Peer-to-Peer Networking and Applications (2019) 12:541–552
https://doi.org/10.1007/s12083-018-0675-8

An efficient coverage hole-healing algorithm for area-coverage improvements in mobile sensor networks

Chakchai So-In[1] · Tri Gia Nguyen[2] · Nhu Gia Nguyen[3]

Received: 1 October 2017 / Accepted: 7 August 2018 / Published online: 31 August 2018

Abstract

Maximizing network coverage is among the key factors in designing efficient sensor-deployment algorithms for wireless sensor networks (WSNs). In this study, we consider a WSN in which mobile sensor nodes (SNs) are randomly deployed over a two-dimensional region with the existence of coverage holes due to the absence of any SNs. To improve the network coverage, we thus propose a novel distributed deployment algorithm – coverage hole-healing algorithm (CHHA) – to maximize the area coverage by healing the coverage holes such that the total SN moving distance is minimized. Once the network is formed after an initial random placement of the SNs, CHHA is applied to detect coverage holes, including hole-boundary SNs, based on computational geometry, i.e., Delaunay triangulation. The distributed deployment feature of CHHA applies a concept to virtual forces that is used to decide the movement of mobile SNs to heal the coverage holes. The simulation results show that our proposed algorithm is capable of exact detection of coverage holes in addition to area-coverage improvement by healing the holes. The results also demonstrate the effectiveness of CHHA compared with other competitive approaches, namely, VFA, VEDGE, and HEAL, in terms of total moving distance.

Keywords Area coverage · Boundary detection · Coverage hole · Distributed algorithm · Sensor deployment · Wireless sensor networks

1 Introduction

Wireless sensor networks (WSNs) have a wide variety of applications for sensing and monitoring physical space, such as environmental monitoring, intrusion detection, biological detection, and battlefield surveillance [1–3]. In these applications, a large number of sensor nodes (SNs) are deployed over a geographic area to detect the existence or emergence of a certain event or specific target. Typically, every point in a region of interest (RoI) should be covered by at least one SN. However, each SN has a limited sensing range [4–6]. Hence, sensing coverage has become a critical factor to consider when deploying the SNs in WSNs [7–11].

Network coverage can be greatly affected by the geometric distribution of the SNs deployed in the network because of the constraints of SNs with limited sensing ranges. In general, a coverage hole may appear anywhere within the RoI if there exists an area in the RoI where not all points are covered by any SN in the network. Deploying a WSN with 100% sensing coverage (full coverage) can become problematic for many reasons, such as random deployment, the existence of obstacles, and energy depletion of some SNs [8–10].

A key factor to maximize the area coverage [12] is sensor placement strategies using mobile SNs. In recent years, with the advance in mobile-capable devices, some SNs, such as Robomote, are able to move automatically [13]. The emerging hardware techniques have also promoted the development of mobile sensors [14]. In addition, SNs embedded in moving devices could also help to reduce cost for deploying a WSN by minimizing the

This article is part of the Topical Collection: Special Issue on *Network Coverage*
Guest Editors: Shibo He, Dong-Hoon Shin, and Yuanchao Shu

✉ Tri Gia Nguyen
nguyengiatri@duytan.edu.vn

Chakchai So-In
chakso@kku.ac.th

Nhu Gia Nguyen
nguyengianhu@duytan.edu.vn

[1] Applied Network Technology (ANT) Laboratory, Department of Computer Science, Faculty of Science, Khon Kaen University, Khon Kaen, Thailand

[2] Faculty of Information Technology, Duy Tan University, Da Nang, Vietnam

[3] Graduate School, Duy Tan University, Da Nang, Vietnam

number of randomly deployed sensors. With the ability to move autonomously, mobile sensors within a WSN (mobile sensor networks) are able to self-deploy and self-repair [15–18].

Note that some researchers have focused on designing deployment algorithms to optimize network coverage, called movement-assisted sensor deployment [19–22]. In this specific deployment, mobile SNs are able to self-configure after an initial deployment to increase network coverage and achieve the area-coverage goal (optimally full coverage). The common goal of these studies is to solve the specific problems of how and where to move these mobile SNs efficiently so that the final deployment satisfies the coverage goal of the network. The main difference among these studies is how the new positions of the SNs are computed [15] (with the goals of minimizing the total movement to manage energy consumption and to prolong the overall network lifetime of the WSN).

In this paper, our key contribution is minimizing the movement cost of mobile SNs by optimizing a hybrid model of two methods, i.e., computational geometry-based and virtual forces-based approaches. First, we focus on the problem of hole-boundary node detection within an RoI. We construct a topology based on Delaunay triangulation. In computational geometry, a Delaunay triangulation for a given set of SNs in a $2D$ network area is a triangulation such that there is no SN inside the circumcircle of any triangles. We propose a novel algorithm, called coverage hole-healing algorithm (CHHA), based on the topology and the properties of the circumcircles of the Delaunay triangles to detect coverage holes and hole-boundary SNs. Second, after detecting holes, CHHA also includes a hole-healing component. CHHA implements a distributed deployment algorithm to determine the required movements of SNs to heal all possible coverage holes to improve area coverage while minimizing the moving distance.

The rest of this paper is organized as follows. Section 2 provides a brief survey of related work on hole-boundary-node detection algorithms, including a movement-assisted sensor-deployment algorithm in WSNs. Section 3 describes the preliminaries of our proposed algorithm – CHHA. Section 4 provides details of hole-boundary-node detection and hole-healing algorithms. Section 5 provides details on the simulation configuration, then presents and discusses the results. The performance of our proposed algorithm, evaluated with respect to various metrics, is compared with that of the virtual forces algorithm (VFA) [20], the combination of the Maxmin-edge algorithm (as an edge-based technique) and the Minimax algorithm (as a vertex-based technique) (VEDGE) [21], and hole-detection and healing algorithm (HEAL) [22]. Finally, conclusions and future research directions are discussed in Section 6.

2 Related work

In recent years, WSN research has become a popular topic; many researchers have focused on designing deployment algorithms to improve area coverage for WSNs. In this section, we briefly summarize related work concerning the two aspects of coverage hole detection and movement-assisted sensor deployment in WSNs.

2.1 Coverage hole and boundary detection

The existing approaches to detecting coverage holes and hole-boundary nodes in WSNs can be classified into a few major categories. Here, we focus on investigating the computational geometry approach.

Computational geometry approaches use the coordinates of the SNs and standard geometrical tools to detect coverage holes and hole-boundary nodes. Ghosh (2004) proposed a novel approach to estimate the coverage holes of a sensor network under random deployment using Voronoi diagrams [23]. Fang et al. (2006) provided the insight of a distributed algorithm for detecting and building routes around coverage holes based on the locations of SNs [24]. Wang et al. (2007) proposed a novel algorithm for an individual SN to identify whether it is a hole-boundary node of a coverage hole based on localized Voronoi polygons [25].

In 2011, Ma et al. [26] proposed a distributed coverage hole-detection protocol based on a computational geometry approach. Their hole-detection protocol was used for an irregular monitoring region. They considered the SN communication range to be equal to its sensing range. Some randomly selected SNs, called reference nodes, are tasked with executing a hole-detection algorithm. The reference nodes collect the information of two-hop neighbors in the neighbor discovery phase. After obtaining the neighbor list, the reference nodes detect coverage holes via computational geometry.

Li et al. (2015) [27] proposed a novel algorithm to detect coverage holes and identify hole-boundary nodes. Their algorithm consists of two phases: first, the algorithm detects coverage holes and identifies the coarse boundary node; second, the algorithm refines the hole-boundary nodes. In the first phase, they construct the Delaunay triangulation to form the network topology. They proposed an algorithm based on the circumcircle of the Delaunay triangle to detect all the coverage holes. Then, they proposed another algorithm to refine the hole-boundary nodes by removing some SNs that are not exact boundary nodes.

In 2016, Li et al. [28] proposed a novel algorithm using trees and graph theory to detect and localize coverage holes in a monitored area. In their algorithm, every SN detects whether coverage holes exist around it based on a hole-detection algorithm. Then, a method for merging holes is provided to present the global view of a coverage hole by identifying the hole location.

2.2 Movement-assisted sensor deployment

Sensor-deployment problems have been widely studied by the research community to achieve better coverage efficiency in large-scale sensor networks [20–24], [29–33]. Movement-assisted sensor deployment has attracted substantial attention in recent years. The design of a scheme to redeploy mobile SNs depends strongly on the desired performance goal. Most research into SN mobility focuses on designing algorithms to redeploy mobile SNs to enhance the area coverage. Sensor mobility is exploited to obtain a new location after the displacement of SNs to their desired locations, which directly improves the sensing coverage.

Zou et al. [20] proposed a sensor-deployment algorithm, called the VFA, to enhance the area coverage of a sensor network after the initial deployment of SNs. VFA uses a force-directed approach that combines attractive and repulsive forces to relocate SNs to maximize the area coverage of the RoI. The SNs do not move during the execution of the force algorithm. After the VFA has identified the desired locations of the SNs, the SNs are moved to their new locations.

In 2014, Mahboubi et al. [21] proposed new techniques to increase coverage in mobile sensor networks. First, they introduced the Maxmin-vertex and Maxmin-edge algorithms to maximize the minimum distance of every SN from the vertices and edges, respectively, of its Voronoi polygon. Then, they proposed the Minimax-edge algorithm to minimize the maximum distance of every SN from the edges of its Voronoi polygon. The final algorithm (VEDGE) combined the Maxmin-edge algorithm and the Minimax algorithm. Although the algorithm guarantees increased area coverage, the main characteristic of the algorithm is that the SN movement is performed iteratively, resulting in considerable energy consumption for the starting and stopping of the movement.

In the same year, Senouci et al. [22] introduced HEAL, a distributed, localized, and comprehensive two-phase protocol that can effectively estimate and enhance the area coverage in a mobile WSN. The first phase of their algorithm consists of three subtasks: hole identification, hole discovery, and border detection. The second phase consists of two subtasks: hole-healing area determination and node relocation. For coverage hole detection, they use the approach proposed in [24] to detect holes and hole borders. They identify an area with SNs that must be relocated to ensure local repair of the hole and then use virtual forces to relocate the SNs to heal the holes. The simulation results showed that this algorithm improves area coverage but is not efficient in some cases, such as dense deployment.

3 Preliminaries

In this section, we describe the network model, assumptions, and definitions that are used in our proposed algorithm.

We consider a WSN in which SNs are deployed randomly over a *2D* region such that some parts of the network have sufficient coverage due to the presence of several redundant SNs, whereas other parts have coverage holes due to absence of SNs. We assume that a WSN consists of $n$sensor nodes s_i such that $\{i|\, 1, ..n\}$. Each i^{th} SN is located at a known coordinate (x_i, y_i), and each SN has a unique identification – *id*. In addition, we assume that the system model is a multi-hop WSN with SNs having omnidirectional transceivers. Moreover, we assume that for all SNs, the communication range is larger than the sensing range [34, 35]. This assumption is realistic based on the actual sensor motes, such as MICAz, whose communication range is much larger than the sensing range of several typical sensors [35].

We assume that each SN knows its position (via low-cost GPS or other location information system) [17, 19], but we also consider possible localization and positioning inaccuracies bounded by a maximum error ε [36]. Therefore, we assume that the actual sensing radius r_s of an SN is equal to the minimum of the sensing radius $r_{\min}$ minus the maximum error ε, i.e., $r_s = r_{\min} - \varepsilon$. For any point P at (x_P, y_P), we denote the Euclidean distance between s_i and P as $d(s_i, P)$. Equation (1) shows the binary sensing model [37, 38] that expresses the coverage $c_P(s_i)$ of point P by SN s_i.

$$c_P(s_i) = \begin{cases} 1, \text{if } d(s_i, P) < r_s \\ 0, \text{otherwise} \end{cases} \tag{1}$$

Definition 1. Beside-node (BN): $s_j$is said to be a beside-node of s_i if and only if the distance between s_i and s_j is less than $2r_s$.

We assume that the communication range of an SN is a circular disk of radius r_c centered at its location. An SN can directly communicate with other SNs within the communication range.

Definition 2. Neighbor node (NN): s_j is said to be a neighbor node of s_i if and only if the Euclidean distance between s_i and s_j is less than r_c. In other words, s_i can directly communicate with s_j.

Definition 3. Coverage hole: A coverage hole is an area that is not covered by any SN.

Definition 4. Hole-boundary node: A hole-boundary node is an SN that is located adjacent to a coverage hole.

Definition 5. Hole manager (HM): An HM is a hole-boundary node that is selected to determine the movement required to heal its coverage hole (similar to the concept of a cluster head [39]).

We use a method based on Delaunay triangulation to detect hole-boundary nodes. In mathematics and computational geometry, the Delaunay triangulation for a given set P of discrete points in a plane is a triangulation $DT(P)$ such that no point in P is inside the circumcircle of any triangle in $DT(P)$.

Definition 6. Circumcircle: The notion "circumcircle" in this paper is used to express the circumcircle of a Delaunay triangle.

The following messages are used to achieve coverage of a specific area of interest with hole-boundary node detection and coverage hole healing:

- *Info-Msg* (node id, node location): Each SN broadcasts this message to its neighbor nodes.
- *BD-Msg* (node id, message id, node location, HM probability): Each SN broadcasts this message to its neighbor nodes. The HM probability is the probability that an SN will become an HM node.
- *BDFull-Msg* (node id, HM node id, list of node id and location): A hole-boundary node sends this message to another hole-boundary node who has the highest HM probability.
- *HM2-Msg* (HM node id, set of hole-boundary nodes): An HM node sends this message to another HM node.
- *HM-Msg* (HM node id, node location): An HM node broadcasts this message to corresponding neighbor nodes.
- *Join-Msg* (node id, HM node id): This message is sent by an SN to notify an HM node that it wants to become a group member.
- *Move-Msg* (HM node id, node id, new location): An HM node sends this message to its member nodes to notify the member nodes that these nodes must move to the new location.

4 Coverage hole-healing algorithm (CHHA)

The coverage hole-healing algorithm will be discussed in detail in this section. Our algorithm is a collaborative mechanism that consists of two phases, i.e., detection of hole-boundary nodes and healing of coverage holes. Once the hole-boundary nodes and the effective sensor positions with moving cost constraints are identified, a one-time movement is performed to redeploy the SNs at these positions.

4.1 Hole-boundary node detection phase

In this phase, we propose a self-organized algorithm to detect coverage holes and boundary nodes. At the beginning of this phase, each SN broadcasts an *Info-Msg* message that contains its location information to its neighbor nodes. We assume that SNs decide to broadcast and receive the location information of one- or two-hop neighbor nodes depending on the size of the network area and its communication range.

Once each SN obtains the list of its neighbor nodes, it constructs the localized Delaunay triangulation by using the method provided by Li et al. 2003 [40]. Then, each SN as a vertex of one or more Delaunay triangles calculates the center and radius of the circumcircle of the triangles. An example of SN deployment and the Delaunay triangulation is shown in Fig. 1. To exactly detect hole-boundary nodes, we provide two important theorems in the following.

Theorem 1 Consider a Delaunay triangle. If the radius of the circumcircle of the Delaunay triangle is greater than the sensing radius, there definitely exists a coverage hole in the circumcircle.

Proof If the radius of the circumcircle of a Delaunay triangle is greater than the sensing radius, the center of the circumcircle is clearly out of the sensing range of any SN. Hence, there definitely exists an uncovered region in the circumcircle, as shown in Fig. 2.

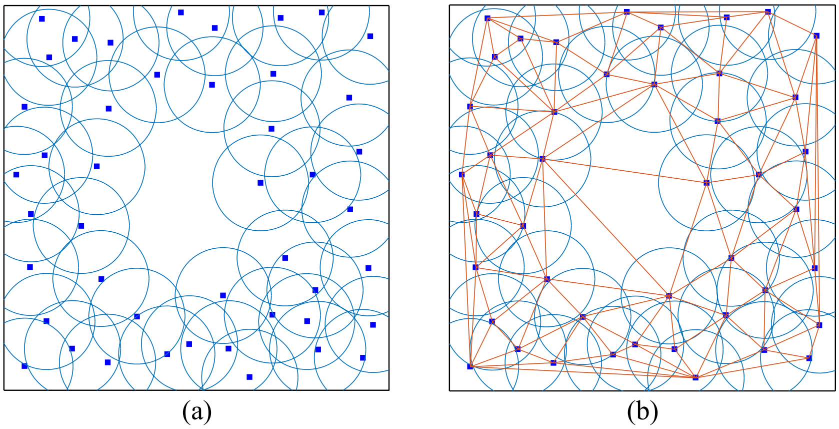

Fig. 1 **a.** An example of sensor deployment; **b.** The Delaunay triangulation

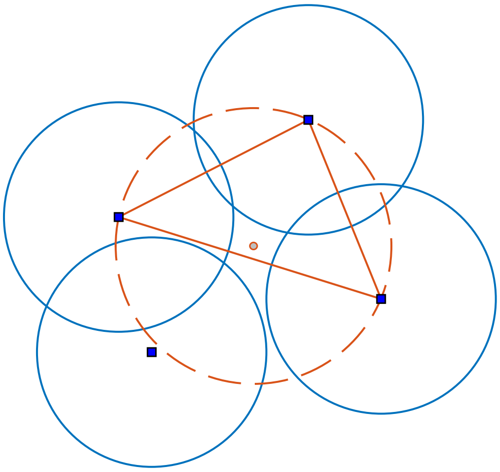

Fig. 2 Example of a coverage hole in the circumcircle: solid line = sensing range, dashed line = circumcircle

Theorem 2 Consider two-neighbor Delaunay triangles that share one or two common vertexes. If there exists an uncovered region in their circumcircles (according to Theorem 1) and the line segment between two circumcircle centers does not cross any edge of a Delaunay triangle, which is less than $2r_s$, then the two circumcircles contain the same uncovered region (in other words, the same coverage hole).

Proof If the line segment of the two circumcircle centers does not cross any edge (less than $2r_s$) of a Delaunay triangle, there definitely exists a line segment or a curve segment that connects two circumcircle centers so that all points on that segment are not covered by any SN. Therefore, the two circumcircles contain the same coverage hole. Figures 3 and 4 show examples where the two circumcircles do or do not contain the same coverage hole.

Based on the two theorems, the algorithm to detect the hole-boundary nodes is as follows:

- Consider Theorem 1. Each SN calculates the radius of the Delaunay triangle circumcircles and compares the value with its sensing radius. If a radius of a circumcircle is greater than the sensing radius, then the SN is a boundary node of a coverage hole.
- Consider Theorem 2. Each SN can determine its beside-nodes as the boundary nodes of the same coverage hole. At this time, each boundary node knows only local information about the hole. It cannot know all the boundary nodes around the hole. Thus, each hole-boundary node has to perform a process to exchange the information.
- Each hole-boundary node creates a *BD-Msg* (boundary-discovery) message that contains a message *id* (same value of its *id*), its *id*, its location information, and its probability to become a HM node and then sends *BD-Msg* to its beside-nodes that are hole-boundary nodes of the same coverage hole.

  – When a hole-boundary node receives *BD-Msg*, it inserts its *id*, its location information, and its probability to become an HM node and forwards this message to another beside-node that is the hole-boundary node of the same coverage hole. It should be noted that to minimize the number of forwarded messages, a hole-boundary node does not create its *BD-Msg* if it has previously received and forwarded *BD-Msg*. This process is completed when a *BD-Msg* has traveled around the hole. In other words, if a hole-boundary node receives two *BD-Msg* with the same message *id* from its two beside-nodes, the forwarding of *BD-Msg* is complete.
  – Then, the hole-boundary node extracts the information from the two messages, combines the information into one message (*BDFull-Msg*) and forwards the message to the hole-boundary node with the highest probability to become an HM node. Once the hole-boundary node with the highest probability to become an HM node has received the message with full information of all boundary nodes of the coverage hole, it becomes the HM node of its coverage hole. It should be noted that a hole-boundary node with a large number of neighbor nodes will have higher probability to become the HM node. Note that each hole-boundary

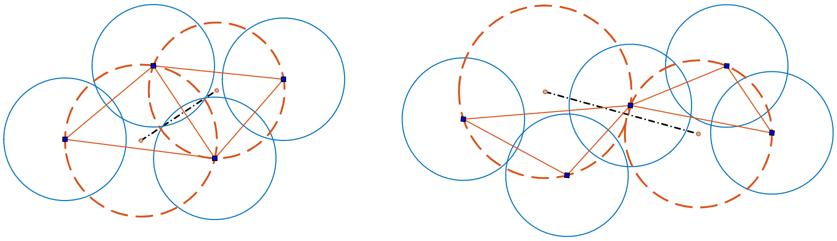

Fig. 3 Two examples of two circumcircles containing the same coverage hole

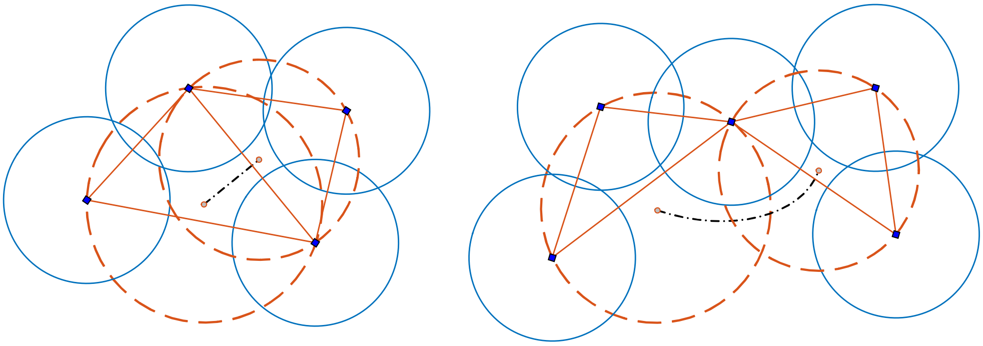

Fig. 4 Two examples of two circumcircles that do not contain the same coverage hole

node determines the probability to become the HM node based on the following formula.

$$p_i = b_i + \alpha_i \quad (2)$$

where p_i is the probability of SN s_i to become an HM node, b_i is the number of neighbor nodes of s_i, and α_i is a real random value distributed in the range of 0 to 1. α_i helps to avoid the situation in which two or more hole-boundary nodes fall into the same probability.

Once the HM of a hole is determined, the hole-boundary node detection phase is complete. Note that each hole has an *id* that is the same value as the HM node *id*. The second phase of CHHA is conducted on the HM node. Figure 5 shows an example of hole-boundary nodes and the HM node of the hole. The pseudo code shown in algorithm 1 provides the details of the hole-boundary nodes detection and the HM node selection.

Algorithm 1 Hole-boundary nodes detection and HM node selection

```
$s_i$ broadcasts Info-Msg
receives and processes Info-Msg from other nodes
constructs the localized Delaunay triangulation
if ($s_i$ is a hole-boundary node) then
    if ($s_i$ receives a BD-Msg from its beside-node $s_j$) then
        inserts information and forwards that BD-Msg to its beside-node $s_k$ that is a hole-boundary node of
        the same coverage hole with $s_j$
    else
        creates and sends a BD-Msg to its beside-nodes that are the hole-boundary nodes of the same
        coverage hole
    end if
    if ($s_i$ receives two BD-Msg with the same id from its two beside-nodes) then
        combines the information into one message (BDFull-Msg) and forwards the message to the hole-
        boundary node with the highest HM probability
    end if
    if ($s_i$ receives BDFull-Msg from a hole-boundary node) then
        becomes a HM node
        maintains a table containing the information of the hole-boundary nodes
    end if
end if
```

4.2 Hole-healing phase

After the hole-boundary nodes are detected and the HM nodes are chosen, CHHA begins the hole-healing phase. In this phase, the HM nodes determine the movement of SNs within a group to heal coverage holes. The process includes three steps.

First, the HM nodes calculate the size of the hole according to the following formula.

$$h_{HM_l} = \left| \left(\sum_{k=1}^{m-1} x_k \cdot y_{k+1} + x_k \cdot y_1 \right) - \left(\sum_{k=1}^{m-1} y_k \cdot x_{k+1} + y_k \cdot x_1 \right) \right| \div 2 \quad (3)$$

where h_{HM_l} is the size of the hole of the HM_l node, which has the set of boundary nodes $\{b_1, b_2, \ldots, b_m\}$, and (x_k, y_k) is the coordinates of hole-boundary node b_k.

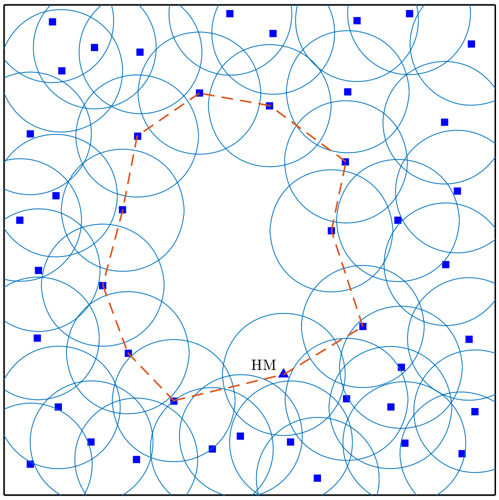

Fig. 5 The hole-boundary nodes and the HM

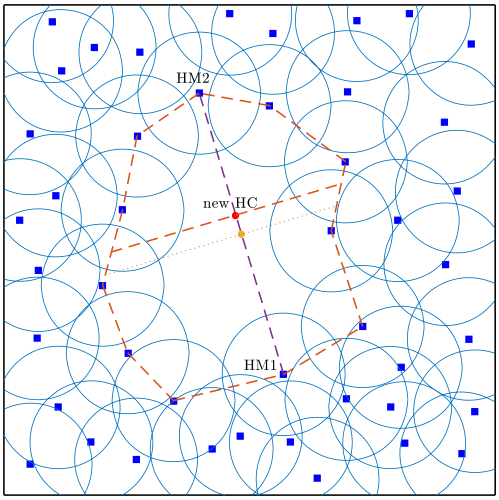

Fig. 7 The selection of the new HC

The HM node of each hole will choose another HM node of the same hole. The first HM node selects the farthest hole-boundary node from it to become the second HM node by sending a notification (*HM2-Msg*) to that node. The boundary nodes of each hole are divided into two groups and managed by the two HM nodes. Then, each HM broadcasts *HM-Msg* with its location information to its neighbor nodes. If an SN receives *HM-Msg* from an HM node, it continues to wait to determine whether it has received *HM-Msg* from the other HM nodes. Each SN maintains a table containing the information of all HM nodes from which it has received *HM-Msg* and the distance to each HM node. The information about the distance to each HM node is used by the SN to select the closest HM node. The SNs then send *Join-Msg* to their selected HM nodes. Note that an SN will become a member of an HM node after sending *Join-Msg* as an acknowledgment.

Second, the midpoint of line segment between two HM nodes of a hole is chosen as the hole center (HC). The HC point is changed depending on the number of members in the two groups managed by the two HM nodes. Each hole is divided into two sub-holes by a line segment through the HC and perpendicular to the line segment between the two HM nodes. The new HC point is determined according to the number of neighbor nodes of each HM node, which helps to minimize the movement distance for relocation of a sensor network in which the initial sensor deployment contains many coverage holes or a non-uniform node distribution. Let the force acting on the initial HC point be denoted as $\overrightarrow{F_{HC}}$. The force on the HC point is expressed as follows.

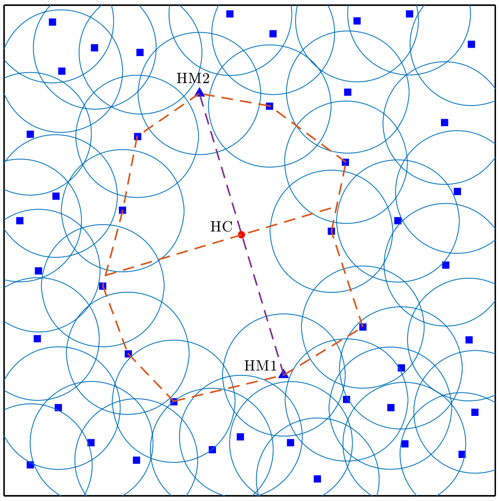

Fig. 6 The selection of HM2 and the initial HC

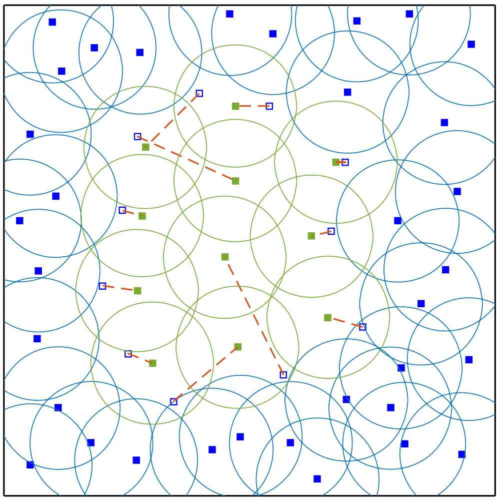

Fig. 8 The relocation to heal a coverage hole

$$\overrightarrow{F_{HC}} = \begin{cases} \left(\frac{d(m_1,\mathrm{m}_2)}{2} - u_{M_2}\cdot\frac{d(m_1,\mathrm{m}_2)}{h_l}, \beta(m_1,\mathrm{m}_2)\right), & \text{if } \left(u_{m_1} > \frac{u_l}{2}\right) \text{ and } \left(u_{m_2} < \frac{u_l}{2}\right) \\ \left(\frac{d(m_1,\mathrm{m}_2)}{2} - u_{M_1}\cdot\frac{d(m_1,\mathrm{m}_2)}{h_l}, \beta(m_1,\mathrm{m}_2)+\pi\right), & \text{if } \left(u_{m_1} < \frac{u_l}{2}\right) \text{ and } \left(u_{m_2} > \frac{u_l}{2}\right) \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

$$u_l = \frac{h_l}{2{r_s}^2} \tag{5}$$

where $d(m_1, \mathrm{m}_2)$ is the distance between two HM nodes HM_1 and HM_2; u_{m_1} and u_{m_2} are the number of members of HM_1 and HM_2, respectively; $\beta(m_1, m_2)$ is the angle of a line segment from HM_1 and HM_2; and u_l is the number of SNs required to heal the hole. Figures 6 and 7 show examples of a selecting an HC point and the division of holes into two sub-holes.

Third, we describe a virtual force calculation. Let $\overrightarrow{F_i}$ denote the total force acting on s_i. The total force $\overrightarrow{F_i}$ is a vector whose orientation is determined by the vector sum of all the forces acting on s_i. Let $\overrightarrow{F_{iS}}$ denote the total force exerted on s_i by another SN in the same group. In addition to the attractive and repulsive forces of other SNs, an SN s_i is also subjected to the forces exerted by the boundary of the two sub-holes. The total force exerted on s_i by another SN in the same group is expressed as follows.

$$\overrightarrow{F_{iS}} = \sum_{j=1, j\neq i}^{u} \overrightarrow{F_{ij}} \tag{6}$$

$$\overrightarrow{F_{ij}} = \begin{cases} \left(\delta_a\left(d\left(s_i,s_j\right) - d_s\right), \beta\left(s_i,s_j\right)\right), & \text{if } d\left(s_i,s_j\right) > d_s \\ \left(\delta_r\left(d_s - d\left(s_i,s_j\right)\right), \beta\left(s_i,s_j\right)+\pi\right), & \text{if } d\left(s_i,s_j\right) < d_s \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

where $\overrightarrow{F_{ij}}$ is the force exerted on s_i by another node s_j, $d(s_i, s_j)$ is the Euclidean distance between s_i and s_j, d_s is the threshold distance that determines how close SNs get to each other, $\beta(s_i, s_j)$ is the angle of a line segment from s_i to s_j, and δ_a and δ_r are the measure of the attractive and repulsive forces. δ_a and δ_r should be selected based on the computing capability of the SNs and the network scenario.

Because each HM node manages and decides the movement for its group, in our virtual force model, we assume that obstacle is the boundary of two sub-holes (a line segment that divides a hole into two sub-holes, see also Fig. 7). Let $\overrightarrow{F_{iB}}$ be the total force on s_i due to the boundary of two sub-holes; it is expressed as follows.

$$\overrightarrow{F_{iB}} = \begin{cases} \left(\delta_a(d(s_i,ls) - d_{ls}), \beta(s_i,ls)\right), & \text{if } d(s_i,ls) > d_{ls} \\ \left(\delta_r(d_{ls} - d(s_i,ls)), \beta(s_i,ls)+\pi\right), & \text{if } d(s_i,ls) < d_{ls} \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

where $d(s_i, ls)$ is the Euclidean distance between s_i and the line segment of the boundary of two sub-holes, d_{ls} is the threshold distance that determines how close the SNs get to the line segment ls, and $\beta(s_i, ls)$ is the angle of the line segment from s_i to the line segment ls.

The total force $\overrightarrow{F_i}$ on s_i is expressed as follows.

$$\overrightarrow{F_i} = \overrightarrow{F_{iS}} + \overrightarrow{F_{iB}} \tag{9}$$

Note that the hole-healing algorithm is executed on the HM node only. Each HM node uses the virtual forces moving to find appropriate SN locations to heal the hole. Note that an SN has to move if and only if its beside-nodes move and create a hole. Finally, when each HM node has completed the virtual forces moving and achieved the coverage requirement, a *Move-Msg* include the new locations (if any) are sent to its members. Figure 8 shows an example of the relocation to heal a coverage hole. The pseudocode shown in algorithm 2 provides the details of the hole healing phase.

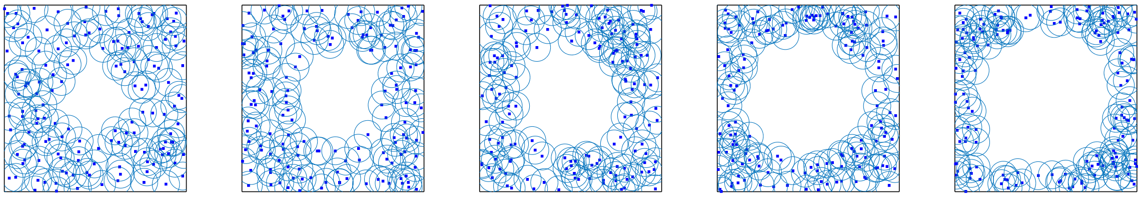

Fig. 9 The SN deployment under different hole radii

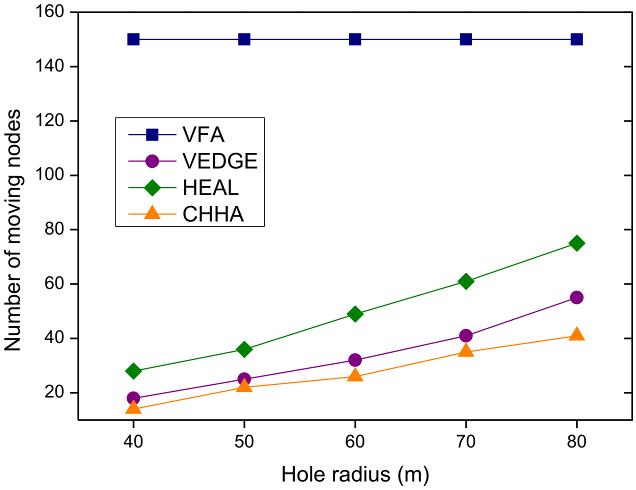

Fig. 10 Number of moving nodes under different hole radii

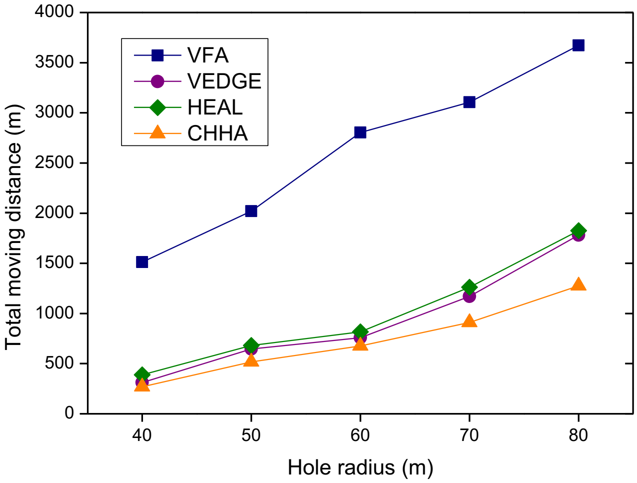

Fig. 11 Total moving distance under different hole radii

5 Performance evaluation

In this section, we perform an intensive simulation to confirm the performance of our proposed algorithm (CHHA) in terms of the moving cost (moving distance) of the SNs in a network. To strengthen our performance evaluation, CHHA is compared with three other algorithms, VFA [20], VEDGE (21), and HEAL [22].

5.1 Simulation setup

A MATLAB simulator is used to evaluate the performance of the algorithms. In these simulations, for comparison purposes, 150 SNs are randomly distributed across a plain area of 200 × 200 m^2 with the existence of a coverage hole due to the absence of SNs. We set the sensing radius r_s to 15 m and the communication radius to 80 m [35]. Note that to determine the energy consumption of the movement, we follow the energy models defined in [17, 19, 36], i.e., the energy consumed when moving 1 m is 1 unit and starting or stopping a movement costs the equivalent of 1 m movement [36].

We evaluate the performance of CHHA by comparison with three other algorithms, namely, VFA [20], VEDGE [21], and HEAL [22]. The evaluations consider three main metrics: the number of SNs that are moved, the total moving distance of all the SNs, and the total energy consumption of the movement. The presented results of all the simulations are the averages of twenty-five simulation runs in each configuration.

5.2 Simulation results and discussion

To evaluate the performance of the four algorithms, we simulate an experiment that consists of two scenarios: different hole radii and different numbers of holes. In the simulation, the four algorithms (CHHA, VFA, VEDGE, and HEAL) achieve full coverage (100%) after relocating the SNs. Therefore, we focus on evaluating only the movement metrics. In the first scenario, the hole radius is varied from 40 m to 80 m, as shown in Fig. 9.

Figure 10 compares the numbers of SNs that are required to move to a new location. The VFA algorithm applies a force-directed approach to improve coverage; the forces exerted on all SNs result in VFA moving all the SNs. VEDGE and HEAL move approximately 12 to 37% and 19 to 50% of the SNs when the hole radius is increased from 40 to 80 m, respectively. As shown in Fig. 10, CHHA moves fewer SNs than do the other algorithms. CHHA moves only approximately 9 to 27% of the SNs under different hole radii.

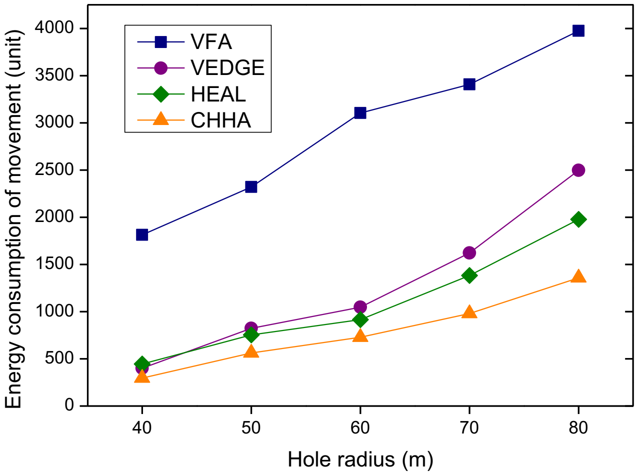

Fig. 12 Energy consumption of movement under different hole radii

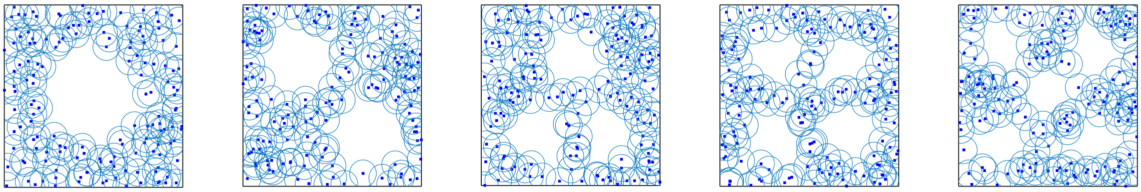

Fig. 13 The SNs deployment under different numbers of holes

Figure 11 depicts the total moving distance of the SNs for each algorithm. As the hole radius increases, the total moving distances of the four algorithms also increase. We observe a similar trend of the total moving distance for the four algorithms, in which our CHHA algorithm results in the lowest total distance of only approximately 18 to 35%, 72 to 89%, and 69 to 83% of that of VFA, VEDGE, and HEAL, respectively.

Figure 12 shows the total energy consumption. In CHHA, VFA, and HEAL, the SNs move only once, whereas VEDGE performs movement in each round. Therefore, VEDGE consumes more energy for starting and stopping movement. As the hole radius increases, the number of rounds of VEDGE also increases, resulting in greater energy consumption. The total energy consumption of CHHA is the best, i.e., approximately 16 to 34%, 54 to 74%, and 67 to 79% of that of VFA, VEDGE, and HEAL, respectively.

In the second scenario, we simulate the four algorithms with various number of holes (1 to 5 holes). Figure 13 shows the SN deployment under different numbers of holes.

Figure 14 shows the numbers of moving SNs. In this scenario, as the number of holes increases, the number of SNs moved by HEAL increases substantially. As shown in the chart, CHHA moves fewer SNs than do the other algorithms.

Figure 15 compares the total SN moving distance of each algorithm under different numbers of holes. CHHA outperforms the other algorithm, with a total moving distance of only approximately 18 to 27%, 60 to 87%, and 56 to 79% of that of VFA, VEDGE, and HEAL, respectively.

The amount of energy used during the movement is another important parameter to consider. Figure 16 compares the total energy consumption of movement for each algorithm. Again, CHHA is superior. The total energy consumption of CHHA is only approximately at 18 to 26%, 54 to 73%, and 54 to 78% of that of VFA, VEDGE, and HEAL, respectively. These results clearly illustrate the effectiveness of CHHA in relocating SNs to maximize the coverage area.

6 Conclusion

In this research, we proposed a distributed deployment algorithm to improve WSN area-coverage, namely, the coverage hole-healing algorithm (CHHA). Our contributions are two-fold. We proposed a novel algorithm to detect hole-boundary nodes based on the Delaunay triangulation topology and the property of circumcircles of the Delaunay triangle to detect coverage holes and hole-boundary-nodes. We then proposed a hole-healing algorithm to determine the movement of the SNs required to heal all possible coverage holes to improve area coverage while minimizing the moving distance.

To evaluate the performance of our proposed scheme, we compared the number of SNs moved, the total movement distance, and the total energy consumption of

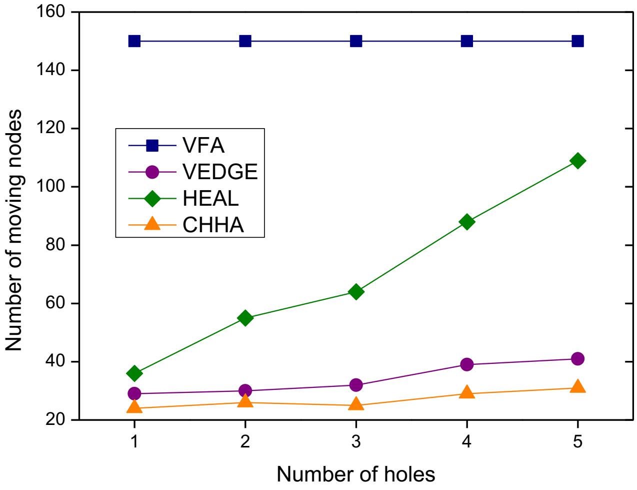

Fig. 14 Number of moving nodes under different numbers of holes

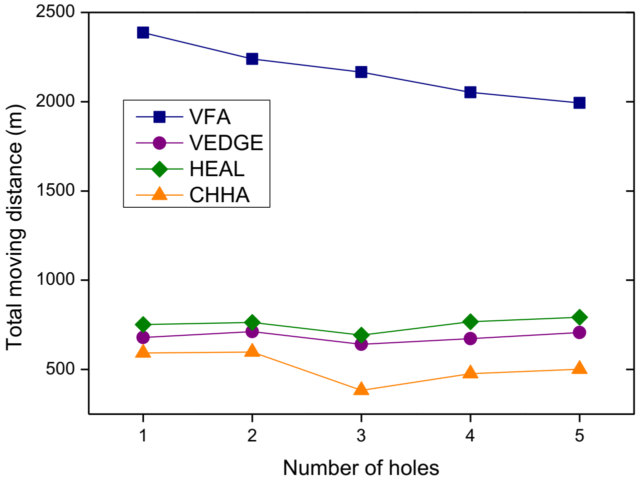

Fig. 15 Total moving distance under different numbers of holes

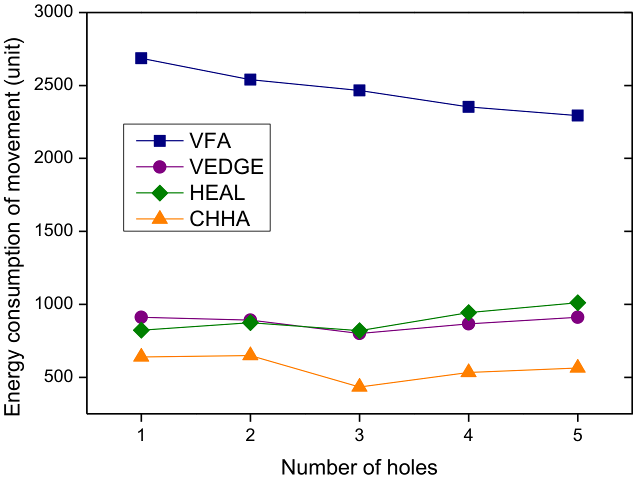

Fig. 16 Energy consumption of movement under different numbers of holes

movement with those of the VFA, VEDGE, and HEAL algorithms. Through intensive simulation experiments, the four algorithms are shown to improve the area coverage by moving the SNs to the optimal locations, thereby achieving full coverage. However, our algorithm outperforms the others on all three metrics. We can confirm that CHHA is suitable for minimizing the total movement distance while moving fewer SNs.

Furthermore, although CHHA can improve area coverage, additional modifications can produce further enhancements. Our future work will solve the area-coverage deployment problem by applying soft-computing algorithms to achieve global optimization.

Acknowledgments This work was supported by grants from Research and Academic Affairs Promotion Fund, Faculty of Science, Khon Kaen University (RAAPF), Fiscal year 2017 and Department of Computer Science, Faculty of Science, Khon Kaen University.

References

1. Akyildiz IF, Vuran MC (2010) Wireless sensor networks. John Wiley & Sons
2. Cordeiro CM, Agrawal DP (2006) Ad hoc & sensor networks: theory and applications. World Scientific Publishing
3. Dargie W, Poellabauer C (2010) Fundamentals of wireless sensor networks: theory and practice. John Wiley & Sons
4. Nguyen TG, So-In C (2016) An energy-efficient point-coverage-aware clustering protocol in wireless sensor networks. Int J Ad Hoc Ubiq Co 28(3):148–167
5. Huang CF, Tseng YC (2005) The coverage problem in a wireless sensor network. Mobile Netw Appl 10:519–528
6. Wang B (2010) Coverage control in sensor networks. Computer Communications and Networks
7. Wang Z, Cao Q, Qi H, Chen H, Wang Q (2017) Cost-effective barrier coverage formation in heterogeneous wireless sensor networks. Ad Hoc Netw 64:65–79
8. Al-Karaki JN, Gawanmeh A (2017) The optimal deployment, coverage, and connectivity problems in wireless sensor networks: revisited. IEEE Access 5:18051–18065
9. Wang Y, Wu S, Chen Z, Gao X, Chen G (2017) Coverage problem with uncertain properties in wireless sensor networks: a survey. Comput Netw 123:200–232
10. Sangwan A, Singh RP (2015) Survey on coverage problems in wireless sensor networks. Wirel Pers Commun 80(4):1475–1500
11. Wang Z, Chen H, Cao Q, Qi H, Wang Z, Wang Q (2017) Achieving location error tolerant barrier coverage for wireless sensor networks. Comput Netw 112:314–328
12. Nguyen TG, So-In C, Nguyen NG, Phoemphon S (2017) A novel energy-efficient clustering protocol with area coverage awareness for wireless sensor networks. Peer Peer Netw Appl 10(3):519–536
13. Sibley G, Rahimi M, Sukhatme G (2002) Robomote: a tiny mobile robot platform for large-scale ad-hoc sensor networks. Proc of the IEEE Int Conf Robot 2:1143–1148
14. Laibowitz M, Paradiso J (2005) Parasitic mobility for pervasive sensor networks. Proc of 3rd Int Conf on Per Co (PERVASIVE'05) 255–278
15. Senouci MR, Mellouk A (2016) Deploying wireless sensor networks: theory and practice. Elsevier Ltd
16. Abdollahzadeh S, Navimipour N (2016) Deployment strategies in the wireless sensor network: a comprehensive review. Comput Commun 91–92:1–16
17. Bartolini N, Calamoneri T, La Porta TF, Silvestri S (2011) Autonomous deployment of heterogeneous mobile sensors. IEEE T Mobile Comput 10(6):753–766
18. Nguyen TG, So-In C, Nguyen NG (2017) Barrier coverage deployment algorithms for mobile sensor networks. J Internet Technol 8(7):1689–1699
19. Wang G, Cao G, La Porta T (2006) Movement-assisted sensor deployment. IEEE T Mobile Comput 5(6):640–652
20. Zou Y, Chakrabarty K (2003) Sensor deployment and target localization based on virtual forces. IEEE INFOCOM 23rd Annual Joint Conf of the IEEE Comp and Commun Societies 1293–1303
21. Mahboubi H, Moezzi K, Aghdam AG, Sayrafian-Pour K, Marbukh V (2014) Distributed deployment algorithms for improved coverage in a network of wireless mobile sensors. IEEE T Ind Inform 10(1):163–174
22. Senouci MR, Mellouk A, Assnoune K (2014) Localized movement-assisted sensor deployment algorithm for hole detection and healing. IEEE T Parall Distr 25(5):1267–1277
23. Ghosh A (2004) Estimating coverage holes and enhancing coverage in mixed sensor networks. Proc of the IEEE Int Conf on Local Comp Netw 68–76
24. Fang Q, Gao J, Guibas LJ (2006) Locating and bypassing holes in sensor networks. Mobile Netw Appl 11(2):187–200
25. Wang G, Cao G, Berman P, Porta TFL (2007) Bidding protocols for deploying mobile sensors. IEEE T Mobile Comput 6(5):515–528
26. Ma HC, Sahoo PK, Chen YW (2011) Computational geometry based distributed coverage hole detection protocol for the wireless sensor networks. J Netw Comput Appl 34(2011):1743–1756
27. Li W, Zhang W (2015) Coverage hole and boundary nodes detection in wireless sensor networks. J Netw Comp Appl 48:35–43
28. Li W, Wu Y (2016) Tree-based coverage hole detection and healing method in wireless sensor networks. Comput Netw 103:33–43
29. Wang Y, Wu S, Gao X, Wu F, Chen G (2017) Minimizing mobile sensor movements to form a line Kcoverage. Peer Peer Netw Appl 10(4):1063–1078
30. Nguyen TG, So-In C, Nguyen NG (2018) Distributed deployment algorithm for barrier coverage in mobile sensor networks. IEEE Access 6:21042–21052

31. Yu X, Liu N, Huang W, et al. (2013) A node deployment algorithm based on Van Der Waals force in wireless sensor networks. Inter J of Distr Sensor Netw 1–8
32. Yoon Y, Kim YH (2013) An efficient genetic algorithm for maximum coverage deployment in wireless sensor networks. IEEE T Cybernetics 43(5):1473–1483
33. Mahboubi H, Moezzi K, Aghdam AG, Sayrafian-Pour K (2014) Distributed deployment algorithms for efficient coverage in a network of mobile sensors with nonidentical sensing capabilities. IEEE Trans Veh Technol 63(8):3998–4016
34. Zhang H, Hou JC (2005) Maintaining sensing coverage and connectivity in large sensor networks. Ad Hoc & Sens Wirel Ne 1(1–2):89–124
35. MICAz.: http://www.memsic.com/userfiles/files/Datasheets/WSN/micaz_datasheet-t.pdf
36. Silvestri S, Goss K (2017) MobiBar: an autonomous deployment algorithm for barrier coverage with mobile sensors. Ad Hoc Netw 54:111–129
37. Locateli M, Raber U (2002) Packing equal circles in a square: a deterministic global optimization approach. Discret Appl Math 122:139–166
38. Chakrabarty K, Iyengar SS, Qi H, Cho E (2002) Grid coverage for surveillance and target location in distributed sensor networks. IEEE T Comput 51:1448–1453
39. Heinzelman WB, Chandrakasan AP, Balakrishnan H (2002) An application-specific protocol architecture for wireless microsensor networks. IEEE T Wirel Commun 1(4):660–670
40. Li X, Calinescu G, Wan P, Wang Y (2003) Localized Delaunay triangulation with applications in wireless ad hoc networks. IEEE T Parall Distr 14(10):1035–1047

Chakchai So-In (IEEE/ACM SMs) is a Professor at the Department of Computer Science at Khon Kaen University, Thailand, received PhD in Computer Engineering from Washington University in St. Louis, USA, in 2010. He was Interns at CNAP/NTU, Cisco Systems, WiMAX Forum, and Bell Lab. His research interests include mobile computing, wireless and sensor networks, and computer networking. He has served as (guest) Editor members in SpringerPlus, PeeJ, Mobile Info. Syst., and IET Intell. Trans. Syst.; and committee members for many conferences/journals, such as Globecom, ICC, VTC, WCNC, ICNP, ICNC; IEEE Trans. (Wirel. Commun., Comput., Vehicular Technol., Mobile Comput., Industrial Info.) and IEEE Commun. Mag. and Letter. His publications are over 80s and 10 books including IEEE JSAC, Commun. Mag., and Wireless Commun. Mag.; and Android App. Dev. and Comput. Netw. and Sec. Labs.

Tri Gia Nguyen received the B.Ed. degree from the Hue University of Education, Vietnam, in 2011, the M.Sc.degree from Duy Tan University, Vietnam, in 2013, and the Ph.D. degree from Khon Kaen University,Thailand, in 2017, in computer science. He is currently a Post Doctoral Researcher with the Department of Computer Science, Faculty of Science, Khon Kaen University. His research interests include the Internet of Things, wireless communications, sensor networks, energy harvesting, wireless rechargeable sensor networks, mobile computing, parallel and distributed systems, computer systems, network security, and modeling and analysis.

Nhu Gia Nguyen received the PhD degree in Computer Science from Ha Noi University of Science-VietNam National University, Vietnam. Now, he is Dean of Graduate School- Duy Tan University, Viet Nam. He has a total academic teaching experience of 18 years with more than 50 publications in reputed international conferences, journals and online book chapter contributions (Indexed By: SCI, SCIE, SSCI, Scopus, ACM, DBLP). His area of research includes: Network Communication, Security and Vulnerability, Network Performance Analysis and Simulation, Cloud Computing, Image Processing in Biomedical. Recently, he has been the technique program committee, the technique reviews, the track chair for international conferences: FICTA 2014, ICICT 2015, INDIA 2015, IC3T 2015, INDIA 2016, FICTA 2016, IC3T 2016, IUKM 2016, INDIA 2017, under Springer-ASIC/LNAI Series.