected, and $\mathbb{P}$ contains the new mobile relay positions to which we should move mobile relays. Note that it may not work if we simply combine the individual movement

[5]In Line 12, $\mathbb{P}_{uv}$ is the set of new mobile relay positions to connect boundary sensors u and v, determined based on Theorem 2.

Algorithm 2: The Greedy Algorithm MRSC

```
Input: the set of sensors $\mathbb{S} = \{s_1, s_2, \cdots, s_n\}$, the set of
       mobile relays $\mathbb{M} = \{m_1, \cdots, m_N\}$ and their
       maximum moving capabilities, and the set of
       blocks $\mathbb{B}_1, \cdots, \mathbb{B}_K$
Output: a mobile relay scheduling scheme (if exists)
        which connects all blocks
1  $\mathbf{G} = (\mathbb{V}, \mathbb{E})$, $\mathbb{V} \leftarrow \{s | s \in \mathbb{S}, s$ is a boundary sensor$\}$,
   $\mathbb{E} \leftarrow \{(u, v) | u, v \in \mathbb{V}, u$ and $v$ are in different blocks$\}$;
2  for each $(u, v) \in \mathbb{E}$ do
3      $w(u, v) \leftarrow$ ApproximateMovementCost($u, v$)
       (here if $w(u, v)$ is $\infty$, then remove $(u, v)$ from $\mathbb{E}$);
4  $\mathbb{P} \leftarrow \emptyset$;
5  $\mathbf{G}' = (\mathbb{V}, \mathbb{E}')$, $\mathbb{E}' \leftarrow \emptyset$;
6  for each block, form a path connecting all boundary
   sensors and add corresponding edges into $\mathbb{E}'$;
7  sort all edges in $\mathbb{E}$ in ascending order by weight;
8  $cnt \leftarrow 0$;
9  for all $(u, v) \in \mathbb{E}$ in sorted order do
10     if $u$ and $v$ are not connected in $\mathbf{G}'$ then
11         $\mathbb{E}' \leftarrow \mathbb{E}' \cup \{(u, v)\}$;
12         $\mathbb{P} \leftarrow \mathbb{P} \cup \mathbb{P}_{uv}$;
13         $cnt \leftarrow cnt + 1$;
14     if $cnt = K - 1$ then break;
15 if $\mathbf{G}'$ is not connected then there is no feasible solution,
   and return;
16 $\mathbf{G}_b = (\mathbb{V}_b, \mathbb{E}_b)$, $\mathbb{V}_b = \mathbb{M} \cup \mathbb{P}$,
   $\mathbb{E}_b \leftarrow \{(m, p) | m \in \mathbb{M}, p \in \mathbb{P}, m$ can reach position $p\}$,
   $c(m, p) \leftarrow$the distance between $m$ and $p$, $(m, p) \in \mathbb{E}_b$.
   Here $c(m, p)$ is the movement cost (movement distance)
   of mobile relay $m$ to move to position $p$;
17 compute a maximum cardinality matching of $\mathbf{G}_b$,
   denoted as $\mathcal{M}^*$, with minimum cost, which corresponds
   to a mobile relay scheduling scheme that connects all
   blocks if $|\mathcal{M}^*| = |\mathbb{P}|$.
```

scheduling for all the added edges (boundary sensor pairs)[6], because the movement scheduling for two edges may need to move the same mobile relay to different positions. Therefore, we use a bipartite matching instead (Lines 16–17), and get a movement scheme (if it exists) that moves mobile relays to all positions in $\mathbb{P}$ with minimal movement cost (defined as movement distance).

We still take the partitioned WSN in Fig. 3(a) as an example. Four paths are constructed, respectively connecting boundary sensors of the four blocks, as shown in Fig. 3(c). Then, one edge is added between blocks $\mathbb{B}_1$ and $\mathbb{B}_3$, since the edge has the minimal approximate movement cost. Then one more edge is added to connect $\mathbb{B}_1$ with $\mathbb{B}_2$, and one other edge is added to connect $\mathbb{B}_3$ and $\mathbb{B}_4$. Then the network is connected. Finally, by bipartite matching, mobile relays m_1, m_4, m_6, m_8 and m_{10} are moved to connect the three edges. It can be seen that the total movement cost is approximately one-third of the total movement cost of the MST-based algorithm shown in Fig. 3(b).

Fig. 4 gives another example to connect the partitioned WSN shown in Fig. 2. In this example, the total movement distance in the MST-based algorithm and in the MRSC algorithm is around 3100 meters and 1850 meters, respectively.

The complexity of the MRSC algorithm is analyzed as follows. Recall that n^2 is the maximal number of boundary sensor pairs, N is the number of mobile relays, and λ is the maximum number of mobile relays to connect two boundary sensors from two blocks. The MRSC algorithm has four main operations:

- Calculating weights of boundary sensor pairs (edges) in $\mathbb{E}$, with complexity $O(n^2 N \lambda)$ as analyzed in Section V-B1;
- Sorting all edges in $\mathbb{E}$, with complexity $O(|\mathbb{E}| \cdot \log |\mathbb{E}|)$;
- Adding edges from $\mathbb{E}$ to graph $\mathbf{G}'$ until $K-1$ edges have been added, or all edges have been checked: Checking all edges in $\mathbb{E}$ costs $O(|\mathbb{E}|)$. To check whether or not a graph $\mathbf{G}' = (\mathbb{V}, \mathbb{E}')$ is connected, we can use a Depth-First Search (DFS) algorithm with complexity $O(|\mathbb{V}| + |\mathbb{E}'|)$. Therefore, the worst-case total complexity is $O(|\mathbb{E}|)$;
- Performing bipartite matching to get a movement scheme, with complexity $O(|\mathbb{V}_b| \cdot |\mathbb{E}_b| \cdot \log |\mathbb{V}_b|)$ for bipartite graph $\mathbf{G}_b = (\mathbb{V}_b, \mathbb{E}_b)$.

Overall, the complexity of the MRSC algorithm is dominated by the first operation, and is $O(n^2 N \lambda)$.

C. Proposed Mobile Relay Scheduling with Fault Tolerance

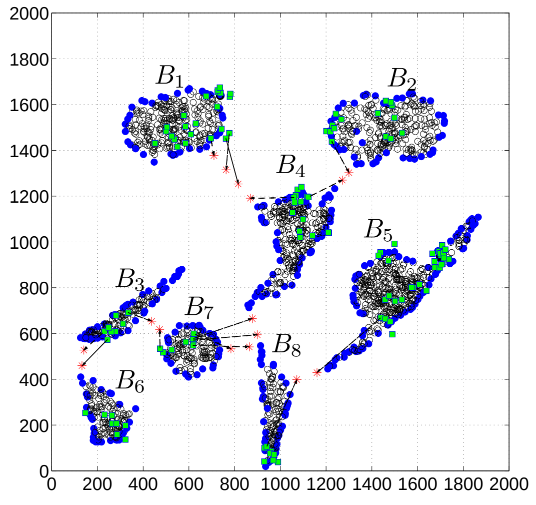
The MRSC algorithm reconnects a partitioned WSN by connecting selected boundary sensor pairs through moving mobile relays. Those selected boundary sensors are gateways of blocks, and those moved mobile relays are routers. The gateways and routers have large amounts of data to forward. If one of the gateways or routers fails (e.g., due to energy depletion), the WSN may become disconnected again. Therefore, it is desired to add fault tolerance to the rebuilt WSN. Here we hope the rebuilt WSN can tolerate failure of one gateway or one router. Next we discuss how to achieve this goal.

For a rebuilt WSN, consider a gateway of a block (say block $\mathbb{B}_i$). If the gateway is a cut-vertex of block $\mathbb{B}_i$, then failure of the gateway will disconnect block $\mathbb{B}_i$, which will also disconnect the WSN.[7] Therefore, to make the rebuilt WSN tolerant to failure of one gateway, the gateways of the blocks should be non-cut-vertices of the blocks. Thus, when we select boundary sensor pairs of the blocks to connect the partitioned WSN, we should only consider pairs of non-cut-vertex boundary sensors (i.e., boundary sensors that are not cut-vertices of the corresponding blocks). Cut-vertices of a block can be identified by a DFS algorithm.

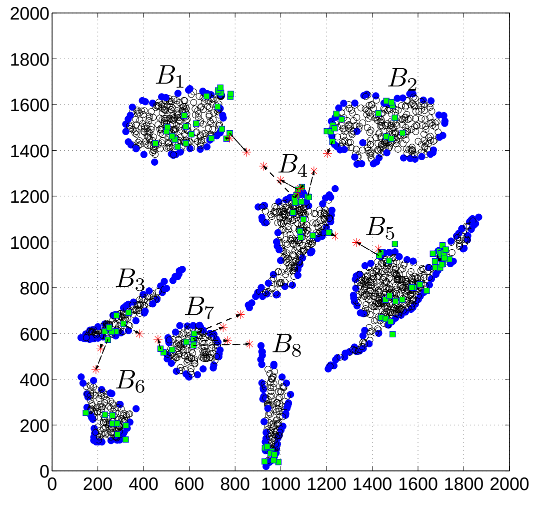
Note that non-cut-vertex boundary sensors in the same block are connected, and any single failure of them does not disconnect the block. Therefore, to simplify the description of our proposed algorithm, for each block, a virtual ring is

[6]Recalling that Algorithm 1 gives the approximate movement cost to connect two boundary sensors, as well as a movement scheduling that decides to move what mobile relays to the target new positions $p_1, p_2, ..., p_\kappa$.

[7]For a network, if removal of a node disconnects the network, the node is called a cut-vertex of the network.



(a) MST-based algorithm



(b) MRSC

Fig. 4. Reconnecting the partitioned WSN shown in Fig. 2 by using the MST-based algorithm and the proposed MRSC algorithm.

constructed to connect all non-cut-vertex boundary sensors in the block. Further, for a rebuilt WSN, we define its *virtual boundary network* as the network including all moved mobile relays and all virtual rings of the blocks. Then it can be seen that if the rebuilt WSN can tolerate failure of one gateway or one router, this equivalently means that the virtual boundary network is bi-connected.[8] Thus, our target is to have a bi-connected virtual boundary network.

For a partitioned WSN, to move mobile relays such that the virtual boundary network is bi-connected, we can have the following modifications to MRSC (Algorithm 2): the vertex set $\mathbb{V}$ in Line 1 contains all non-cut-vertex boundary sensors; Line 6 changes to "for each block, form a virtual ring connecting all non-cut-vertex boundary sensors and add corresponding edges into $\mathbb{E}'$"; Line 10 changes to: "**If** the two blocks containing u and v are not bi-connected in $\mathbf{G}'$ **then**";[9] Line 11 changes to: "$\mathbb{E}' \leftarrow \mathbb{E}' \cup \{(u,v)\}$, and remove edges in $\mathbb{E}$ that have u or v as one end[10]"; Line 14 changes to: "**If** $cnt \geq K$ and $\mathbf{G}'$ is bi-connected **then break**";[11] Line 15 changes to: "**If** $\mathbf{G}'$ is not bi-connected **then** there is no feasible solution, and **return**". The resulted algorithm is called generalized MRSC. The major idea of the generalized MRSC is that we keep adding edges until the virtual boundary network $\mathbf{G}'$ becomes bi-connected.

We apply the generalized MRSC to the partitioned WSN in Fig. 3(a). Here, we assume the boundary sensors shown in Fig. 3(a) are non-cut-vertex boundary sensors. The resulted scheduling scheme is shown in Fig. 3(d). It can be seen that two edges are added for each of the block pairs $\mathbb{B}_1\&\mathbb{B}_2$, $\mathbb{B}_1\&\mathbb{B}_3$, and $\mathbb{B}_3\&\mathbb{B}_4$. Indeed, if the generalized MRSC adds one edge between two blocks, it is very likely that one more edge between the two blocks will be added subsequently, because the two blocks probably have other non-cut-vertex boundary sensor pairs with weight similar to that of the just added edge. Therefore, the generalized MRSC algorithm adds six edges in Fig. 3(d). On the other hand, to make the virtual boundary network in Fig. 3(a) bi-connected, alternatively we may need only four edges for $\mathbb{B}_1\&\mathbb{B}_2$, $\mathbb{B}_1\&\mathbb{B}_3$, $\mathbb{B}_3\&\mathbb{B}_4$, and $\mathbb{B}_2\&\mathbb{B}_4$, respectively, to form a "ring".

It is seen that adding two edges between two blocks may result in more edges than necessary to be added. So, if an edge has been already added between two blocks, we should give lower priority to other non-cut-vertex boundary sensor pairs from the two blocks. Based on this idea, we propose a greedy algorithm called MRSB (Mobile Relay Scheduling for making virtual boundary network Bi-connected), which is shown in Algorithm 3. Different from the generalized MRSC, in Line 12, if an edge (u,v) is added into $\mathbf{G}'$ to connect two blocks, then all other boundary sensor pairs from the two blocks are moved from $\mathbb{E}$ to a lower-priority edge set $\mathbb{E}^\dagger$. After all edges in $\mathbb{E}$ are checked, if the virtual boundary network $\mathbf{G}'$ is not bi-connected, then we add edges from the lower-priority edge set $\mathbb{E}^\dagger$ until the virtual boundary network $\mathbf{G}'$ becomes bi-connected, or all edges in $\mathbb{E}^\dagger$ have been checked (Line 15–21)[12].

At the end of the above procedure, we may have added more edges than necessary to make the virtual boundary network bi-connected. Therefore, in Lines 23–26, we add a further procedure to remove the edges in $\mathbb{E}'$ one by one according to descending order of their weights until no more edges can be removed, under the condition that the virtual boundary network $\mathbf{G}'$ keeps bi-connected.

We still take the partitioned WSN in Fig. 3(a) as an example. The scheduling scheme of MRSB is shown in Fig. 3(e). Four edges are added, which form a ring. Seven mobile relays are moved, with total movement distance being 16 units. As a

[8]For a network, if any single node failure does not disconnect the network, then we say the network is bi-connected.

[9]Two blocks are said to be bi-connected in $\mathbf{G}'$ if any two nodes from them can still be connected if any single node failure happens in $\mathbf{G}'$.

[10]The reason for this removal is that for bi-connectivity, we need to add edges with no common nodes.

[11]Note that to make K blocks bi-connected, we need at least K block-to-block edges.

[12]If one edge (u,v) from the lower-priority edge set $\mathbb{E}^\dagger$ is added to $\mathbb{E}'$, it means the two blocks that u and v belong to are already connected by two edges. So it is not necessary to add other edges to connect the two blocks (Line 20 of Algorithm 3).

Algorithm 3: The Greedy Algorithm MRSB

```
Input: the set of sensors $\mathbb{S} = \{s_1, s_2, \cdots, s_n\}$, the set of
       mobile relays $\mathbb{M} = \{m_1, \cdots, m_N\}$ and their
       maximum movement capabilities, and the set of
       blocks $\mathbb{B}_1, \cdots, \mathbb{B}_K$
Output: a mobile relay scheduling scheme (if exists)
        which makes the virtual boundary network
        bi-connected
1  $\mathbf{G} = (\mathbb{V}, \mathbb{E})$,
   $\mathbb{V} \leftarrow \{s | s \in \mathbb{S}, s$ is a non-cut-vertex boundary sensor$\}$,
   $\mathbb{E} \leftarrow \{(u, v) | u, v \in \mathbb{V}, u$ and $v$ are in different blocks$\}$;
2  for each $(u, v) \in \mathbb{E}$ do
3      $w(u, v) \leftarrow$ ApproximateMovementCost($u, v$)
       (here if $w(u, v)$ is $\infty$, then remove $(u, v)$ from $\mathbb{E}$);
4  $\mathbb{P} \leftarrow \emptyset$;
5  $\mathbf{G}' = (\mathbb{V}, \mathbb{E}')$, $\mathbb{E}' \leftarrow \emptyset$, $\mathbb{E}^\dagger \leftarrow \emptyset$;
6  for each block, form a virtual ring of all non-cut-vertex
   boundary sensors and add corresponding edges into $\mathbb{E}'$;
7  sort all edges in $\mathbb{E}$ in ascending order by weight;
8  $cnt \leftarrow 0$;
9  for all $(u, v) \in \mathbb{E}$ in sorted order do
10     if the two blocks containing $u$ and $v$ are not
       bi-connected in graph $\mathbf{G}'$ then
11         $\mathbb{E}' \leftarrow \mathbb{E}' \cup \{(u, v)\}$, and remove edges in $\mathbb{E}$ that
           have $u$ or $v$ as one end;
12         Move edges in $\mathbb{E}$ that connect blocks of $u$ and $v$
           to $\mathbb{E}^\dagger$;
13         $\mathbb{P} \leftarrow \mathbb{P} \cup \mathbb{P}_{uv}$, $cnt \leftarrow cnt + 1$;
14     if $cnt \geq K$ and $\mathbf{G}'$ is bi-connected then break;
15 if $\mathbf{G}'$ is not bi-connected then
16     sort all edges in $\mathbb{E}^\dagger$ in ascending order by weight;
17     for all $(u, v) \in \mathbb{E}^\dagger$ in sorted order do
18         if $\mathbf{G}'$ is bi-connected then break;
19         $\mathbb{E}' \leftarrow \mathbb{E}' \cup \{(u, v)\}$, and remove edges in $\mathbb{E}^\dagger$ that
           have $u$ or $v$ as one end;
20         Remove edges in $\mathbb{E}^\dagger$ that connect blocks of $u$ and
           $v$;
21         $\mathbb{P} \leftarrow \mathbb{P} \cup \mathbb{P}_{uv}$;
22 if $\mathbf{G}'$ is not bi-connected then there is no feasible
   solution, and return;
23 sort all edges in $\mathbb{E}'$ in descending order of their weights;
24 for all $(u, v) \in \mathbb{E}'$ in sorted order do
25     if $\mathbf{G}' = (\mathbb{V}, \mathbb{E}')$ is still bi-connected after removal of
       edge $(u, v)$ then
26         $\mathbb{E}' \leftarrow \mathbb{E}' \backslash \{(u, v)\}$, $\mathbb{P} \leftarrow \mathbb{P} \backslash \mathbb{P}_{uv}$;
27 $\mathbf{G}_b = (\mathbb{V}_b, \mathbb{E}_b)$, $\mathbb{V}_b = \mathbb{M} \cup \mathbb{P}$,
   $\mathbb{E}_b \leftarrow \{(m, p) | m \in \mathbb{M}, p \in \mathbb{P}, m$ can reach position $p\}$,
   $c(m, p) \leftarrow$the distance between $m$ and $p$, $(m, p) \in \mathbb{E}_b$.
   Here $c(m, p)$ is the movement cost (movement distance)
   of mobile relay $m$ to move to position $p$;
28 compute a maximum cardinality matching of $\mathbf{G}_b$,
   denoted as $\mathcal{M}^*$, with minimum cost, which corresponds
   to a mobile relay scheduling scheme that results in a
   bi-connected virtual boundary network if $|\mathcal{M}^*| = |\mathbb{P}|$.
```

comparison, for generalized MRSC in Fig. 3(d), eleven mobile relays are moved, with total movement distance being 29 units.

For the computation complexity, MRSB differs from MRSC in that it needs to check whether or not a graph is bi-connected, which can also be done by a DFS algorithm. So MRSB has the same complexity as MRSC.

D. Discussion

Implementation: Like the algorithms in [4], [5], [7]–[11], [30], MRSC and MRSB belong to centralized algorithms. These algorithms all need the information of the partitioned networks. To collect the information, one possible approach is to use a powerful *mobile robot*, for example, an unmanned aerial vehicle (UAV) [38]–[40]. In particular, to facilitate information collection, a rendezvous-based method [41] can be used: At each block, some nodes are assigned the roles of rendezvous points, and collect information of the block. Then, a UAV can be sent to pass by the blocks, collect information from those rendezvous points, run the proposed MRSC or MRSB algorithm, and inform rendezvous nodes of the movement decisions. Then rendezvous nodes pass the movement decisions to mobile relays in the corresponding blocks.

Another possible approach to collect information of the partitioned network is to dispatch mobile nodes to collect information in a distributed manner. For example, if a WSN is partitioned due to large-scale damages, a block does not have information of the number of nearby blocks or their locations. In [42], each block sends mobile nodes, which carry information of the block and move to a predetermined point (e.g., the center of the deployment area). When these mobile nodes "meet" at the predetermined point, the information of the partitioned network can be obtained, movement decisions can be made and sent back to the blocks. This method can also be adopted by MRSC and MRSB to collect information and send back movement decisions.

Tolerance to any single node failure: Our MRSB algorithm makes the rebuilt WSN tolerant to the failure of one gateway or one router. Actually our MRSB algorithm can be extended to make the rebuilt WSN tolerant to any single node failure, as follows. Suppose the sensors in block $\mathbb{B}_i$ are connected but not bi-connected, which means block $\mathbb{B}_i$ has cut-vertices. Then Block $\mathbb{B}_i$ can be viewed as a number of bi-connected sub-blocks (called bi-connected components) that are connected by cut-vertices of block $\mathbb{B}_i$. The bi-connected components and cut-vertices of a block can be identified by a DFS algorithm. If a block is bi-connected, it has a single bi-connected component. If we treat a "bi-connected component" as a "block", we can use MRSB to connect those bi-connected components. Note that since bi-connected components of a same block are connected by cut-vertices of the block, there are existing edges between the bi-connected components. So in the MRSB algorithm, when determining whether or not graph $\mathbf{G}'$ is bi-connected, these existing edges can be considered in graph $\mathbf{G}'$, which can reduce the number of mobile relays that need to be moved. Since all bi-connected components are already bi-connected, the rebuilt WSN can tolerate the failure of any node. In other words, the rebuilt WSN is bi-connected. The details are omitted due to space limit.

VI. Performance Evaluation

A. Algorithms for comparison

We conduct extensive computer simulations to evaluate the performance of the proposed MRSC and MRSB algorithms, by using a customized C++ simulator with a computational geometry library [43]. We also make comparison with three existing algorithms in the literature that make a disconnected WSN become connected or bi-connected, as follows.

- Connected Inter-Segment Topology (CIST) [5]: CIST strives to use fewer relays to reconnect a partitioned WSN. The major idea is to select between two methods that connect three sensors in different blocks: 1) to place relays along two edges among the three sensors, or 2) to first place a relay on the *Fermat Point* of the triangle formed by the three sensors, and then place relays along the edges from the Fermat Point to the three sensors. The method that uses fewer relays is selected. To apply CIST in our considered problem, after the positions of relays are determined by CIST, we move mobile relays to the relay positions with minimum total movement distance by computing an optimal bipartite matching (as in Lines 16–17 of our Algorithm 2).
- MST based Bi-connected subgraph (MSTB) [4]: MSTB makes a number of nodes k-connected by adding extra edges sequentially. Since this algorithm focuses on nodes instead of blocks, we modify this algorithm as follows to make the virtual boundary network in our considered WSN become bi-connected. We apply the algorithm in [4] to determine the edges to be added such that the virtual boundary network becomes bi-connected (i.e., $k = 2$). We then apply a further procedure (Lines 23–26 of our Algorithm 3) to remove redundant edges while keeping the virtual boundary network bi-connected. After that, for each added edge, we use our Theorem 2 to determine the relay positions, and move mobile relays to the relay positions with minimum total movement distance by computing an optimal bipartite matching (as in Lines 27–28 of our Algorithm 3).
- Connected Restoration with Assured Fault Tolerance (CRAFT) [30]: CRAFT first selects a sensor (called gateway) in each block to represent the block. Then relays are placed into the network such that the gateways and relays form a bi-connected network. In other words, the rebuilt network by CRAFT can tolerate the failure of a gateway or a router (relay). CRAFT consists of two phases. In the first phase, a convex polygon is formed by some selected Steiner points and gateways such that all other gateways are outside the convex polygon. The convex polygon is called a backbone polygon (BP). Then relays are placed on the edges of the BP. In the second phase, extra relays are placed to make each outside gateway have two disjoint paths to the BP. For our considered problem, we apply CRAFT to determine the relay positions, and move mobile relays to the relay positions with minimum total movement distance by computing an optimal bipartite matching (as in Lines 27–28 of our Algorithm 3).

B. Simulation Setup

In the simulations, we consider a partitioned WSN that is deployed in a 2000m $\times$ 2000m square area. The transmission radius of a sensor and a mobile relay is 40 m and 100 m, respectively. The moving capabilities (maximum movement distance) of mobile relays are independent and randomly selected from the range [100m, 300m].

The following four performance metrics are used in the simulations for different algorithms: success probability in making network connected (for MRSC and CIST) or making the virtual boundary network bi-connected (for MRSB, MSTB, and CRAFT), the total movement distance of mobile relays, the number of moved mobile relays, and the maximum movement distance of a mobile relay.

In each simulation run, sensors and mobile relays are uniformly deployed in the 2000m $\times$ 2000m square area, and initially, sensors form a connected network. For the initial deployed WSN, denoting the number of sensors in it as A, we randomly deploy $\lceil \eta A \rceil$ mobile relays in the deployment area, in which η is referred to as the *mobile relay percentage*. Then, we randomly fail some nodes including sensors and mobile relays until the network becomes partitioned into K blocks (partitions) with $K = 2, 3, ..., 9$.

In the simulations, the value of the mobile relay percentage varies from 5%, 10%, to 15%. The simulation statistics are collected in 1000 simulation runs, and 95% confidence intervals are shown for statistical data.

C. Simulation Results

Since the number of mobile relays is limited and the moving capabilities of the mobile relays are also limited, it is possible that an algorithm may fail to make the partitioned WSN connected or make the virtual boundary network bi-connected. Figs. 5-6 show the success probabilities of the five algorithms when the number of blocks varies from 2 to 9 and the mobile relay percentage varies from 5%, 10%, to 15%. The success probabilities of the five algorithms increase when the mobile relay percentage increases, and tend to decrease when there are more blocks. For making the partitioned WSN connected, MRSC has a higher success probability than CIST. This is because MRSC takes into account distribution and moving capabilities of mobile relays. This feature of MRSC makes it have fairly high success probability even when the mobile relay percentage is low. For making the virtual boundary network bi-connected, MRSB has a higher success probability than MSTB and CRAFT. CRAFT cannot make the virtual boundary network bi-connected when the mobile relay percentage is 5% and the number of blocks is more than 6. This is mainly because the shapes of blocks are neglected by CRAFT during selecting relay positions. MSTB considers the shapes of blocks when choosing relay positions. On the other hand, our proposed MRSB takes into account the shapes of blocks as well as the distribution and moving capabilities of mobile relays. Further, MRSB uses an efficient way to

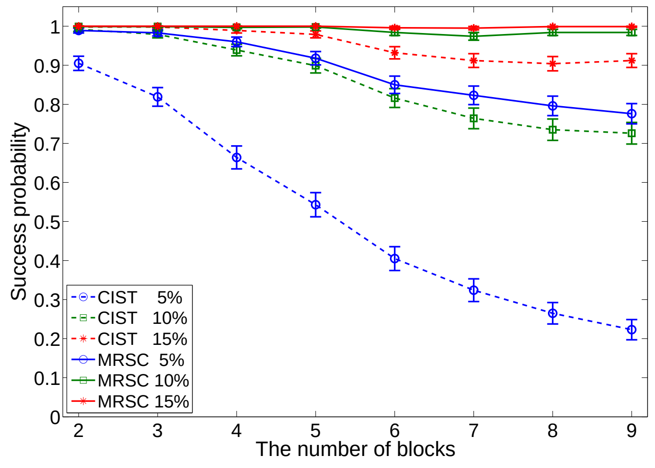


Fig. 5. Success probability of CIST and MRSC algorithms.

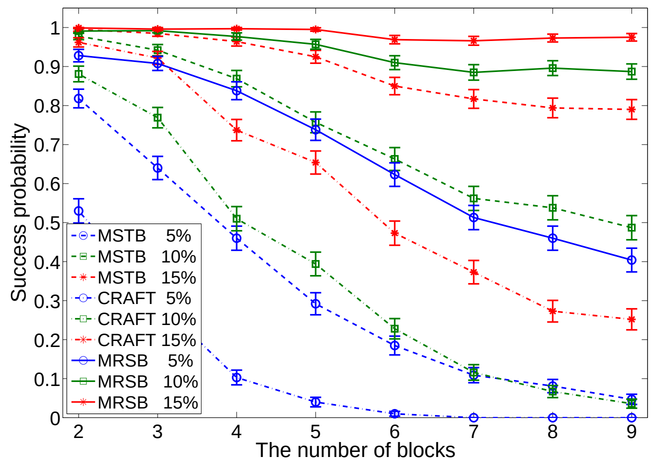


Fig. 6. Success probability of CRAFT, MSTB and MRSB algorithm.

make the virtual boundary network bi-connected by avoiding adding two edges between two blocks if possible. Therefore, MRSB achieves the highest success probability among the three algorithms.

Next we show other performance metrics (total movement

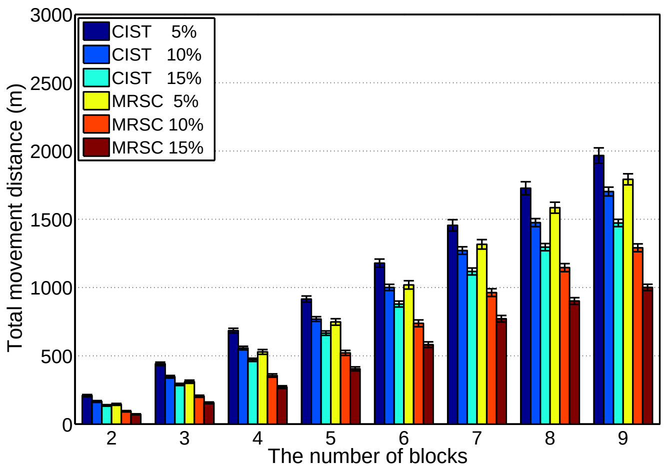


Fig. 7. Total movement distance of CIST and MRSC algorithms.

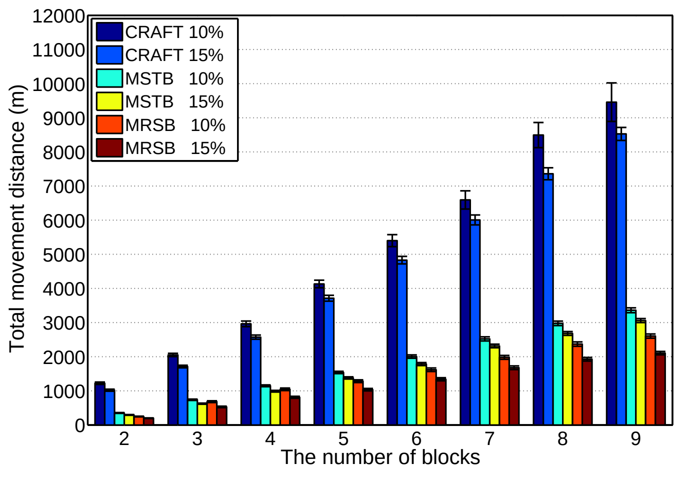


Fig. 8. Total movement distance of CRAFT, MSTB and MRSB algorithms.

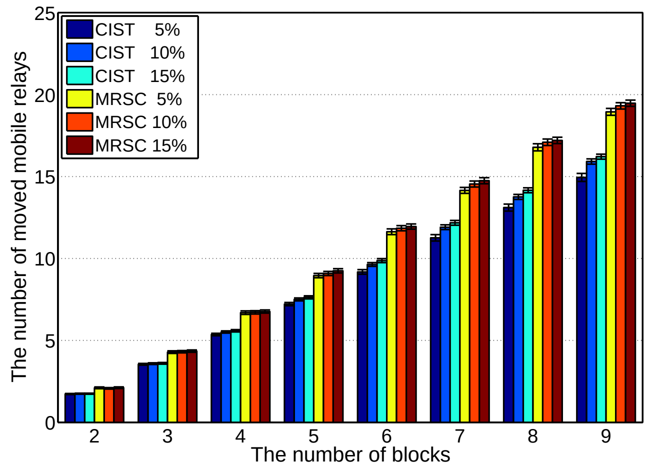


Fig. 9. Number of moved mobile relays in CIST and MRSC algorithms.

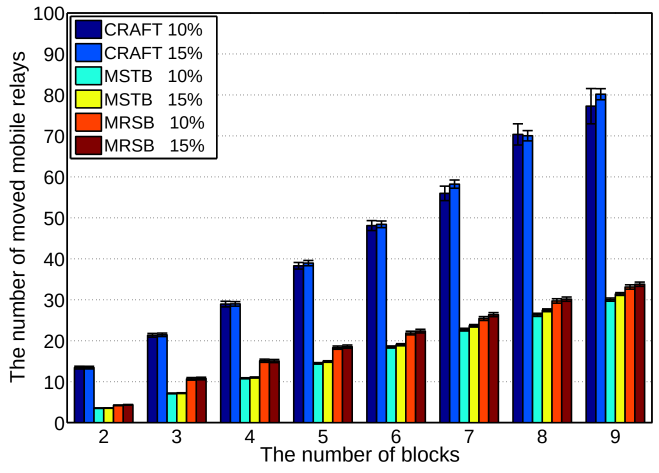


Fig. 10. Number of moved mobile relays in CRAFT, MSTB and MRSB algorithms.

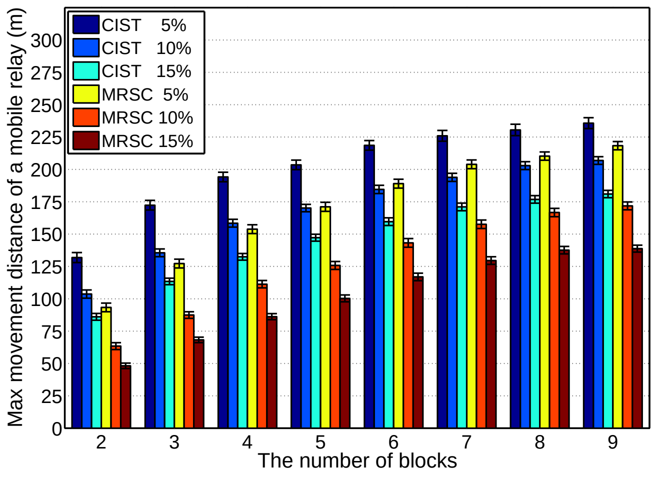


Fig. 11. Maximum movement distance of a mobile relay in CIST and MRSC algorithms.

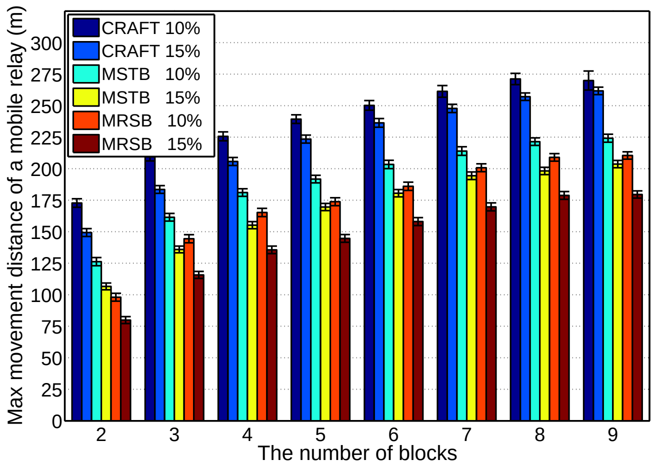


Fig. 12. Maximum movement distance of a mobile relay in CRAFT, MSTB and MRSB algorithms.

distance, number of moved mobile relays, and maximum movement distance of a mobile relay). For each algorithm, the statistics are collected only for simulation runs in which the algorithm is successful.

Figs. 7-8 show the total movement distance of the five algorithms with different mobile relay percentage and different number of blocks. Since the successful probability of CRAFT and MSTB are not high when the mobile relay percentage is 5% and the number of blocks is more than 6, we only show the total movement distance of CRAFT, MSTB, and MRSB when the mobile relay percentage is 10% and 15%. It can be seen that, for each algorithm, the movement distance is less with a higher mobile relay percentage because, with a larger mobile relay pool, mobile relays at better positions can be selected. The movement distance is higher with more blocks since more relays are needed to connect more blocks. From Fig. 7, MRSC has much smaller total movement distance than that of CIST. Specifically, the total movement distance of MRSC is 17.7%, 31.2%, and 37.9% lower than that of CIST when the mobile relay percentage is 5%, 10%, and 15%, respectively. From Fig. 8, for making the virtual boundary network bi-connected, the total movement distance of MRSB is 18.3% and 25.3% lower than that of MSTB, and 70.6% and 72.9% lower than that of CRAFT, when the mobile relay percentage is 10% and 15%, respectively. These movement cost savings are also because the distribution and moving capabilities of mobile relays are taken into account by MRSC and MRSB, and further, a better way to reconnect blocks is adopted by MRSB.

Figs. 9-10 show the number of moved mobile relays of the five algorithms. CIST (or MSTB) targets at a small number of mobile relays used in making the network connected (or making the virtual boundary network bi-connected), whereas the proposed MRSC (or MRSB) targets at low movement cost. It can be seen that CIST (or MSTB) uses fewer mobile relays than MRSC (or MRSB) does. However, as discussed before, MRSC and MRSB have much higher success probability and smaller total movement distance than CIST and MSTB, respectively.

Figs. 11-12 show maximum movement distance of a mobile relay in the five algorithms. If a mobile relay has a larger movement distance, which costs more energy, then its energy left for communication will be less. From this perspective, it is desired to minimize the maximum movement distance of a mobile relay. It can be seen that the maximum movement distance of a mobile relay in MRSC and MRSB are smaller than those in their counterparts.

D. *Networks with Inaccessible Areas.*

Next we investigate how the performance of our MRSC algorithm is affected when the deployment area has inaccessible areas (i.e., the areas that mobile relays cannot move into), due to, for example, obstacles.

Consider that the 2000m$\times$2000m deployment area is divided into 50m$\times$50m grids. We randomly choose 20 percent of the grids as inaccessible grids. Other simulation settings are the same as those in Section VI-B. It is required that new positions of mobile relays are not allowed to be in inaccessible grids, and a mobile relay is not allowed to move to a new position if its moving path crosses an inaccessible grid. Accordingly, we make the following modifications to our MRSC algorithm to meet the requirement. For a boundary sensor pair from two blocks, the optimal positions of mobile relays that connect the boundary sensor pair are determined by Theorem 2. If any of the positions is within an inaccessible grid, then the weight of the edge of the boundary sensor pair is ∞. For a mobile relay to move to a new position, if its moving path crosses an inaccessible grid, then the movement distance is set to be ∞.

We run simulations for the modified MRSC algorithm, and the simulation statistics are collected in 1000 simulation runs. Table II shows the average change (in percentage) of performance metrics in scenarios with inaccessible areas compared with those in scenarios without inaccessible areas. Inaccessible areas certainly impact the success probability of recovery, especially when the mobile relay percentage η is low (i.e., 5%). This impact, however, can be reduced by increasing the number of mobile relays. Inaccessible areas slightly increase

TABLE II
AVERAGE CHANGE (IN PERCENTAGE) OF MRSC'S PERFORMANCE METRICS IN SCENARIOS WITH INACCESSIBLE AREAS COMPARED WITH THOSE IN SCENARIOS WITHOUT INACCESSIBLE AREAS

	Success probability	Total movement distance	# of moved mobile relays	Maximum movement distance
η =5%	-13.25%	+5.7%	+0.2%	+5.9%
η =10%	-3.2%	+10.7%	+0.8%	+10.2%
η =15%	-1.8%	+11.3%	+0.9%	+11.1%

the number of moved mobile relays. The movement distance of a moved mobile relay also tends to increase. Thus, the total movement distance as well as the maximum movement distance of a mobile relay tend to increase.

The impact of inaccessible areas on our MRSB has similar trends, and thus, the simulation results are omitted here.

A more advanced method to deal with inaccessible areas can be used if mobile relays have the ability to move around an inaccessible area. If the line segment of a boundary sensor pair (from two blocks) crosses an inaccessible area, mobile relays are allowed to be placed around the inaccessible area. Then Theorem 2 (which determines the optimal positions of mobile relays to connect a boundary sensor pair) should be modified accordingly. And the measure of movement distance of a mobile relay to a new position should be modified, as well as the method in Algorithm 1 (which approximates the movement cost to connect two boundary sensors from two blocks). The details are not discussed here due to space limit.

VII. CONCLUSION

In this paper, we study the problem of using mobile relays to reconnect a partitioned WSN with minimum movement cost. We first derive the optimal number of mobile relays needed to connect two particular boundary sensors from two blocks, as well as the optimal new positions of these mobile relays. To connect the partitioned WSN including multiple isolated blocks, the mobile relay scheduling problem is shown to be NP-complete. Thus, we focus on heuristic algorithms, and propose two greedy mobile relay scheduling algorithms. One makes a partitioned WSN connected by scheduling mobile relays to connect selected boundary sensor pairs. The other makes a rebuilt WSN still connected if subsequently one mobile relay or one selected boundary sensor fails. Extensive simulations demonstrate that our proposed algorithms can indeed achieve higher success probability and smaller movement cost than other existing methods. Future research topics may include mobile relay scheduling algorithms for dynamic WSN where nodes can fail continuously over time.

APPENDIX: PROOFS OF THEOREMS

A. Proof of Theorem 1

Without loss of generality, we consider s_i and s_j are two non-boundary sensors in blocks $\mathbb{B}_i$ and $\mathbb{B}_j$, respectively (the case when only one of s_i and s_j is non-boundary sensor can be proven similarly). By the definition of non-boundary sensor, the line from s_j to s_i must go into the communication area of another sensor in block $\mathbb{B}_i$ before the line goes into the communication area of s_i. An example is shown in Fig. 13, in which the circles are perimeters of communication areas of the sensors. Suppose the first communication area (in block $\mathbb{B}_i$) that the line from s_j to s_i goes into belongs to sensor s'_i. Then s'_i is a boundary sensor because the intersection point of the line from s_j to s_i and the circle of s'_i is not covered by other circles. Due to the same reason, we also have another boundary sensor s'_j in block $\mathbb{B}_j$ such that the first communication area (in block $\mathbb{B}_j$) that the line from s_i to s_j goes into belongs to sensor s'_j. Denote the intersection points of the line from s_i to s_j with the circles of s'_i and s'_j as x_i and x_j, respectively, as shown in Fig. 13. It follows $\|s'_i x_i\| = r$, $\|s'_j x_j\| = r$, $\|s_i x_i\| \geq r$, and $\|x_j s_j\| \geq r$. Then we have

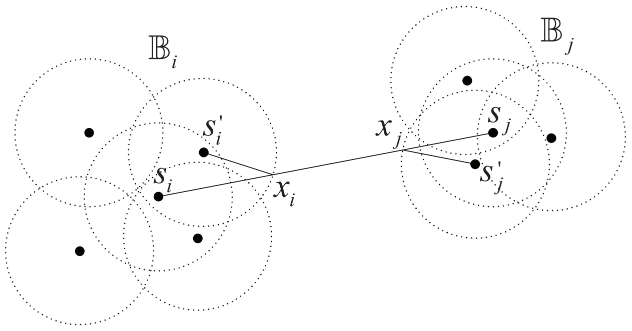


Fig. 13. The illustration for the proof of Theorem 1.

$$\begin{aligned}
\|s'_i s'_j\| &\leq \|s'_i x_i\| + \|x_i s'_j\| \\
&\leq \|s'_i x_i\| + \|x_i x_j\| + \|x_j s'_j\| \\
&= \|x_i x_j\| + 2r \\
&\leq \|s_i x_i\| + \|x_i x_j\| + \|x_j s_j\| \\
&= \|s_i s_j\|
\end{aligned}$$

where the first and the second inequality follow from the triangle inequality, and the third inequality is based on $\|s_i x_i\| \geq r$ and $\|x_j s_j\| \geq r$. Therefore, we have $L(s'_i, s'_j) \leq L(s_i, s_j)$ since $L(x, y)$ is a non-decreasing function of $\|xy\|$. This completes the proof. ■

B. Proof of Theorem 2

For the range of path loss exponent α ($2 \sim 4$), the energy consumption function $f(d)$ given in (2) is a convex function [44]. Since $f(\cdot)$ is convex, from [44], we have

$$f(a \cdot y_1 + b \cdot y_2) \leq af(y_1) + bf(y_2) \tag{3}$$

with $y_1 > 0, y_2 > 0$, $a + b = 1$, $a \geq 0$, $b \geq 0$.

For convex function $f(\cdot)$, we claim that

$$f\left(\sum_{i=1}^{L}\theta_i y_i\right) \leq \sum_{i=1}^{L}\theta_i f(y_i) \tag{4}$$

with $L \geq 2$, $y_1, y_2, \cdots, y_L > 0$, $\theta_1, \theta_2, \cdots, \theta_L \geq 0$, $\theta_1+\theta_2+\cdots+\theta_L = 1$, which can be proven by mathematical induction as follows. The result in (4) is true for $L=2$ since the case is simplified to (3). Assume the result is true for $L = m-1$ ($m \geq 3$ being an integer). Then for $L=m$, we have

$$\begin{aligned} f\left(\sum_{i=1}^{m}\theta_i y_i\right) &= f\left(\theta_m y_m + (1-\theta_m)\sum_{i=1}^{m-1}\tfrac{\theta_i}{1-\theta_m}y_i\right) \\ &\leq \theta_m f(y_m) + (1-\theta_m) f\left(\sum_{i=1}^{m-1}\tfrac{\theta_i}{1-\theta_m}y_i\right) \\ &\leq \theta_m f(y_m) + (1-\theta_m)\sum_{i=1}^{m-1}\tfrac{\theta_i}{1-\theta_m}f(y_i) \\ &= \sum_{i=1}^{m}\theta_i f(y_i) \end{aligned}$$

where the first inequality follows from (3), and the second inequality is from the induction hypothesis. Therefore, the inequality (4) holds for any $L \geq 2$.

Now we prove the first case in Theorem 2. For $(\kappa+1)r \geq \|s_i s_j\| > r$, $\kappa \geq 1$, denote the new positions of mobile relays as $(p'_1, p'_2, \cdots, p'_\kappa)$, and also denote s_i and s_j as p'_0 and $p'_{\kappa+1}$, respectively. To forward one bit information between s_i and s_j, the total energy consumption of $p'_0, p'_1, ..., p'_\kappa$ is

$$\begin{aligned} E_c &= \sum_{\ell=0}^{\kappa} c\cdot\left\|p'_\ell p'_{\ell+1}\right\|^\alpha \\ &= (\kappa+1)\sum_{\ell=0}^{\kappa}\left(\frac{1}{\kappa+1}\right)\cdot c\cdot\left\|p'_\ell p'_{\ell+1}\right\|^\alpha \\ &\geq (\kappa+1)c \\ &\quad\times\left(\frac{\|s_i p'_1\|+\|p'_1 p'_2\|+...+\left\|p'_{\kappa-1}p'_\kappa\right\|+\|p'_\kappa s_j\|}{\kappa+1}\right)^\alpha \\ &\geq (\kappa+1)\cdot c\cdot\left(\frac{\|s_i s_j\|}{\kappa+1}\right)^\alpha \end{aligned} \tag{5}$$

where the first inequality follows from (4) and the last inequality comes from $\|s_i p'_1\|+\|p'_1 p'_2\|+...+\left\|p'_{\kappa-1}p'_\kappa\right\|+\|p'_\kappa s_j\| \geq \|s_i s_j\|$. Inequality (5) means that the optimal new positions $p_1, p_2, \cdots, p_\kappa$ of mobile relays are on the line segment between s_i and s_j and satisfy $\|p_\ell p_{\ell+1}\| = \|s_i s_j\|/(\kappa+1)$, $\ell \in \{0,1,...,\kappa\}$. When mobile relays move to those optimal new positions, since $(\kappa+1)r \geq \|s_i s_j\|$, we have $\|p_\ell p_{\ell+1}\| = \|s_i s_j\|/(\kappa+1) \leq r$, $\ell \in \{0,1,...,\kappa\}$, which means that p_ℓ and $p_{\ell+1}$ can communicate directly with each other. In other words, the mobile relays form a communication path that connects s_i and s_j.

Next we prove the second case in Theorem 2. For $(\kappa-1)R+2r \geq \|s_i s_j\| > (\kappa+1)r$, also denote the new positions of the κ mobile relays as $(p'_1, p'_2, \cdots, p'_\kappa)$. To forward one bit information from s_i to s_j, the total energy consumption of $s_i, p'_1, ..., p'_\kappa$ is

$$\begin{aligned} E_c &= c\cdot\|s_i p'_1\|^\alpha + \sum_{\ell=1}^{\kappa-1} c\cdot\left\|p'_\ell p'_{\ell+1}\right\|^\alpha + c\cdot\|p'_\kappa s_j\|^\alpha \\ &\geq 2c\left(\frac{\|s_i p'_1\|+\|p'_\kappa s_j\|}{2}\right)^\alpha + \sum_{\ell=1}^{\kappa-1} c\cdot\left\|p'_\ell p'_{\ell+1}\right\|^\alpha \\ &= 2c\left(\frac{\|s_i p'_1\|+\|p'_\kappa s_j\|}{2}\right)^\alpha \\ &\quad+(\kappa-1)\sum_{\ell=1}^{\kappa-1}\frac{1}{\kappa-1}c\left\|p'_\ell p'_{\ell+1}\right\|^\alpha \\ &\geq 2c\left(\frac{\|s_i p'_1\|+\|p'_\kappa s_j\|}{2}\right)^\alpha \\ &\quad+(\kappa-1)c\left(\frac{\sum_{\ell=1}^{\kappa-1}\left\|p'_\ell p'_{\ell+1}\right\|}{\kappa-1}\right)^\alpha \end{aligned} \tag{6}$$

where the first and second inequalities come from (3) and (4), respectively. Denote $x \overset{\triangle}{=} (\|s_i p'_1\| + \|p'_\kappa s_j\|)/2$. Then from (6) and the triangle inequality, we have

$$E_c \geq 2cx^\alpha + (\kappa-1)c\left(\frac{\|s_i s_j\|-2x}{\kappa-1}\right)^\alpha.$$

Let

$$\phi(x) = 2cx^\alpha + (\kappa-1)c\left(\frac{\|s_i s_j\|-2x}{\kappa-1}\right)^\alpha.$$

The derivative of $\phi(x)$ is

$$\frac{\mathrm{d}\phi(x)}{\mathrm{d}x} = 2c\alpha\left[x^{\alpha-1} - \left(\frac{\|s_i s_j\|-2x}{\kappa-1}\right)^{\alpha-1}\right].$$

Since $\|s_i p'_1\| \leq r$ and $\|p'_\kappa s_j\| \leq r$ (because otherwise s_i and s_j cannot communicate directly with p'_1 and p'_κ, respectively), we have $x \leq r$. And together with $\|s_i s_j\| > (\kappa+1)r$, we have $(\|s_i s_j\|-2x)/(\kappa-1) > r$, which means $\mathrm{d}\phi(x)/\mathrm{d}x < 0$. Thus, $\phi(x)$ is a strictly decreasing function for $0 \leq x \leq r$, which means that $\phi(x)$ achieves its minimum value when $x = r$. Therefore, we have

$$E_c \geq 2cr^\alpha + (\kappa-1)c\left(\frac{\|s_i s_j\|-2r}{\kappa-1}\right)^\alpha. \tag{7}$$

From (6) and (7), it can be seen that the optimal new positions $p_1, p_2, \cdots, p_\kappa$ of mobile relays are on the line segment between s_i and s_j and satisfy $\|s_i p_1\| = \|p_\kappa s_j\| = r$, $\|p_\ell p_{\ell+1}\| = (\|s_i s_j\|-2r)/(\kappa-1)$, $l = 1, 2, ..., \kappa-1$. When mobile relays move to those optimal new positions, since $(\kappa-1)R+2r \geq \|s_i s_j\|$, we have $\|p_\ell p_{\ell+1}\| = (\|s_i s_j\|-2r)/(\kappa-1) \leq R$, $l = 1, 2, ..., \kappa-1$, which means that p_ℓ and $p_{\ell+1}$ (which are positions of mobile relays) can communicate directly with each other. Further, $\|s_i p_1\| = \|p_\kappa s_j\| = r$ means that s_i and p_1 can communicate directly, and p_κ and s_j can communicate directly. In other words, the mobile relays form a communication path that connects s_i and s_j.

This completes the proof. ■

C. Proof of Theorem 3

We show that the MRS problem is NP-complete by restriction [45]. The main idea of our proof is to show that a special case of the MRS problem is equivalent to a known NP-complete problem. Here, the known NP-complete problem is from [11], called constrained relay node placement (CRNP) problem. For a disconnected WSN, the CRNP problem is to place a minimum number of relay nodes to a subset of predetermined candidate positions such that the disconnected WSN becomes connected.

Next we find a special case of the MRS problem to connect a partitioned WSN. Recall that for a pair of boundary sensors from two blocks, Theorem 2 provides the new mobile relay candidate positions for connecting the two boundary sensors. Let $\mathbb{P}$ be the set of all mobile relay candidate positions of all boundary sensor pairs in the partitioned WSN. Let Z denote the number of positions in $\mathbb{P}$. For any two positions in $\mathbb{P}$, there is a distance value. Let $\delta(>0)$ be the minimal distance value of two positions in $\mathbb{P}$.

We have the following special case of the MRS problem: In the MRS problem, there are Z mobile relays. The Z mobile relays are initially located with distance $\delta/3$ to the Z positions of $\mathbb{P}$, respectively. The moving capability of each mobile relay is $\delta/3$. The MRS problem is to schedule mobile relays to some positions in $\mathbb{P}$ such that the partitioned WSN is connected with minimal total movement cost.

In this special case of the MRS problem, each mobile relay can move to only one position of $\mathbb{P}$, with movement distance being $\delta/3$. It can be seen that minimizing total movement cost in the special case of the MRS problem is equivalent to minimizing the total number of placed relay nodes in the CRNP problem. Since the CRNP problem is known to be NP-complete, the special case of the MRS problem is also NP-complete. And thus, the MRS problem is NP-complete.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Commun. Mag.*, vol. 40, no. 8, pp. 102–114, Aug. 2002.

[2] M. Younis, I. F. Senturk, K. Akkaya, S. Lee, and F. Senel, "Topology management techniques for tolerating node failures in wireless sensor networks: A survey," *Computer Networks*, vol. 58, pp. 254–283, Jan. 2014.

[3] V. Ranga, M. Dave, and A. K. Verma, "Network partitioning recovery mechanisms in WSANs: A survey," *Wireless Pers. Commun.*, vol. 72, no. 2, pp. 857–917, Sep. 2013.

[4] J. L. Bredin, E. D. Demaine, M. T. Hajiaghayi, and D. Rus, "Deploying sensor networks with guaranteed fault tolerance," *IEEE/ACM Trans. Netw.*, vol. 18, no. 1, pp. 216-228, Feb. 2010.

[5] F. Senel and M. Younis, "Optimized connectivity restoration in a partitioned wireless sensor network," in *Proc. IEEE GLOBECOM 2011*, pp. 1–5.

[6] M. Y. Sir, I. F. Senturk, E. Sisikoglu, and K. Akkaya, "An optimization-based approach for connecting partitioned mobile sensor/actuator Networks," in *Proc. IEEE INFOCOM Workshops*, pp. 525–530, 2011.

[7] S. Lee and M. Younis, "Optimized relay placement to federate segments in wireless sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 5, pp. 742–752, Jun. 2010.

[8] F. Senel, M. F. Younis, and K. Akkaya, "Bio-inspired relay node placement heuristics for repairing damaged wireless sensor networks," *IEEE Trans. Veh. Technol.*, vol. 60, no. 4, pp. 1835–1848, May 2011.

[9] X. Cheng, D.-Z. Du, L. Wang, and B. Xu, "Relay sensor placement in wireless sensor networks," *Wireless Networks*, vol. 14, no. 3, pp. 347–355, June 2008.

[10] E. L. Lloyd and G. Xue, "Relay node placement in wireless sensor networks," *IEEE Trans. Computers*, vol. 56, no. 1, pp. 134–138, Jan. 2007.

[11] S. Misra, S. D. Hong, G. Xue, and J. Tang, "Constrained relay node placement in wireless sensor networks to meet connectivity and survivability requirements," in *Proc. IEEE INFOCOM*, pp. 879–887, 2008.

[12] M. Younis and K. Akkaya, "Strategies and techniques for node placement in wireless sensor networks: A survey," *Ad Hoc Networks*, vol. 6, no. 4, pp. 621–655, June 2008.

[13] X. Li, R. Falcon, A. Nayak, and I. Stojmenovic, "Servicing wireless sensor networks by mobile robots," *IEEE Commun. Mag.*, vol. 50, no. 7, pp. 147–154, July 2012.

[14] Y.-C. Wang, "Mobile sensor networks: System hardware and dispatch software," *ACM Computing Surveys*, vol. 47, no. 1, Article 12, May 2014.

[15] DelFly, http://www.delfly.nl.

[16] Dragonfly, "Robot dragonfly-micro aerial vehicle," https://www.indiegogo.com/projects/robot-dragonfly-micro-aerial-vehicle.

[17] A. Purohit, Z. Sun, F. Mokaya, and P. Zhang, "SensorFly: Controlled-mobile sensing platform for indoor emergency response applications," in *Proc. Int. Conf. Information Processing in Sensor Networks (IPSN) 2011*, pp. 223–234.

[18] M. Di Francesco, S. K. Das, and G. Anastasi, "Data collection in wireless sensor networks with mobile elements: A survey," *ACM Transactions on Sensor Networks*, vol. 8, no. 1, Article 7, Aug. 2011.

[19] G. Wang, G. Cao, and T. F. La Porta, "Movement-assisted sensor deployment," *IEEE Trans. Mobile Comput.*, vol. 5, no. 6, pp. 640–652, June 2006.

[20] Z. Shen, Y. Chang, H. Jiang, Y. Wang, and Z. Yan, "A generic framework for optimal mobile sensor redeployment," *IEEE Trans. Veh. Technol.*, vol. 59, no. 8, pp. 4043–4057, Oct. 2010.

[21] S. Chellappan, W. Gu, X. Bai, D. Xuan, B. Ma, and K. Zhang, "Deploying wireless sensor networks under limited mobility constraints," *IEEE Trans. Mobile Comput.*, vol. 6, no. 10, pp. 1142–1157, Oct. 2007.

[22] B. Wang, H. B. Lim, and D. Ma, "A survey of movement strategies for improving network coverage in wireless sensor networks," *Computer Communications*, vol. 32, no. 13-14, pp. 1427–1436, Aug. 2009.

[23] E. D. Demaine, M. Hajiaghayi, H. Mahini, A. S. Sayedi-Roshkhar, S. Oveisgharan, and M. Zadimoghaddam, "Minimizing movement," *ACM Transactions on Algorithms*, vol. 5, no. 3, Article 30, July 2009.

[24] K. Akkaya, F. Senel, A. Thimmapuram, and S. Uludag, "Distributed recovery from network partitioning in movable sensor/actor networks via controlled mobility," *IEEE Trans. Computers*, vol. 59, no. 2, pp. 258–271, Feb. 2010.

[25] Z. Yan, Y. Chang, H. Jiang, and Z. Shen, "Fault-tolerance in wireless ad hoc networks: Bi-connectivity through movement of removable nodes," *Wireless Commun. Mobile Comput.*, vol. 13, no. 12, pp. 1095–1110, Aug. 2013.

[26] H. Liu, X. Chu, Y.-W. Leung, and R. Du, "Simple movement control algorithm for bi-connectivity in robotic sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 7, pp. 994–1005, Sep. 2010.

[27] D. Lymberopoulos and A. Savvides, "XYZ: A motion-enabled, power aware sensor node platform for distributed sensor network applications," in *Proc. Int. Symp. Information Processing in Sensor Networks (IPSN) 2005*, pp. 449–454.

[28] J. Luo, D. Wang, and Q. Zhang, "Double mobility: Coverage of the sea surface with mobile sensor networks," in *Proc. IEEE INFOCOM*, pp. 118–126, 2009.

[29] G. Lin and G. Xue, "Steiner tree problem with minimum number of Steiner points and bounded edge-length," *Information Processing Letters*, vol. 69, no. 2, pp. 53–57, Jan. 1999.

[30] S. Lee, M. Younis, and M. Lee, "Connectivity restoration in a partitioned wireless sensor network with assured fault tolerance," *Ad Hoc Networks*, vol. 24, pp. 1–19, Jan. 2015.

[31] A. A. Abbasi, M. Younis, and K. Akkaya, "Movement-assisted connectivity restoration in wireless sensor and actor networks," *IEEE Trans. Parallel Distributed Systems*, vol. 20, no. 9, pp. 1366–1379, Sep. 2009.

[32] S. Wang, X. Mao, S.-J. Tang, X. Li, J. Zhao, and G. Dai, "On movement-assisted connectivity restoration in wireless sensor and actor networks," *IEEE Trans. Parallel Distributed Systems*, vol. 22, no. 4, pp. 687–694, Apr. 2011.

[33] Y. Wang, J. Gao, and J. S. B. Mitchell, "Boundary recognition in sensor networks by topological methods," in *Proc. ACM MobiCom 2006*, pp. 122–133.

[34] J. S. Deogun, S. Das, H. S. Hamza, and S. Goddard, "An algorithm for boundary discovery in wireless sensor networks," in *Proc. High Performance Computing (HiPC) 2005*, pp. 343–352.

[35] M. Fayed and H. T. Mouftah, "Localised convex hulls to identify boundary nodes in sensor networks," *International Journal of Sensor Networks*, vol. 5, no. 2, pp. 112–125, 2009.

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

[37] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.

[38] J. R. Martinez-de Dios, K. Lferd, A. de San Bernabe, G. Nunez, A. Torres-Gonzalez, and A. Ollero, "Cooperation between UAS and wireless sensor networks for efficient data collection in large environments," *Journal of Intelligent and Robotic Systems*, vol. 70, no. 1–4, pp. 491–508, Apr. 2013.

[39] J. Allred, A. B. Hasan, S. Panichsakul, W. Pisano, P. Gray, J. Huang, R. Han, D. Lawrence, and K. Mohseni, "SensorFlock: An airborne wireless sensor network of micro-air vehicles," in *Proc. 5th Int. Conf. Embedded Networked Sensor Systems (SenSys)*, pp. 117–129, 2007.

[40] A. M. Mehta, "Mobility in Wireless Sensor Networks," Technical Report No. UCB/EECS-2012-270, Dec. 2012. (Available at: http://www.eecs.berkeley.edu/Pubs/TechRpts/2012/EECS-2012-270.pdf)

[41] C. Konstantopoulos, G. Pantziou, D. Gavalas, A. Mpitziopoulos, and B. Mamalis, "A rendezvous-based approach enabling energy-efficient sensory data collection with mobile sinks," *IEEE Trans. Parallel Distributed Systems*, vol. 23, no. 5, pp. 809–817, May 2012.

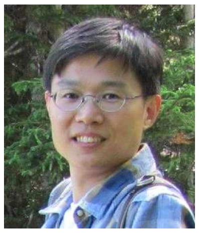
[42] Y. K. Joshi and M. Younis, "Distributed approach for reconnecting disjoint segments," in *Proc. IEEE GLOBECOM 2013*, pp. 255–260.

[43] http://www.codeproject.com/Articles/22568/Computational-Geometry-C-and-Wykobi

[44] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge,U.K.: Cambridge Univ. Press, 2004.

[45] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1979.

Zhong Shen (M'11) received the B.E. degree from the University of Shanghai for Science and Technology, Shanghai, China, in 1992, and the M.E. degree in computer science and the Ph.D. degree in information and communication engineering from Xidian University, Xi'an, China, in 2002 and 2006, respectively. He is currently an Associate Professor with the School of Telecommunications Engineering, Xidian University. His research interests include wireless sensor networks and mobile computing.



Hai Jiang (SM'15) received the B.Sc. and M.Sc. degrees in electronics engineering from Peking University, Beijing, China, in 1995 and 1998, respectively, and the Ph.D. degree in electrical engineering from the University of Waterloo, Waterloo, Ontario, Canada, in 2006. Since July 2007, he has been a faculty member with the University of Alberta, Edmonton, Alberta, Canada, where he is currently a Professor at the Department of Electrical and Computer Engineering. His research interests include radio resource management, cognitive radio networking, and cooperative communications.