Journal of Network and Computer Applications 82 (2017) 77–92

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Constructing event-driven partial barriers with resilience in wireless mobile sensor networks

Hyunbum Kim[a,*], Heekuck Oh[b], Paolo Bellavista[c], Jalel Ben-Othman[d]

[a] Department of Computer Science, University of North Carolina at Wilmington, Wilmington, NC 28403, USA
[b] Department of Computer Science and Engineering, Hanyang University, Ansan, South Korea
[c] Dipartimento di Informatica Scienza e Ingegneria (DISI), University of Bologna, Viale del Risorgimento 2, Bologna 40136, Italy
[d] Department of Computer Science, University of Paris 13, Villetaneuse, France

ARTICLE INFO

Keywords:
Sensor
Mobile
Event-driven
Barrier
Resilient

ABSTRACT

A barrier-coverage in wireless mobile sensor networks (WMSN) has attracted lots of interests recently. It is highly desirable to consider a barrier-coverage that can detect any moving objects between multiple sides in an event-driven environment. In this paper, we introduce a new architecture of barrier, *event-driven partial barrier*, which is able to monitor any movements of objects in the event-driven environment. Also, *resilient event-driven partial barrier* is introduced to consider the case that the constructed barriers collapsed due to failures of some sensors consisting of those barriers. Then, we define two different problems formally. One is to minimize the number of sensors to generate complete *event-driven partial barrier*. Another is to minimize a total movement distance of sensors such that *resilient event-driven partial barrier* is formed to recover from sensor failures without any new addition of sensors. To solve the first problem, we propose two approaches, *Greedy-Shared-Barrier* and *Greedy-Shared-Sensor*, which create the complete *event-driven partial barrier* with possible minimum number of sensors. For the second problem, the proposed schemes, *Uncovered-Sensor-Movement* and *Verified-Sensor-Movement* guarantee a recovery of defective barriers with possible minimum total movement distance of sensors. Then, we analyze their relative performances through extensive simulations with various scenarios and also provide the complexity analysis of the proposed schemes.

1. Introduction

During a recent few decades, wireless mobile sensor networks (WMSN) has been studied widely. WMSN has enormous potential for various applications such as battlefield surveillance, environmental monitoring, and machine health monitoring (Akyildiz et al., 2002; Yick et al., 2008; Akylidiz and Kasimoglu, 2004; Wang et al., 2005; Abbasi et al., 2009; Wang et al., 2011). Basically, WMSN is composed of a large number of sensors. A sensor is able to move after an initial deployment and has limited resources such as low battery, low computation, etc.

Among several challenges, a coverage is one of the most important research topics in WMSN because it provides detection of events in a region of interest (Huang and Tseng, 2003; Zhang and Hou, 2004; Cardei et al., 2005; Vu et al., 2009; Li et al., 2011). A particular coverage form, known as *barrier-coverage*, has been proposed by Kumar et al. (2005, 2010). In contrast to *full coverage* which guarantees every point in the deployment area is covered by the deployed sensors, *barrier coverage* has the goal to detect intruders or mobile objects, when they penetrate into a protected area, by a constructed barrier of sensors. It follows that a barrier is formed by a subset of sensors over a region of interest and at least one sensor in the barrier can detect a penetration from one side to another opposite side. Also, the given protected area is said to be k-barrier covered if all crossing penetrations by intruders are guaranteed to be detected by at least k distinct sensors in the barrier. The *barrier coverage* is considered as an efficient coverage type since it can reduce the number of sensors which are necessary to perform specific objectives when compared with the *full coverage*. Also, the *barrier coverage* can be used for numerous military, public, civil applications such as intrusion detection, border surveillance or patrol, and drug interdiction (Kumar et al., 2005, 2010; Saipulla et al., 2009).

It is highly desirable to consider barriers that are able to detect any moving objects among multiple hubs in an event-driven environment. For example, let us consider an application to monitor the volume of traffic by constructing barriers among multiple cities or hubs in the event-driven environment. Whenever a new event occurs (i.e. new hubs are added into the network) the appropriate barriers with the support

* Corresponding author.
E-mail addresses: kimh@uncw.edu (H. Kim), hkoh@hanyang.ac.kr (H. Oh), paolo.bellavista@unibo.it (P. Bellavista), jalel.ben-othman@univ-paris13.fr (J. Ben-Othman).

http://dx.doi.org/10.1016/j.jnca.2017.01.010
Received 15 March 2016; Received in revised form 17 November 2016; Accepted 15 January 2017
Available online 18 January 2017

of the new event should be constructed to guarantee detection of any movement between the updated hubs. So, in the event-driven environment, the following issues can be addressed: (i) A set of multiple hubs can be updated frequently. (ii) The formation of barriers can be changed whenever a new event occurs. It follows that it is also highly appropriate to consider a barrier-coverage which is able to detect any moving objects among multiple sides or hubs simultaneously in the event-driven environment.

Obviously, it is essential to minimize energy consumption of sensors in order to maximize a network lifetime when we construct appropriate barriers. To do so, one strategy is to adopt sleep–wakeup schedule (Kumar et al., 2005, 2010). If a sensor is not a part of barriers, a sensor will become sleep status to save own resource. And the sensor will wake up and will be active status when it is necessary to build barriers in the network. Also, if we minimize the number of sensors to form barriers satisfying barrier-coverage requirement (i.e. maintaining the same level of security), then the network lifetime can be extended.

Though minimizing energy consumption reduces the probability of sensor failures by energy depletion, a fault management of barriers should be supported automatically. It follows that we should consider the situation that sensors are failed due to energy depletion and the current barriers collapsed by loss of those sensors. Simply, we can implement all processes again. But, such a re-implementation must be a big burden for current network resource. And the additional deployment of sensors will cause an increase of cost. So, it is reasonable to use mobility of sensors to remedy those failures and we finally maintain complete barriers. Because movement of sensors spends significantly more energy than other operations (Abbasi et al., 2009; Wang et al., 2011; Akkaya and Senel, 2009; Kim and Cobb, 2015), the selection of moving sensors should be accomplished strategically as well as the movement distance of sensors should be minimized.

Based on the above motivations, we introduce a new application of barriers in an event-driven environment and also consider resilient event-driven barriers. Then, our contributions of the paper can be summarized as follows:

- We introduce a new type of barriers, *event-driven partial barriers*, such that any moving objects on paths between hubs can be detected in an event-driven environment of WMSN.
- Based on the new barrier-coverage concept, we present *k-event-driven partial barriers*, referred as *k-EP* barriers, in which any movement is guaranteed to be detected by at least *k* different sensors among hubs.
- Also, we consider *resilient event-driven barriers*, which can recover from failed sensors and can maintain *k-EP* barriers continuously.
- We formally define two different problems:
  1. A problem whose objective is to minimize the number of sensors to form *k-EP* barriers in the event-driven environment.
  2. A problem whose objective is to minimize a total moving distance of mobile sensors to recover from failures of sensors and to maintain *k-EP* barriers continuously.
- To solve the first problem, we develop two approaches, *Greedy-Shared-Barrier* and *Greedy-Shared-Sensor*, which generate the complete *event-driven partial barriers* with possible minimum number of sensors. For the second problem, we propose two schemes, *Uncovered-Sensor-Movement* and *Verified-Sensor-Movement*, which guarantee a recovery of defective barriers with possible minimum total movement distance of sensors.
- Different from our previous work (Kim et al., 2016; Kim and Ben-Othman, 2016), we first propose new schemes, *Uncovered-Sensor-Movement* and *Verified-Sensor-Movement*, to support *resilient event-driven barriers* in the paper. With a more formal algorithm description, we also update *Greedy-Shared-Barrier* and *Greedy-Shared-Sensor* schemes including initial setup to construct *event-driven partial barriers*. As another differentiation, we implement all approaches and evaluate their relative performances through extensive experiments by various scenarios. Then, we show *Greedy-Shared-Sensor* and *Verified-Sensor-Movement* outperform other schemes. Moreover, we provide the complexity analysis of those proposed schemes.

This paper is organized as follows. Next, we review related works for barrier-coverage. In Section 3, we introduce a concept of event-driven partial barriers and define a new problem with the represented notations. Then, in Section 4, we describe the proposed network system with initial setup and also propose two novel approaches to solve the defined problem. In Section 5, resilient event-driven partial barrier is introduced for fault management and also define another problem of the fault management. Then, we evaluate the performances of the developed approaches in Section 6. Finally, we conclude this paper in Section 7.

2. Related work

For coverage area, various approaches have been done with different environments. In visual sensor networks, Md et al. (2016) focused on the *k*-coverage problem which decides an orientation of minimal directional sensors such that each target is monitored at least *k* times by those directional sensors. For 3D space of wireless sensor networks Xiang et al. (2016) considered detection issues by probability sensing model based on the spatial correlation and signal detection. The authors proposed 3D space detection and coverage growing algorithm to perform the seamless 3D space coverage with the minimum number of sensors. For complete and partial coverage, Carrabs et al. (2015) addressed the maximum lifetime problem on wireless sensor networks. For problem, authors considered two variants: all sensors should be covered or some part of sensors are able to be neglected sometimes to increase network lifetime.

Initially, barrier-coverage was defined by Gage (1992). The authors considered the network application using robotic sensors. Then, the notion of *k*-barriers was introduced by Kumar et al. (2005). The *k*-barriers by Kumar et al. provide that at least *k* different sensors are able to detect any intruder's penetration which moves from one side to the other side in the given area. Also, Liu et al. (2008) deliberated on the critical conditions of strong barrier-coverage in a strip area and proposed an efficient distributed algorithm to generate barriers in a long strip area. And Li et al. (2011) researched the weak-*k*-barrier-coverage problem and showed a lower bound of weak *k*-barrier-coverage. Moreover, Saipulla et al. (2008) considered a barrier-coverage of harsh environment. The authors derived a lower bound of barrier-coverage which considers line-based deployments of sensors. And A. Chen et al. (2015) considered one-way barrier coverage concept because only one direction of crossing could be illegal for some intrusion detection applications. Hence, one-way barrier will detect the penetration if and only if an attacker is crossing with pre-specified direction.

Also, Kumar et al. proved that finding optimal schedule of sensor barriers is NP-hard and then developed an optimal sleep–wakeup scheduling algorithm called Stint to construct *k*-barriers of wireless sensors with maximizing the network lifetime in Kumar et al. (2010). Then, Ban et al. (2011) proposed a distributed algorithm of *k*-barriers of sensors, which provides a low communication overhead and a low computation cost. Also, Kim and Cobb (2013) introduced a new type of barrier-coverage, reinforced barrier-coverage, which can detect any penetration variation of the intruders. Furthermore, Chen et al. (2005) developed another barrier-coverage type, local barrier-coverage, which is able to detect intruder whose penetration is limited a slice of the belt region. The authors also designed a sleep–wakeup algorithm to maximize the lifetime of the proposed local barrier-coverage. And He et al. (2014) showed a sub-optimality of line-based deployment and introduced a distance-continuous curve. Li et al. (2012) and Chen et al. (2013) allowed the environment that intruders may have different

moving speeds. J. Chen et al. (2015) studied how to provide a perimeter barrier coverage with bistatic radar sensors. The authors considered two problems including the minimum cost bistatic transmitter–receiver placement problem and the mobile radar movement problem of perimeter barrier coverage.

On the other hand, due to recent advanced mobile technology, it is possible to apply mobile sensors to improve a specific coverage. Li and Zhang (2015) considered the problem of coverage hole detection and boundary node discovery for full coverage. They proposed a novel localized geometrical algorithm using the properties of empty circles based on Delaunay Triangulation in order to identify the existence of coverage holes and the exact nodes on the boundaries of coverage holes. Given a set of static sensors and a small of mobile nodes, Vecchio and Lopez-Valcarce (2015) introduced a distributed path-planning and coordination scheme of mobile nodes to improve the sensing area coverage. The proposed scheme iteratively computes the trajectories of mobile nodes with a greedy approach and the static sensors take roles as assistants.

In particular, there are several studies about barrier-coverage using mobile sensors. Bhattacharya et al. (2009) defined the problem of locating mobile sensors to the perimeter of polygon area in order to detect intruders. And, He et al. (2012) developed a periodic monitoring scheduling algorithm which allows each point along the barrier line to be monitored by using mobile sensors. Also, Saipulla et al. (2010) proposed a scheme which constructs the maximum number of sensor barriers with limit of movements of sensors. Kong et al. (2010) considered a mobile barrier-coverage of dynamic objects and they designed a distributed algorithm to maintain barrier-coverage. Moreover, Dobrev et al. (2015) studied problems of line-based barriers using mobile sensors and they considered the problems on minimizing the maximum movement distance of the sensor and on minimizing the total movement distance of sensors. Shen et al. (2015) studied construction of barriers with location errors when used GPS. They studied how to schedule mobile sensors to form a barrier when GPS location errors of sensors occurred. In Kloder and Hutchinson (2007), they focused on how to find a minimum segment barrier in a two-dimensional polygonally bounded region. Also, they formally defined a problem of barrier coverage by avoiding undetected intrusion in a particular area using robot sensors. They solved the problem to generate the barrier with the minimum length in the case of variable bounded range line-of-sight sensors in a two-dimensional polygonally bounded area.

3. A new architecture of barriers: event-driven partial barriers

In this section, we introduce a new application of sensor barrier. Also, we define important barrier concept of the new architecture and also define our problem formally. Then, in order to solve the defined problem, two different approaches are proposed.

3.1. A new type of sensor barrier

Recent network environments tend to occur as new events frequently such as frequent network topology change, location change of source or destination, etc. So, it is highly appropriate to consider such an event-driven environment of WMSN. It follows that our barriers of event-driven environment can support to detect or monitor any events in WMSN.

Fig. 1 describes previous k-barriers by Kumar et al. (2010) and our barrier of event-driven application. Fig. 1(a) shows basic k-barriers by Kumar et al. (2010), which guarantees that at least k sensor (i.e. k=1) can detect any penetration of intruders or mobile objects. Two barriers were found in Fig. 1(a): one is under active mode and another is under sleep mode by sleep–wakeup scheduling by Kumar et al. (2010). Active barrier is represented by dotted circles and sleep barrier is depicted by solid circles. When an intruder tries to penetrate the area from top side or from bottom side, current active barrier detects the penetration of the movement of the intruder. Then, the detecting sensor can report the information to a user.

On the other hand, Fig. 1(b) and (c) depicts an application of our proposed barrier in the event-driven environment. Whenever any events occur in the system (i.e. adding new hubs), a system is able to support those events. Suppose we consider the application to monitor the volume of traffic or to check deliveries by mobile drones among several cities or locations. Then, let us assume that all sensors are randomly deployed in the given area, which are represented as dots in Fig. 1(b) and (c). Also, each city is considered as a hub. Two independent paths are found between Hub 1 and Hub 2, and between Hub 3 and Hub 4, respectively. Each path is composed of a set of sensors. Now, suppose that an objective of the application is to check movements of vehicles between Hub 1 and Hub 2 as well as to monitor the traffic volume of vehicles between Hub 3 and Hub 4. To perform the objective of the application, it is necessary to construct barriers to detect all movements of vehicles. As Fig. 1(b), we have found two distinct paths between Hub 1 and Hub 2 and also two independent paths are found between Hub 3 and Hub 4. Note that those paths are not pre-determined paths and they are found after implementing *Initial-Setup* (which has been described as Algorithm 1 in Section 4.1.2). As Fig. 1(b), Barrier 1 with two sensors can be constructed for monitoring traffics between Hub 1 and Hub 2. Barrier 2 with two sensors is formed for detections of movements between Hub 3 and Hub 4. Because one sensor is commonly used for both barriers, three sensors are operated by Barriers 1 and 2 totally. By those barriers represented by dotted circles in Fig. 1(b), we can guarantee that any movements on paths among hubs are detected by sensors of the barriers.

Similarly, let us see another case that three independent paths are found between Hub 1 and Hub 2 and one path is found between Hub 3 and Hub 4 as Fig. 1(c). As it can be seen in Fig. 1(c), Barrier 1 (represented by dotted circles) is created in order to detect all movements of vehicles between Hub 1 and Hub 2. Also, we have Barrier 2 with one sensor that is responsible for detecting movements between Hub 3 and Hub 4. Note also that one sensor is commonly used for both barriers. By those barriers, we can also guarantee that any movement on paths among hubs are monitored by the constructed event-driven partial barriers.

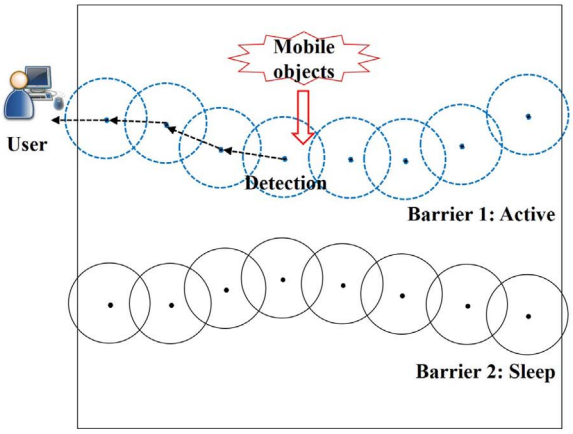
Now, we formally present important definitions of the new architecture.

Definition 1 (e-*barriers*). Given a set of wireless mobile sensors S and a set of hubs H deployed over an square-shaped area A, we found a set of paths connected by sensors between each hub pair. e-barrier is a subset of sensors on paths between two hubs. At least one sensor of e-barrier can detect any movements of mobile objects on paths. Between two hubs, multiple e-barriers can be constructed such that each e-barrier is independent and then it has no sharing sensors with other e-barriers between the two hubs.

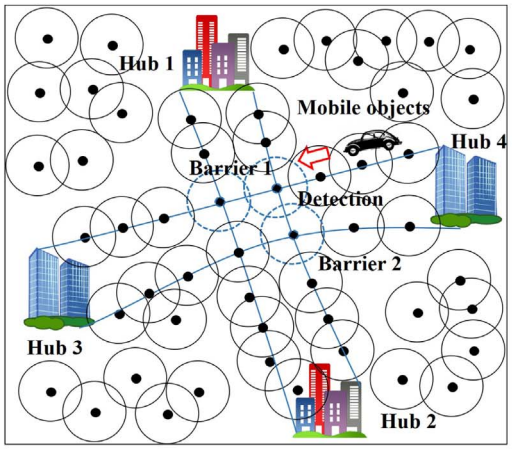
Definition 2 (*Event-driven partial barriers*). It is given that a set of wireless mobile sensors S and a set of hubs H deployed over a square-shaped area A. Given also that we found a set of e-barriers for each hub pair in the network. An event-driven partial barrier is a subset of e-barriers which guarantee that any mobile objects are detected. Also, k-event-driven partial barriers, referred as k-*EP* barriers, guarantee that any mobile objects on paths are detected by at least k sensors of k-*EP* barriers for every hub pair.

3.2. Notations

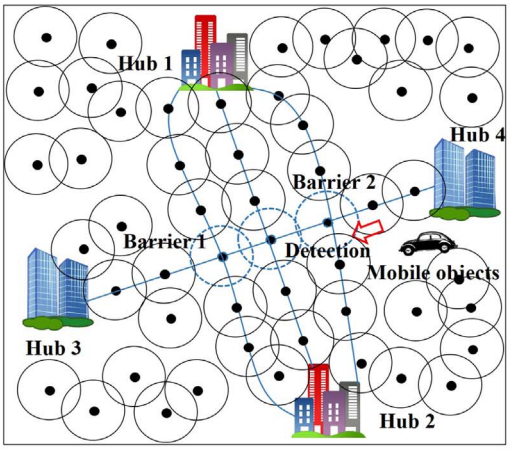
We represent notations that are used in the paper. The notations and their descriptions are summarized in Table 1.



(a) Basic k-barrier with $k = 1$



(b) The first example of event-driven partial barrier



(c) The second example of event-driven partial barrier

Fig. 1. Examples of previous k-barrier and our event-driven partial barrier.

3.3. Problem definition

To maximize the network lifetime, it is highly appropriate to reduce the number of sensors that are necessary to perform a specific goal. Therefore, in our system, it is also highly desirable to minimize the number of sensors to create *k-EP* barriers. Then, we formally define our problem as follows.

Definition 3 (MinSkEP). Given a set of wireless mobile sensors S and a set of hubs H deployed over a square-shaped area A, the minimum number of sensors for k-EP barriers (*MinSkEP*) problem is to find the set of *k-EP* barriers with smallest number of sensors from e-barriers such that the *k-EP* barriers can detect any movements of objects among given hubs.

In addition, the term *partial* indicates *reducible* with dependency in our event-driven partial barrier. It is possible for some sensor node to be a common member among e-barriers for different hub pairs. Such a property shows a dependency in the proposed event-driven partial barrier. It follows that, to construct k-event-driven partial barriers, we choose the set of e-barriers with possibly reducible number of sensors by using the dependency property among e-barriers.

For example, let us check Fig. 2. Suppose that we consider four different hubs: h_1, h_2, h_3, h_4. And assume that a small triangle represents a hub and a small dot means a sensor node. Also, dotted lines depict paths between hubs in Fig. 2. In Fig. 2(a), suppose we have found a set of e-barriers, $E_{1,2} = \{e_1, e_2, e_3, e_4, e_5, e_6\}$ between h_1 and h_2. So, by Definition 1, any mobile objects on paths can be detected by at least one sensor of e-barrier. Similarly, in Fig. 2(b), assume that we have found a set of e-barriers, $E_{3,4} = \{e_7, e_8, e_9, e_{10}, e_{11}, e_{12}\}$ between h_3 and h_4. Now, suppose we add hubs h_3 and h_4 into h_1 and h_2 in the event-driven environment and we set up k as 3. Then, if we separately consider those e-barriers without any consideration of reducibility with dependency, it is possible that we simply generate the below barrier formation provided that the number of sensors in the barriers is 18 as it can be seen in Fig. 2(c). By the formed barriers including e_1, e_3, e_5, e_7, e_9, e_{10}, we may provide the property that any mobile objects among hubs are detected by at least k sensors. But, our k-event-driven partial barriers pursues the reducibility based on dependency. As you can see in Fig. 2(d), some sensor nodes are used as common members at the set of e-barriers $E_{1,2}$ and $E_{3,4}$. By using the dependency, we can create k-event-driven partial barriers provided that the number of sensors in the barriers is 15 as Fig. 2(d). Compared with the formation of barriers in Fig. 2(c), k-event-driven partial barriers in Fig. 2(d) show a view of partially constructed e-barriers as well as the reduced number of sensors in the k-event-driven partial barriers.

Furthermore, Fig. 3 is an example of construction of *k-EP* barrier and our *MinSkEP* problem. In Fig. 3, assume that a set of sensor nodes S is deployed randomly in a two dimensional square-shaped area. And

Table 1
Notations.

Notations	Description
A	A square-shaped interested area
S	A set of sensors
n	The number of sensors
r	A communication range of sensor
H	A set of hubs
m	The number of hubs
k	A detection level of *k-EP* barriers
$H_{i,j}$	A hub pair of hub h_i and h_j where $i \neq j$
$P_{H_{i,j}}$	A set of node-disjoint paths between hub h_i and h_j
$E_{i,j}$	A set of *e*-barriers of $H_{i,j}$
$\mathcal{G}$	A flow graph
$V(\mathcal{G})$	A set of vertices of $\mathcal{G}$
$E(\mathcal{G})$	A set of edges of $\mathcal{G}$
F	A set of frequency of sensor
e_{max}	*e*-barrier with maximum number of shared sensors
EPS	The number of sensors consisting of *k-EP* barriers
E_{act}	A set of current activated *e*-barriers
S_{act}	A set of sensors covered by E_{act}
$\lvert S_{act} \rvert$	A size of S_{act}
s_{max}	A sensor with largest frequency in F
$S_{failures}$	A set of failure sensors in *k-EP* barriers
$S_{uncovered}$	A set of uncovered sensors by *e*-barriers
$totalmove$	A total movement distance of sensors
S_{active}	A set of covered sensors by *e*-barriers
$EucDist[a, b]$	Euclidean distance between a and b
(a, b)	a and b are neighbors at each other

we have a set of hubs $H = \{h_1, h_2, h_3, h_4, h_5, h_6\}$. Note that a sensor is depicted as a small circle and a hub is represented as a small triangle. Then, let us assume that we have three different hub pairs: $H_{1,2} = \{h_1, h_2\}$, $H_{3,4} = \{h_3, h_4\}$, $H_{5,6} = \{h_5, h_6\}$, respectively. As it can be seen in Fig. 3, a small circle represents a sensor and a hub is depicted as a triangle. Also, possible independent paths between each hub pair $H_{1,2}$, $H_{3,4}$, $H_{5,6}$ are represented as dotted lines.

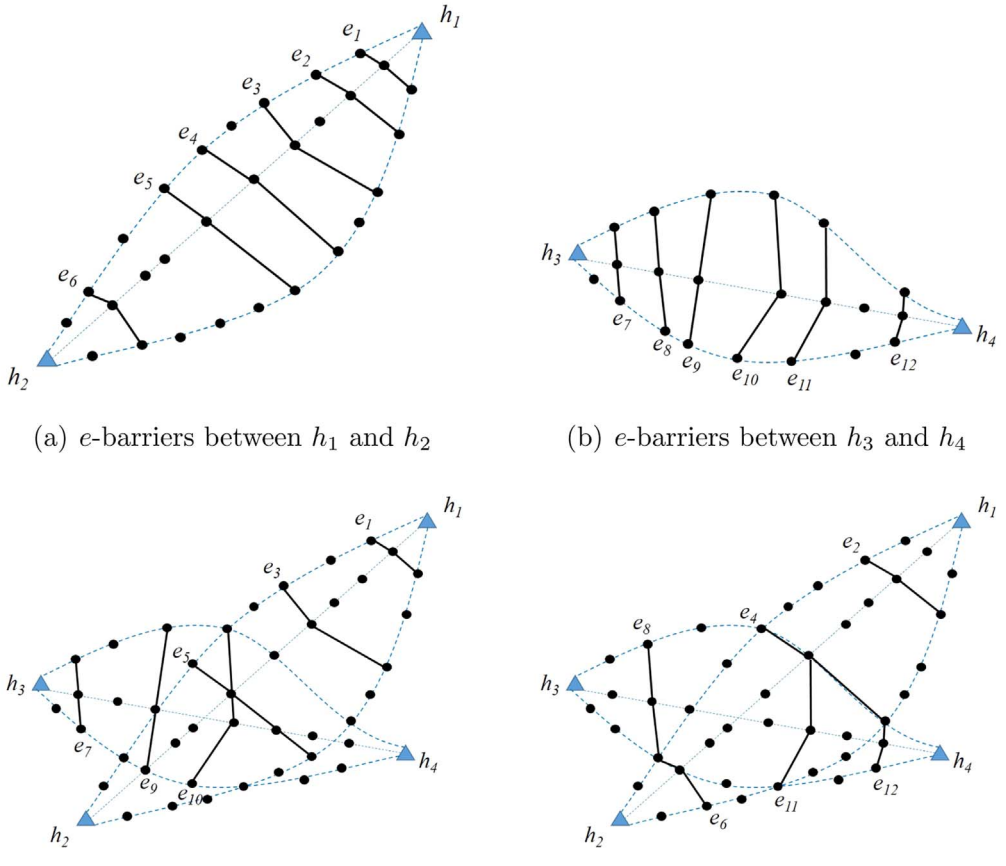
Then, we search for independent paths between each hub pair, where each path is composed of a subset of sensors. As Fig. 3(a), suppose that we found four independent paths between h_1 and h_2, three independent paths for h_3 and h_4 and also there are three paths between h_5 and h_6. Fig. 3(b) shows a construction of *k-EP* barriers with k=1. So, at least one sensor of *k-EP* barriers can detect any movements among hubs. The constructed *k-EP* barriers are represented as solid lines. Remind our *MinSkEP* problem and note that the number of sensors of current *k-EP* barriers is 9 in Fig. 3(b). Also, Fig. 3(c) and (d) shows the created *k-EP* barriers with k=2 and k=3. In Fig. 3(c), the formed *k-EP* barriers with k=2 use 18 sensors. And, in Fig. 3(d), 26 sensors are used to construct *k-EP* barriers with k=3.

4. Construction of event-driven partial barriers

4.1. An initial setup of event-driven partial barriers

4.1.1. Network system model

Now, we describe our network system architecture with the considered components and explain how to perform an initial setup



(a) e-barriers between h_1 and h_2

(b) e-barriers between h_3 and h_4

(c) Separate construction of barriers with $k = 3$ without considering a dependency for h_1, h_2, h_3, h_4

(d) Construction of k-event-driven partial barriers with $k = 3$ using the reducibility based on dependency for h_1, h_2, h_3, h_4

Fig. 2. A description of reducibility with dependency in event-driven partial barriers.

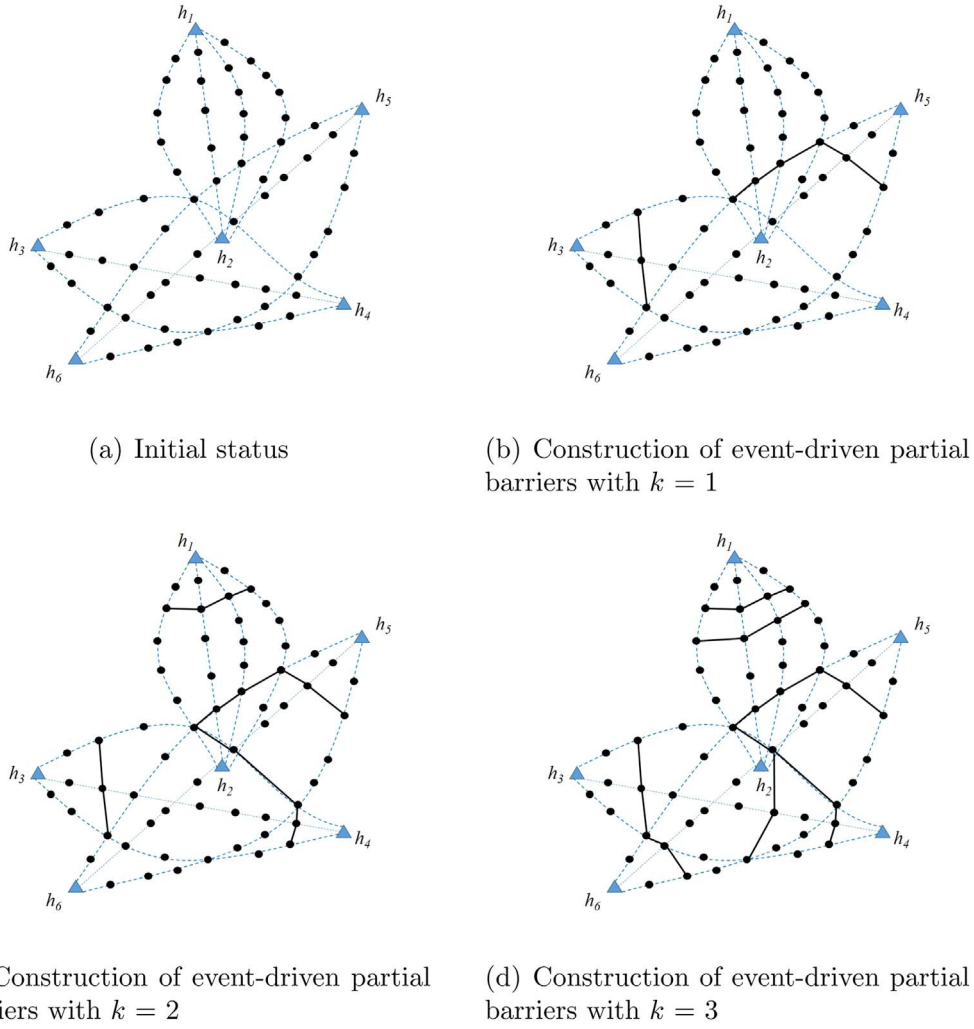


Fig. 3. Possible formation of event-driven partial barriers with different k values.

before constructing *k-EP* barriers in the architecture.

In our system, a set of mobile sensors S with size n is randomly deployed in a square area A. After the deployment, each sensor initially maintains its own location without movement to reduce its own energy. When there are failures in the network, each sensor is movable to perform specific goal. Also, a set of interested hubs H with size m is considered within the area A. The set H can be changed frequently. (i.e. adding new hubs, location changes of the hubs). To minimize energy consumption, it is possible for each sensor to have two modes: one is a sleep mode in which a sensor uses a negligible amount of battery and another is a service mode in which a sensor senses own environment. Assume that each sensor has an equal amount of resources and thus, the same lifetime. Also, we assume that each sensor has the equal communication range denoted by r.

We also assume that two sensors, s_a, s_b where $s_a, s_b \in S$, have neighbor relationship with connection if the *Euclidean distance* between them is at most $2{\cdot}r$. That is, if $EucDist\,[s_a, s_b] \leq 2{\cdot}r$ where $s_a, s_b \in S$ and $a \neq b$, then s_a and s_b are connected and it exists an edge between two sensors. A *node-disjoint path* (or an *independent path*) is a sequence of sensors, $s_1, s_2, \ldots, s_g \in S$, between two hubs h_i, $h_j \in H_{i,j}$. Then, s_1 can be a neighbor of h_i, so, we have (s_1, h_i). Also, s_g becomes a neighbor of h_j, so (s_g, h_j).

Note that it is possible to have multiple node-disjoint paths for a hub pair $H_{i,j}$. Hence, it is defined that a node-disjoint path set $P_{Hi,j} = p_1, p_2, \ldots, p_c$ between a hub pair $H_{i,j}$, where $c > 0$. By Definition 1, it is possible to have multiple *e*-barriers. So, it can be defined that a set of *e*-barriers $E_{i,j} = e_1, e_2, \ldots, e_d$ where $d > 0$. Note that also each *e*-barrier in $E_{i,j}$ is an independent one and is constructed as a sequence of connected sensors on different paths, $s_1, s_2, \ldots, s_a$, where $s_1 \in p_1$, $s_2 \in p_2, \ldots, s_a \in p_c$.

4.1.2. Initial-Setup

In order to find possible node-disjoint paths $P_{Hi,j}$ for each hub pair (i.e. from h_i to h_j), we execute an initial algorithm which is referred as *Initial-Setup*. Given S, H, r, k, *Initial-Setup* is implemented by the following three steps:

Step 1:

- A *flow graph* $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$ is created as follows:
  1. For each sensor $s \in S$, create two vertices s_{in} and s_{out}, then include them into $V(\mathcal{G})$.
  2. For each sensor $s \in S$, add a directed edge $\overrightarrow{s_{in}, s_{out}}$ and then include it into $E(\mathcal{G})$.
  3. For every pair of sensors, $s, u \in S$ where $EucDist\,[s, u] \leq 2{\cdot}r$, $s \neq u$, add the following two directed edges $\overrightarrow{s_{out}, u_{in}}$ and $\overrightarrow{u_{out}, s_{in}}$ into $E(\mathcal{G})$.
  4. Add a hub pair, h_i, $h_j \in H$ into $V(\mathcal{G})$.
  5. For the neighbor sensor s of h_i, create a directed edge $\overrightarrow{h_i, s_{in}}$ and add it into $E(\mathcal{G})$.
  6. For the neighbor sensor u of h_j, create a directed edge $\overrightarrow{h_j, u_{in}}$ and include it into $E(\mathcal{G})$.

- Assign a capacity value of 1 to every edge in the flow graph $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$.
- Suppose that h_i and h_j are source and destination, respectively. Then, execute a maximum flow algorithm, Edmonds–Karp algorithm (Edmonds and Karp, 1972), which allows us to find the maximum flows between h_i and h_j. Note that a flow is equivalent to a node-disjoint path. It follows that each node-disjoint path is a part of $P_{H_{i,j}}$. So, we get $P_{H_{i,j}} = p_1, p_2, \ldots, p_c$ where c is a maximum flow value by Edmonds–Karp algorithm (Edmonds and Karp, 1972).

Step 2:

- Using the found $P_{H_{i,j}}$, search for a set of e-barriers, $E_{i,j} = e_1, e_2, \ldots, e_d$ where $d > 0$. Each e-barrier in $E_{i,j}$ is an independent one and it is generated by a sequence of sensors on different paths. Therefore, $e_l = s_1, s_2, \ldots, s_a$ where $e_l \in E_{i,j}$ and $s_1 \in p_1$, $s_2 \in p_2, \ldots, s_{a-1} \in p_{c-1}$, $s_a \in p_c$ and also we have neighbor relationships, $(s_1, s_2), (s_2, s_3), \ldots, (s_{a-1}, s_a)$.

Step 3:

- For every hub pair $h_i, h_j \in H$ in a hub set $H = \{h_1, h_2, \ldots, h_m\}$, we perform the above steps 1 and 2. Hence, we get a different set of e-barriers for each hub pair. That is, $E_{i,j} \neq E_{k,l}$. But, a sensor can be shared by a different e-barriers. It follows that a sensor used at $E_{i,j}$ can be a part of $E_{k,l}$.
- For each $E_{i,j}$, check whether there exists $|E_{i,j}| < k$ for each $E_{i,j}$ for every hub pair $h_i, h_j \in H$.
- If there exists $|E_{i,j}|$ where $|E_{i,j}| < k$, then return *failure*.
- For each sensor, calculate its frequency that is the number of times used in e-barriers of different hub pairs. Then, we have a set of frequency value of each sensor, $F = \{f_1, f_2, \ldots, f_n\}$ and return F.

The pseudocode for above three steps is described in Algorithm 1, which we call as *Initial-Setup*. Because we consider an event-driven environment, *Initial-Setup* will be performed whenever new events occur such as additions of new hubs.

Algorithm 1. *Initial-Setup*.

Inputs: S, H, r, k, Output: F or *failure*
1: set $F = \emptyset$;
2: find a node-disjoint path set $P_{H_{i,j}}$ for each hub pair $H_{i,j}$;
3: find a e-barrier set $E_{i,j}$ for each $H_{i,j}$;
4: check if there exists $|E_{i,j}| < k$ for each $E_{i,j}$;
5: **if** there is $|E_{i,j}|$ where $|E_{i,j}| < k$ **then**
6: return *failure*
7: **end if**
8: set a frequency f_a of each sensor s_a shared by different e-barriers: $f_a = 0$;
9: set a count $a = 1$;
10: **while** $a <= n$ **do**
11: **while** true **do**
12: for sensor $s_a \in S$, check whether s_a is used as e-barrier for a different hub pair;
13: **if** there exists **then**
14: update a frequency of the sensor: $f_a = f_a + 1$;
15: **else**
16: $F \leftarrow f_a$;
17: a^{++};
18: break;
19: **end if**
20: **end while**
21: **end while**
22: return F

4.2. Proposed approaches for MinSkEP problem

In this section, we present our two different approaches to solve *MinSkEP* problem. Both approaches use results by *Initial-Setup*.

4.2.1. Greedy-Shared-Barrier approach

To construct k-*EP* barriers, our first approach, which we referred to as *Greedy-Shared-Barrier*, first chooses the e-barrier in which it has the totally maximum number of shared sensors by other e-barriers. Then, find e-barriers affected by sensors in the chosen e-barrier and activate the e-barriers. *Greedy-Shared-Barrier* finally returns *EPS* which is the number of sensors consisting of complete k-*EP* barriers. The steps of *Greedy-Shared-Barrier* are as follows:

- Choose e-barrier, e_{max}, which has the maximum number of shared sensors by other e-barriers.
- Identify e-barriers affected by sensors of the selected e_{max}.
- Activate the identified e-barriers as well as e_{max} barrier.
- Update those current activated e-barriers E_{act} as members of k-*EP* barriers.
- Verify a sensor set, S_{act}, which is covered by the current activated e-barriers E_{act} and then calculate the size of S_{act}, $|S_{act}|$.
- Update *EPS* by adding $|S_{act}|$.

The above steps are iterated until k-*EP* barriers are formed completely. The pseudocode of *Greedy-Shared-Barrier* is presented in Algorithm 2.

Algorithm 2. *Greedy-Shared-Barrier*.

Inputs: S, H, F, r, k, Output: *EPS*.
1: set *EPS*=0;
2: **while** k-*EP* barriers are not constructed **do**
3: set a set of current activated e-barriers: $E_{act} = \emptyset$;
4: set a sensor set which is covered by E_{act}: $S_{act} = \emptyset$;
5: choose e_{max} with the totally largest number of shared sensors by other e-barriers;
6: activate e_{max} as well as e-barriers affected by sensors within e_{max} and add those activated barriers to E_{act};
7: add E_{act} to k-*EP* barriers;
8: identify S_{act} of E_{act};
9: calculate $|S_{act}|$ which is the number of sensors of S_{act};
10: update $EPS = EPS + |S_{act}|$;
11: **if** k-*EP* barriers are constructed **then**
12: break;
13: **end if**
14: **end while**
15: return *EPS*

4.2.2. Greedy-Shared-Sensor approach

When we form k-*EP* barriers completely, our second approach, *Greedy-Shared-Sensor*, search from a sensor s_{max} with the largest frequency and activate e-barriers including the found s_{max}. Then, those activated e-barriers become a part of k-*EP* barriers. *Greedy-Shared-Sensor* also finally returns *EPS* after forming complete k-*EP* barriers. *Greedy-Shared-Sensor* does the following steps iteratively:

- Search for a sensor s_{max} which is a sensor with largest frequency in the set of frequency value F.
- Identify e-barriers covering s_{max}.
- Activate the identified e-barriers and consider them as current activated e-barriers E_{act}.
- Update E_{act} as members of k-*EP* barriers.
- Find a sensor set S_{act} which is contained in E_{act} and then calculate the size of S_{act}, $|S_{act}|$.

- Update *EPS* by adding $|S_{act}|$.

Algorithm 3. *Greedy-Shared-Sensor.*

Inputs: S, H, F, r, k, Output: *EPS*.
1: set *EPS*=0;
2: **while** *k-EP* barriers are not constructed **do**
3: set a set of current activated *e*-barriers: E_{act}=∅;
4: set a sensor set which is covered by E_{act}: S_{act}=∅;
5: choose a sensor s_{max} with the largest frequency value f_{max} in F;
6: activate *e*-barriers affected by s_{max} and add them to E_{act};
7: add E_{act} to *k-EP* barriers;
8: identify S_{act} of E_{act};
9: calculate $|S_{act}|$ which is the number of sensors of S_{act};
10: update $EPS=EPS+|S_{act}|$;
11: **if** *k-EP* barriers are constructed **then**
12: break;
13: **end if**
14: **end while**
15: return *EPS*

Until *k-EP* barriers are constructed completely, we iterate the above steps. The pseudocode is represented in Algorithm 3 in more detail.

5. Resilient event-driven partial barriers

In this section, as another critical issue of our contribution, we introduce a fault management of event-driven partial barriers using movements of mobile sensors.

5.1. Fault management of event-driven partial barriers using mobile sensors

Although minimizing energy consumption reduces the probability of sensor failures due to energy exhaustion, it is highly appropriate for the system to support a fault management of barriers. That is, even though *k-EP* barriers should be maintained continuously, several sensors in barriers can be failed because of energy depletion and then such a failure of sensor will cause a collapse of the system as well as a breakdown of *k-EP* barriers. Simply, we may think of additional deployments of sensors into the network. However, the additional deployment will cause an increase of cost. Hence, it is desirable to use movements of current sensors in the network to handle those failures and maintain *k-EP* barriers resiliently. Then, mobile sensors finally can recover current imperfect *k-EP* barriers without any additional deployment of sensors.

We present the formal definition of a *resilient event-driven partial barriers*.

Definition 4 (*Resilient event-driven partial barriers*). Given a set of mobile sensors S, a set of hubs H deployed over a square-shaped area A and *k-EP* is formed initially. Then, when failed sensors occur in the network and *k-EP* barriers collapsed due to those failures, resilient event-driven partial barriers are to remedy the failures and to maintain complete *k-EP* barriers by moving mobile sensors.

5.2. Problem definition

When sensors move to remedy incomplete *k-EP* barriers, excessive movement of sensors will accelerate collapse of the whole system because a movement operation will result in more energy consumption than other operations. Therefore, minimizing movement of sensors should be considered in resilient event-driven partial barriers. Based on this observation, our second problem is defined formally as follows:

Definition 5 (MinTMove). For resilient event-driven partial barriers, minimizing total movement of sensors (*MinTMove*) problem is to minimize a total movement distance of sensors to remedy current failures and to maintain complete *k-EP* barriers without any additional lost connections as well as any additional deployment of sensors.

A difficulty of fault management and *MinTMove* problem is a dependency property of the system with *k-EP* barriers. Simply, it is not possible to select the closest sensor from current failed sensor and to relocate the chosen sensor into the position of the failed sensor. Such a movement may provoke additional serious problems in the system. (i.e. loss of *e*-barrier, loss of another *k-EP* barrier, failure of maintenance of same number of node-disjoint paths on current hubs.) Also, we have to pursue minimizing a total movement distance of sensors with the dependency property so as to maximize the network lifetime. Hence, the selection of moving sensors and their movement strategies should be considered carefully to solve *MinTMove* problem.

Now, we show possible cases of moving sensor selection to remedy the failure. Fig. 4 describes an occurrence of sensor failure and examples of moving sensor selection to remedy the failure. In Fig. 4, a triangle depicts a hub and a small circle represents a sensor. Also, possible node-disjoint paths between hub are drawn as dotted lines and barriers are described as solid lines and a failed sensor is depicted as a small red rectangle. Then, suppose that we have two hub pairs, $H_{1,2}=\{h_1, h_2\}$ and $H_{3,4}=\{h_3, h_4\}$. Assume that *k-EP* barriers with k=3 were formed in the network initially as Fig. 4(a). Suppose sensor s_f is failed due to energy depletion. Then, such a failure of s_f results in collapse of a barrier with sensors s_a, s_f, s_b, s_e as well as loss of both edges $\overline{s_a, s_f}$ and $\overline{s_f, s_b}$ as Fig. 4(a). To recover the failure, the closest sensor s_d of failure sensor s_f is chosen and is moved to the location of s_f in Fig. 4(a). Though the movement may remedy $\overline{s_a, s_f}$ and $\overline{s_f, s_b}$, it may cause an additional lost connection of sensor s_l if the moved location is out of transmission range of sensor s_l. For the worse, the lost connection also may cause to lose one entire node-disjoint path between h_3 and h_4. Then, as Fig. 4(c), we may try to select the next closest sensor s_b of s_f. If s_b moves to the location of s_f to recover the failure, such a movement will cause additional lost connections of $\overline{s_b, s_g}$, $\overline{s_b, s_e}$. It follows that those lost connections will destroy current normal barriers as depicted in Fig. 4(c). Moreover, Fig. 4(d) shows the selection of the next possible closest sensor, s_a, which is on another node-disjoint path between h_3 and h_4. If s_a is replaced with s_f, additional lost connections of $\overline{s_a, s_i}$ and $\overline{s_a, s_c}$ will be caused and the network has a collapse possibility of node-disjoint path between h_3 and h_4 if s_i is out of communication range with s_c. In Fig. 4(e), we may choose the next possible closest sensor s_j which is positioned on the same node-disjoint path with s_f. Similar with case 1 of 4(b), if s_j moves to the location of s_f, the movement will cause to lose a connection between s_j and s_k if the location of s_f is out of communication range with s_k. Then, in Fig. 4(f), we describe a case of proper movement by selecting s_i. The movement of s_i towards s_f does not have both any additional lost connection and collapse of barriers. Thus, after s_i is replaced with s_f, we can recover the failure finally and can maintain complete *k-EP* barriers continuously.

5.3. Proposed approaches for MinTMove problem

In this section, we present our approaches to solve *MinTMove* problem. Before implementing the proposed approaches, we use the results by *Initial-Setup* and *Greedy-Shared-Sensor* to construct *k-EP* barriers. Then, assume that current constructed network have failure sensors because of energy depletion. An objective of the proposed approaches is to determine moving sensors to recover from failures to solve *MinTMove* problem. When implementing our approach, we first find the set of failed sensors. And each failure sensor in the set is replaced with the chosen sensor.

5.3.1. Uncovered-Sensor-Movement approach

Algorithm 4. *Uncovered-Sensor-Movement.*

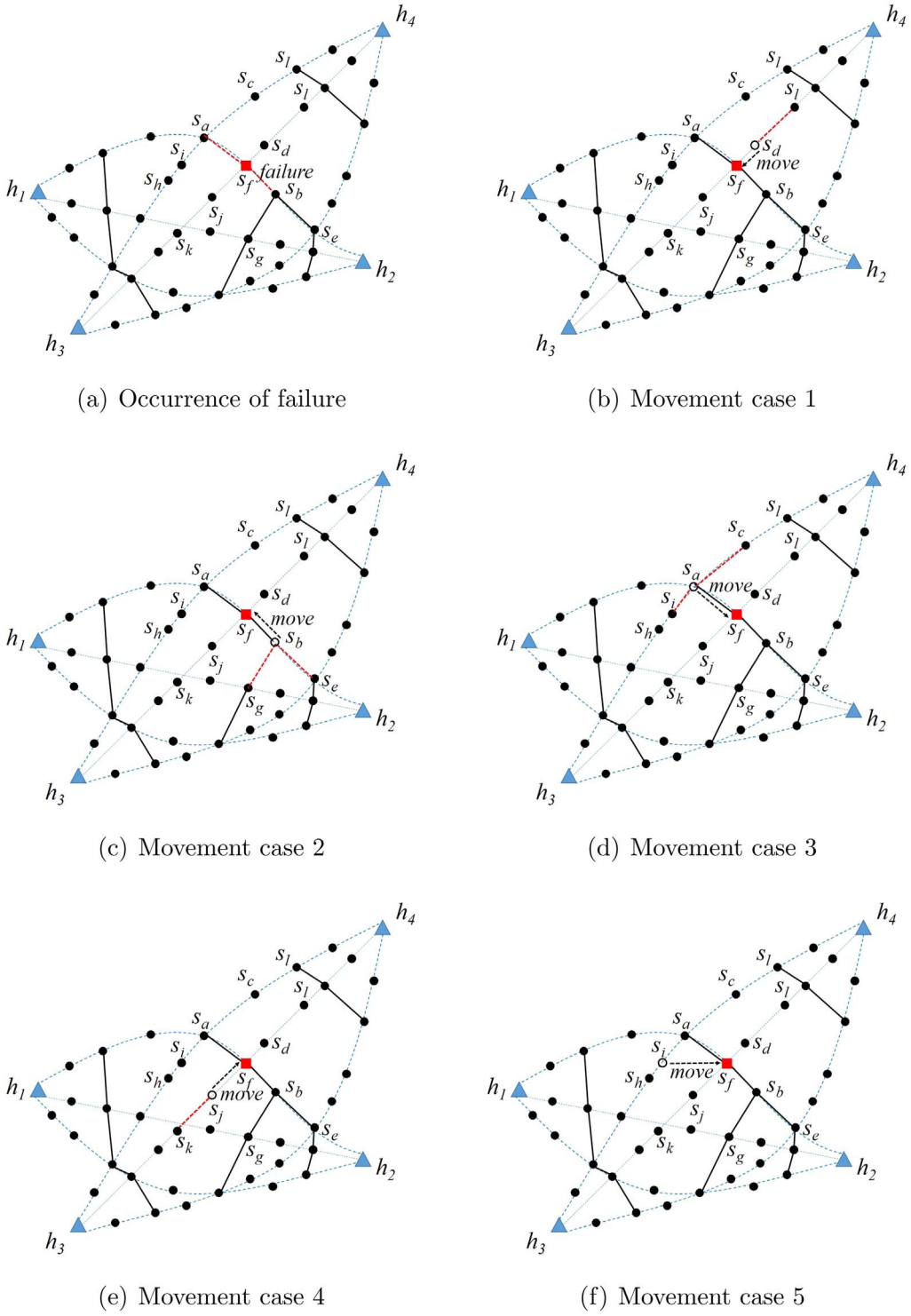


Fig. 4. An example of sensor failure and cases of moving sensor selection to recover the failure. (For interpretation of the references to color in this figure, the reader is referred to the web version of this paper.)

Inputs: S, H, r, k, n, Output: *totalmove* or *failure*
1: set a set of failure sensors: $S_{failures}$= ∅;
2: set a set of uncovered sensors: $S_{uncovered}$=∅;
3: set total movement distance *totalmove*=0;
4 identify all failure sensors and add them to $S_{failures}$;
5: identify all unused sensors at current e-barriers and add them to $S_{uncovered}$;
6: **if** $|S_{failures}| > |S_{uncovered}|$ **then**
7: return *failure*;
8: **end if**
9: **while** $S_{failures} \neq \emptyset$ **do**
10: set movement distance d=0;
11: choose a failure sensor s_f from $S_{failures}$;
12: find the closest sensor s_c from $S_{uncovered}$ with s_f;
13: calculate the distance m between the s_c and s_f;
14: *totalmove*=*totalmove* + d;

15: move s_c to the location of s_f;
16: $S_{failures} \leftarrow S_{failures} - s_f$;
17: $S_{uncovered} \leftarrow S_{uncovered} - s_c$;
18: **end while**
19: return *totalmove*;

Basically, approach 1 of *MinTMove* problem, *Uncovered-Sensor-Movement*, is to find pairs of closest available sensors with failure sensors. We consider those available sensors as the set of uncovered sensors by any *e*-barriers for every hub. The steps of *Uncovered-Sensor-Movement* are as follows:

- Identify failure sensors $S_{failures}$ from current *k-EP* barriers.
- Find unused sensors $S_{uncovered}$ at current *e*-barriers.
- Select a failure sensor s_f from $S_{failures}$.
- Search for the closest sensor s_c from $S_{uncovered}$ with s_f and calculate the moving distance between s_c and s_f.
- Add the distance to total movement.
- Move s_c to the location of s_f;
- Remove s_c from $S_{uncovered}$ and s_f from $S_{failures}$, respectively.

The above steps are iterated until all failure sensors are recovered and the complete *k-EP* barriers are formed successfully. The pseudo-code is described in Algorithm 4 in more detail.

5.3.2. Verified-Sensor-Movement approach

Our approach 2, *Verified-Sensor-Movement*, is to consider selecting a sensor from active sensors used by current *k-EP* barriers first. If the selected sensor's movement into the location of failure sensor does not occur any lost connection with current network, the sensor can move and can be used for recovering current failure sensor. The steps of *Verified-Sensor-Movement* are as follows:

- Identify failure sensors $S_{failures}$ from current *k-EP* barriers.
- Find active sensors S_{active} at current *e*-barriers.
- Find unused sensors $S_{uncovered}$ at current *e*-barriers.
- Select a failure sensor s_f from $S_{failures}$.
- Search for the closest sensor s_c with s_f.
- If s_c is a member of S_{active}, verify whether there is any lost connection with current neighbors of s_c when s_c moves to the location of s_f.
- If there is no lost connection, calculate the moving distance between s_c and s_f.
  1. Add the distance to total movement.
  2. Move s_c to the location of s_f and remove s_f from $S_{failures}$.
- If there is lost connection, find the next closest sensor s_c from $S_{uncovered}$ with s_f and calculate the moving distance between s_c and s_f.
  1. Add the distance to total movement.
  2. Move s_c to the location of s_f.
  3. Remove s_c from $S_{uncovered}$ and s_f from $S_{failures}$, respectively.

Algorithm 5. *Verified-Sensor-Movement*.

Inputs: S, H, r, k, n, Output: *totalmove* or *failure*
1: set a set of failure sensors: $S_{failures}= \emptyset$;
2: set a set of active sensors in *e*-barriers: $S_{active}= \emptyset$;
3: set a set of uncovered sensors: $S_{uncovered}=\emptyset$;
4: set a total movement distance *totalmove*=0;
5: identify all failure sensors and add them to $S_{failures}$
6: identify all active sensors at current *e*-barriers and add them to S_{active}
7: identify all unused sensors at current *e*-barriers and add them to $S_{uncovered}$
8: **if** $|S_{failures}| > |S_{active} + S_{uncovered}|$ **then**
9: return *failure*
10: **end if**
11: **while** $S_{failures} \neq \emptyset$ **do**
12: set a movement distance d=0;
13: choose a failure sensor s_f from $S_{failures}$;
14: find the closest sensor s_c with s_f;
15: **if** $s_c \in S_{active}$ **then**
16: **if** there is no lost connection for movement from s_c to s_f **then**
17: calculate the distance d between the s_c and s_f;
18: *totalmove=totalmove* + d;
19: move s_c to the location of s_f;
20: $S_{failures} \leftarrow S_{failures} - s_f$;
21: **else if** there exists lost connection **then**
22: find the next closest sensor s_c from $S_{uncovered}$ with s_f;
23: *totalmove=totalmove* + d;
24: move s_c to the location of s_f;
25: $S_{uncovered} \leftarrow S_{uncovered} - s_c$;
26: $S_{failures} \leftarrow S_{failures} - s_f$;
27: **end if**
28: **end if**
29: **end while**
30: return *totalmove*

The above steps are iterated until all failure sensors are recovered and *k-EP* barriers are generated completely. The pseudocode is represented in Algorithm 5 in more detail.

6. Performance evaluation

6.1. Experimental evaluation

In this section, we analyze performances of our proposed approaches to solve *MinSkEP* problem and *MinTMove* problem, respectively. Because this is an initial work on the proposed barrier type of event-driven partial barrier, we simulate and evaluate the results of the proposed approach by ad hoc simulator based on Unix system with a use of C++ language instead of the platform with a real deployment of real mobile sensors. Then, we explain the specifications for our simulation environment as follows. For constructing event-driven partial barriers with resilience, we have used the Unix server which has four Dual Core AMD Opteron Processor 285 with cache size 1024 KB, CPU op-mode with 32 bit and 64-bit and L2 cache with 1024K. The codes based on C++ are compiled at the server and the simulation results were obtained. Also, based on those results, the result graphs have been created by MATLAB.

In our simulations, we also considered various square shaped-areas 500×500 m^2, 500×400 m^2 and 500×300 m^2, respectively. Within each area, a set S of sensor nodes with size n are randomly deployed and a set of H hubs with size m are considered with random locations initially. Due to the failure possibility of *Initial-Setup*, our simulation results are obtained from successful 100 different initial graphs after implementing *Initial-Setup* with given input S, H, r, k. Also, each simulation in the result graph represents the average result of the 100 different graph sets in the given area. The number of sensors n which we have used in the simulation is 100 and the number of hubs m is ranging from 4 to 14. Also, a communication range of the sensor, r, is ranging from 50 to 60.

Moreover, for approaches of *MinTMove* problem, we considered that the number of failure sensors is ranging from 1 to 6. Our simulation consists of four different scenarios largely: two scenarios are implemented to analyze approaches of *MinSkEP* problem. Others are simulated to evaluate approaches of *MinTMove* problem. Now, we evaluate the performances by those different scenarios.

In our first simulation, we implemented *Greedy-Shared-Barrier* and *Greedy-Shared-Sensor* with different k values and sensing range r

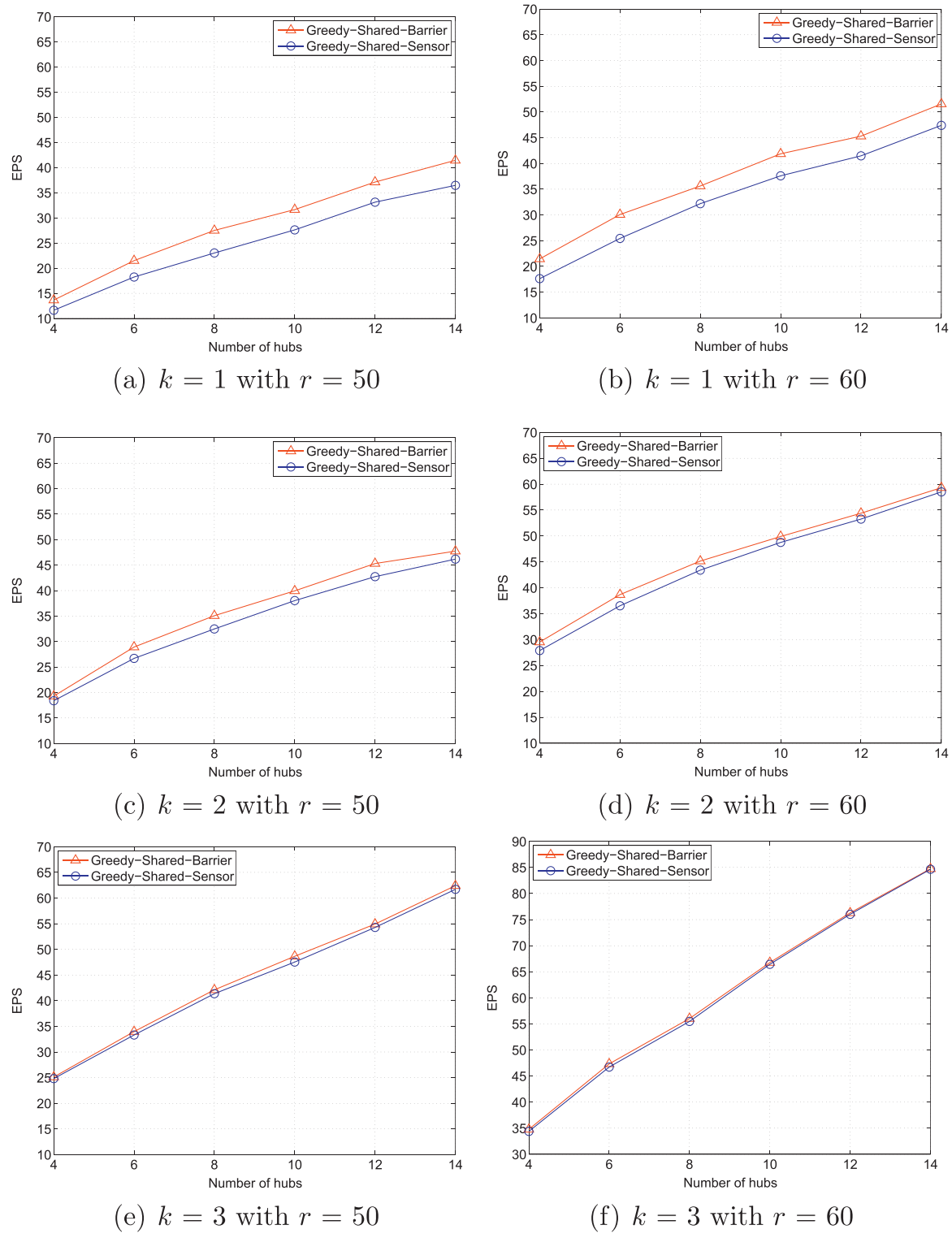


Fig. 5. Comparison of *EPS* for different k values and different sensing range r by *Greedy-Shared-Barrier* and *Greedy-Shared-Sensor* in 500×500 area.

to solve *MinSkEP* problem in the 500×500 m^2 area. Both approaches return the result *EPS*, the number of sensors consisting of *k-EP* barriers. Fig. 5(a), (c), and (d) shows the performance with r=50 by different k=1, 2, 3. Also, Fig. 5 (b), (d), and (f) describes the results with r=60 by different k=1, 2, 3. At these experiments, we verified that *Greedy-Shared-Sensor* shows better performances than *Greedy-Shared-Barrier*. It follows that *Greedy-Shared-Sensor* returns smaller *EPS* than *Greedy-Shared-Barrier*. When k=1, *Greedy-Shared-Sensor* outperforms *Greedy-Shared-Barrier*. And when k=3, *Greedy-Shared-Sensor* shows a slightly better result than *Greedy-Shared-Barrier*.

In the second experiment, we consider various field sizes of the network. So, we consider various square shaped areas, 500×500 m^2, 500×400 m^2 and 500×300 m^2, respectively. In those areas, we consider r=50 and k=2 as parameters. As it can be seen from Fig. 6, *Greedy-Shared-Sensor* shows better performances than *Greedy-Shared-Barrier* for all areas. Also, through this simulation, we have checked that *EPS* in 500×300 m^2 is greater than 500×500 m^2 as a whole.

At the third group of simulations, in order to solve *MinTMove*

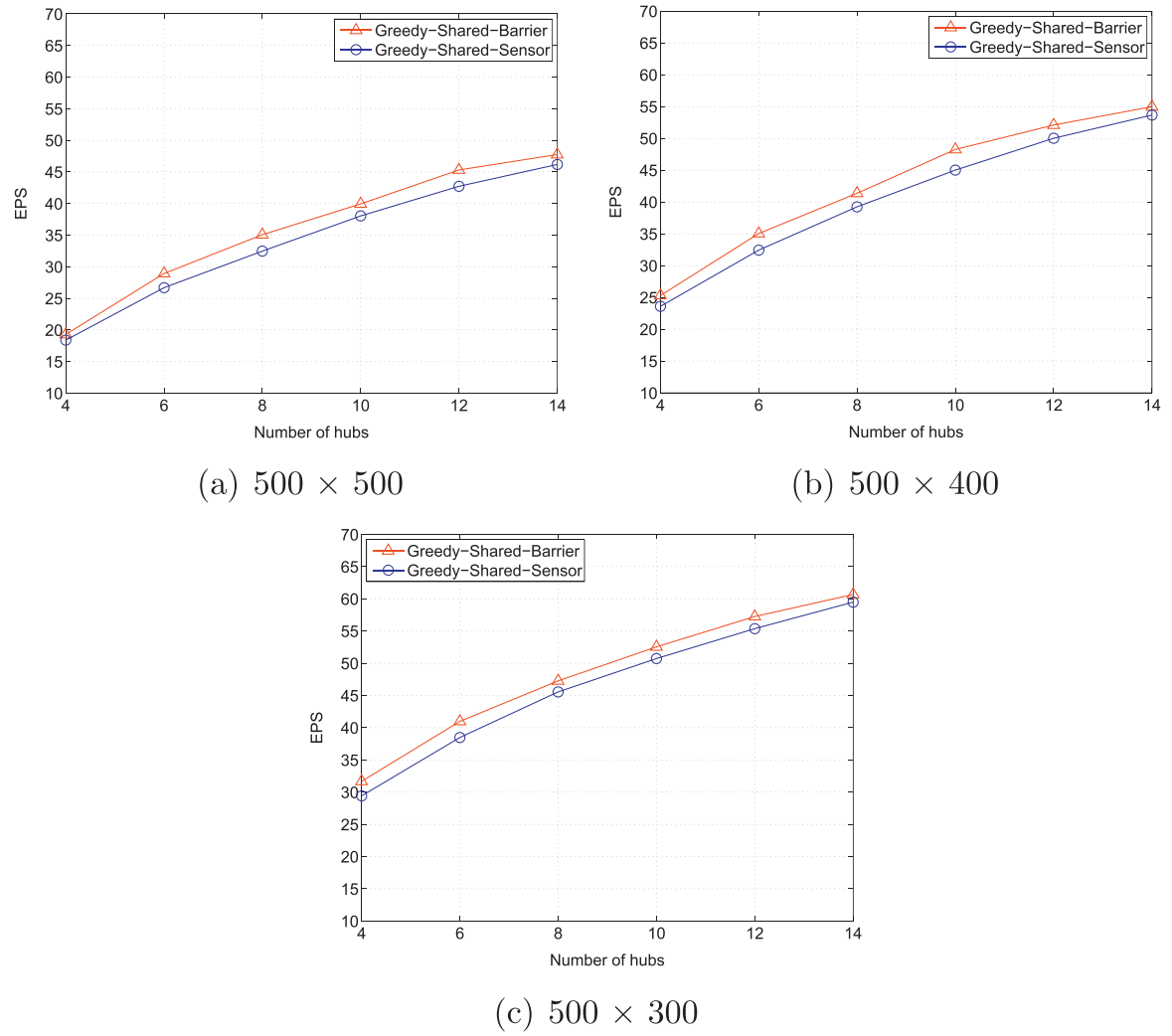


Fig. 6. Comparison of *EPS* for different area sizes with r=50 and with k=2 by *Greedy-Shared-Barrier* and *Greedy-Shared-Sensor*.

problem, we implemented *Uncovered-Sensor-Movement* and *Verified-Sensor-Movement* with different k values and communication ranges in the 500×500 m^2 area. We considered that the number of failure sensors is ranging from 1 to 6 with the number of hubs=6. Both algorithms return total movement of mobile sensors to maintain complete *k-EP* barriers as well as to recover those failures. As you can see Fig. 7, Fig. 7(a), (c), and (d) shows the total movement with r=50 by different k=1, 2, 3. And Fig. 7(b), (d), and (f) describes the results with r=60 by different k=1, 2, 3. At these experiments, we have checked that *Verified-Sensor-Movement* outperforms *Uncovered-Sensor-Movement*. Especially, as the number of failure sensors increases, *Verified-Sensor-Movement* shows much better performance than *Uncovered-Sensor-Movement*.

At the fourth simulation, both *Uncovered-Sensor-Movement* and *Verified-Sensor-Movement* are performed with different area sizes of the network. Therefore, we implemented both approaches at various square shaped areas, 500×500 m^2, 500×400 m^2 and 500×300 m^2, respectively. Also, as parameters, r is set to 50 and k is also set to 2. As Fig. 8, *Verified-Sensor-Movement* shows better results than *Uncovered-Sensor-Movement* at all considered areas. Furthermore, we can verify that *Verified-Sensor-Movement* returns better performance of total movement than *Uncovered-Sensor-Movement* as the number of failure sensor nodes increases in the network.

In summary of our experiments, we have performed four different groups of simulations. The first two groups had the goal of pointing out the behavior of *Greedy-Shared-Barrier* and *Greedy-Shared-Sensor* approaches to solve *MinSkEP* problem. The other two groups are to evaluate the performance of *Uncovered-Sensor-Movement* and *Verified-Sensor-Movement* to solve *MinTMove* problem. On the one hand, through the first two groups of experiments, we can observe that *Greedy-Shared-Sensor* overall shows a better performance with the less value of *EPS* than *Greedy-Shared-Barrier* when they construct k-event-driven partial barriers in those areas with different sizes. Also, for both approaches, *EPS* value increases as the number of hubs increases because an addition of new hubs basically indicates an increment of both the number of sensors on paths and e-barriers to be considered between hubs. Specifically, we have an observation that the difference of *EPS* is smaller as the k value increases in k-event-driven partial barriers. So, we could check that when k is 1, we have the biggest difference of EPS between two approaches. For the reason, we analyze it as follows. The proposed approaches use the strategy of choosing the e-barriers with possible more common sensors among hubs because even-driven partial barriers basically has the property of dependence. When k is 1, the selection of sensor nodes to form k-event-driven partial barriers from e-barriers can be different between *Greedy-Shared-Barrier* and *Greedy-Shared-Sensor*. However, as the k value increases, the selection of sensors from e-barriers may not be different between two approaches. If most of common sensors have been chosen in the case of the small k value (i.e. k=1) and the number of potential choices will be decreased at the bigger k value (i.e. k=2, 3),

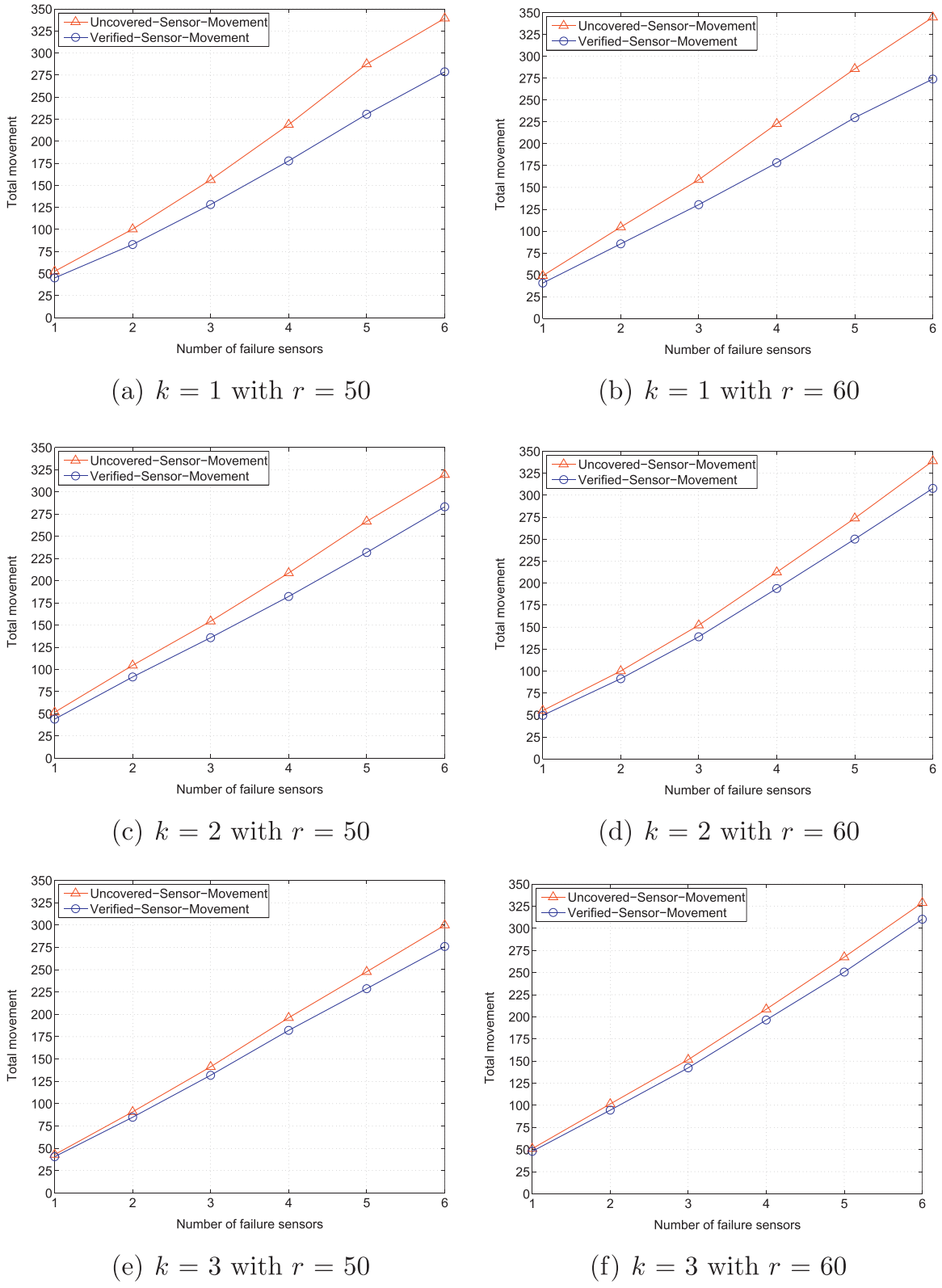


Fig. 7. Comparison of total movement for different k values and different range r with the number of hubs=6 by *Uncovered-Sensor-Movement* and *Verified-Sensor-Movement* in 500×500 area.

then we may finally reach the status that current e-barriers have no common sensors. From the status, the selections by both approaches will not be different largely.

On the other hand, about the other two groups of simulations, we can observe that *Uncovered-Sensor-Movement* overall shows a better performance for the total movement than *Verified-Sensor-Movement* because *Verified-Sensor-Movement* have more flexibility by considering active sensors at current e-barriers as well as unused sensors than

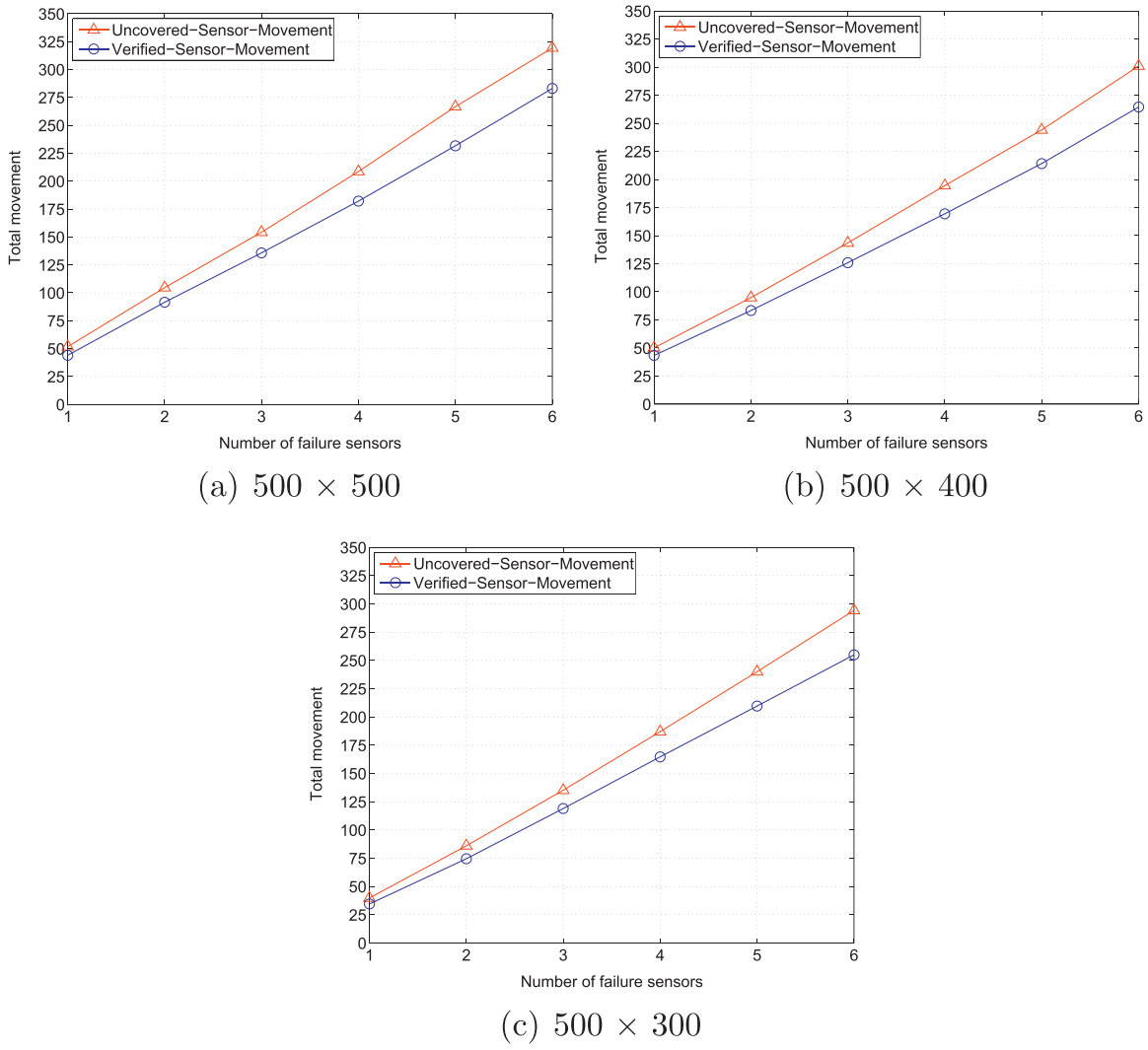


Fig. 8. Comparison of total movement for different area sizes with r=50 and with k=2, the number of hubs=6 by *Uncovered-Sensor-Movement* and *Verified-Sensor-Movement*.

Uncovered-Sensor-Movement when they decide a moving sensor to recover the failure. Also, we could check that for both *Uncovered-Sensor-Movement* and *Verified-Sensor-Movement*, the total movement by sensors increases as the number of failure sensors since the increment of failure sensors indicates the additional movement to recover the failures. In particular, we observe that the difference of total movement is smaller as the k value increases. For the reason, as k value increases, the flexibility of selecting from active sensors at e-barriers at *Verified-Sensor-Movement* is decreased. Hence, we could check that when k is small, we get the result of the big difference of total movement between two approaches.

6.2. Complexity analysis of the proposed approaches

In this section, we also analyze the complexity of the proposed algorithms (i.e. Algorithms 2–5). Let us assume that the total number of sensors is n and the total number of e-barriers is q. Also, assume that the number of hubs is m, where $n > q$ and $n > m$.

If we first consider the complexity of Algorithm 2, it basically implements iterations until k-*EP* barriers are constructed. For those iterations, the worst case is n_e iterations because we may activate all e-barriers. Then, at each iteration, we choose e_{max} from e-barriers. To find e_{max}, we need q comparisons. Then, we activate e_{max} as well as e-barriers affected by sensors within e_{max}. To activate those barriers, we need to consider two loops. One loop is from 1 to q. Within the loop, we consider the inner loop which is from 1 to n. So, the total number of loop iterations is (q*n). Then, we should find a sensor set S_{act} which is covered by the activated e-barriers. To do this, we also have two loops. One loop is from 1 to q. Within the loop, we consider the nested loop which is from 1 to n. So, the total number of loop iterations is (q*n). Then, we update *EPS* by adding $|S_{act}|$, which is $O(1)$. If so, the total number of iterations will be q*(q + (q*n) + (q*n) + 1). It follows that $q^2 + 2q{\cdot}n$. Since q and 1 are constants, the asymptotic upper bound is $O(n)$. So, the complexity of Algorithm 2 is considered as $O(n)$.

Secondly, if we consider the complexity of Algorithm 3, it also implements iterations until k-*EP* barriers are generated. Similarly, for those iterations, the number of the worst case will be q. At each iteration, Algorithm 3 selects s_{max} with the largest frequency f_{max}. To do this, we have two loops. Outer loop is from 1 to n. Inner loop is from 1 to m. So, the total number of the loop iterations is n*m. Then, we

activate *e*-barriers affected by the found s_{max}. To activate those barriers, two loops are performed. The outer loop is from 1 to q. The inner loop is from 1 to n. So, the total number of loop iterations is $(q*n)$. As the next step, Algorithm 3 identifies S_{act} that is included in the activated *e*-barriers. To do this, we have two loops. The outer loop is from 1 to q. The inner loop which is from 1 to n. So, the total number of loop iterations is $(q*n)$. Then, we update EPS by adding $|S_{act}|$, which is $O(1)$ step. Based on the above part, the total number of iterations will be $q*((n*m) + (q*n) + (q*n) + 1) = (q \cdot m)n + 2q \cdot n + q^2 + q$. Since q and m are constants, the asymptotic upper bound is $O(n)$. Hence, as asymptotic upper bound as the worst case, the complexity of Algorithm 3 is considered as $O(n)$.

For the proposed algorithms 4 and 5, suppose that the total number of sensors is n and the total number of *e*-barriers is q. Also, assume that the size of the set of failure sensors $S_{failures}$ is t. Note that $n > q$ and $n > t$.

Next, we check the complexity of Algorithm 4. Its iterations are done if the failures are recovered by replacement with sensors. So, the maximum number of iterations is t. For each iteration, we have two independent inner loops. One is to choose a failure s_f from $S_{failures}$ and another is to find the closest sensor from s_c from $S_{uncovered}$ with its distance d. So, the first inner loop is from 1 to t and the second inner loop is from 1 to n. Then, we update *totalmove* by adding d, which is $O(1)$. Then, the total number of iterations of Algorithm 4 is $t*(t + n + 1) = t2 + t \cdot n + t$. Since t is a constant, the complexity of Algorithm 4 is considered as $O(n)$.

Lastly, if we derive the complexity of Algorithm 5, it is implemented by iterations to until the failed sensors are recovered by replacement with sensors. Then, the total number of iterations is t. Also, for each iteration, we consider the following loops. The first loop is to select a failure s_f from $S_{failures}$, which is from 1 to t and the second loop is to find the closest sensor from s_c from $S_{failures}$ with its distance d, which is from 1 to n. If the selected sensor s_c is a part of S_{active} with no lost connection when s_c from to s_f, update *totalmove* by adding d. So, the total number of iterations about this is $(t + n + q + 1)$. Then, if there exists the lost connection, find the next closest s_c from $S_{uncovered}$ and also update *totalmove* by adding d. For this process, the total number of iterations is $(t + n + 1)$. If so, the total number of iterations of Algorithm 4 is as follows: $t*((t + n + q + 1) + (t + n + 1)) = t^2 + (t \cdot n) + (t \cdot q) + t + t^2 + (t \cdot n) + t$ $= 2t \cdot n + (t \cdot q + 2t^2 + 2t)$. Since q and t are constants, the complexity of Algorithm 5 is considered as $O(n)$.

7. Conclusion and future works

In this paper, we introduced a new type of barriers, *k-event-driven partial barriers* or *k-EP* barriers, in which at least k sensors can detect any movements of mobile objects among hubs in the event-driven environment. Also, we introduced *resilient event-driven partial barriers*, which can maintain *k-EP* barriers continuously when some sensors are failed due to energy depletion. And we defined two different problems. One is to minimize the number of sensors to construct *k-EP* barriers. Another is to minimize total movement distance of mobile sensors to recover from failures. To solve the problems, we proposed various approaches and analyzed their relative performance by extensive experiments with various scenarios. As a future work, we plan to extend our *event-driven partial barriers* to different shaped regions with deployment of real mobile sensors. On the real platform, we will study additional impacts and requirements of system when we apply to *event-driven partial barriers* to various shaped areas such as circle, convex hull, etc. Furthermore, we plan to apply *event-driven partial barriers* to an application in smart cities using unmanned aerial vehicles (UAV). Because UAVs fly in the air in smart city, it is possible to consider a combination of UAVs in the air and sensors on land with a view of three-dimensional environment. So, we will introduce the new infrastructure and various approaches. Then, those works will be simulated by practical network simulation environments with practical equipments of UAVs.

References

Abbasi, A., Younis, M., Akkaya, K., 2009. Movement assisted connectivity restoration in wireless sensor and actor networks. IEEE Trans. Parallel Distrib. Syst. 20 (9), 1366–1379.

Akkaya, K., Senel, F., 2009. Detecting and connecting disjoint sub-networks in wireless sensor and actor networks. Ad Hoc Netw. 7 (7), 1330–1346.

Akylidiz, I.F., Kasimoglu, I.H., 2004. Wireless sensor and actor networks: research challenges. Ad hoc Netw. 2, 351–367.

Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E., 2002. Wireless sensor networks: a survey. Comput. Netw. 38 (March (4)), 393–422.

Ban, D., Feng, Q., Han, G., Yang, W., Jiang, J., Dou, W., 2011. Distributed scheduling algorithm for barrier coverage in wireless sensor networks. In: Proceedings of the 2011 Third International Conference on Communications and Mobile Computing (CMC), April.

Bhattacharya, B., Burmester, M., Hu, Y., Kranakis, E., Shi, Q., Wiese, A., 2009. Optimal movement of mobile sensors for barrier coverage of a planar region. Theor. Comput. Sci. 410, 5515–5528.

Cardei, M., Thai, M., Li, Y. Wu, W., 2005. energy-efficient target coverage in wireless sensor networks. In: IEEE Annual International Conference on Computer Communications (INFOCOM), March.

Carrabs, F., Cerulli, R., D'Ambrosio, C., Raiconi, A., 2015. A hybrid exact approach for maximizing lifetime in sensor networks with complete and partial coverage constraints. J. Netw. Comput. Appl. 58, 12–22.

Chen, A., Kumar, S., Lai, T.H., 2005. Barrier coverage with wireless sensors. In: Proceedings of the 11th Annual International Conference on Mobile Computing and Networking (MobiCom).

Chen, J., Li, J., Lai, T.H., 2013. Energy-efficient intrusion detection with a barrier of probabilistic sensors: global and local. IEEE Trans. Wirel. Commun. 12 (9), 4742–4755.

Chen, J., Wang, B., Liu, W., 2015. Constructing perimeter barrier coverage with bistatic radar sensors. J. Netw. Comput. Appl. 57, 129–141.

Chen, A., Zhu, Y., Li, Z., Lai, T.H., Liu, C., 2015. Is one-way barrier coverage achievable using comprehensive sensors? Comput. Commun. 57, 100–114.

Dobrev, S., Durocher, S., Eftekhari, M., Georgiou, K., Kranakis, E., Krizanc, D., Narayanan, L., Opatrny, J., Shende, S., Urrutia, J., 2015. Complexity of barrier coverage with relocatable sensors in the plane. Theor. Comput. Sci. 579, 64–73.

Edmonds, J., Karp, R.M., 1972. Theoretical improvements in algorithmic efficiency for network flow problems. J. ACM 19 (2), 248–264.

Gage, D., 1992. Command control for many-robot systems. In: Proceedings of the Nineteenth Annual AUVS Technical Symposium (AUVS-92).

He, S., Chen, J., Li, X., Shen, X., Sun, Y., 2012. Cost-effective barrier coverage by mobile sensor networks. In: IEEE Annual International Conference on Computer Communications (INFOCOM), pp. 819–827.

He, S., Gong, X., Zhang, J., Chen, J., Sun, Y., 2014. Curve-based deployment for barrier coverage in wireless sensor networks. IEEE Trans. Wirel. Commun. 13 (2), 724–735.

Huang, C., Tseng, Y., 2003. The coverage problem in a wireless sensor network. In: Proceedings of ACM International Workshop on Wireless Sensor Networks and Applications (WSNA).

Kim, H., Ben-Othman, J., 2016. On resilient event-driven partial barriers in mobile sensor networks. In: Proceedings of IEEE International Conference on Communications (ICC).

Kim, H., Cobb, J.A., 2013. Maximum lifetime reinforced barrier-coverage in wireless sensor networks. In: Proceedings of the 19th IEEE International Conference on Networks (ICON).

Kim, H., Cobb, J.A., 2015. Optimization algorithms for transmission range and actor movement in wireless sensor and actor networks. Comput. Netw. 36, 116–133.

Kim, H., Son, J., Chang, H.J., Oh, H., 2016. Event-driven partial barriers in wireless sensor networks. In: Proceedings of IEEE International Conference on Computing, Networking and Communications (ICNC).

Kloder, S., Hutchinson, S., 2007. Barrier coverage for variable bounded-range line-of-sight guard. In: Proceedings of the IEEE International Conference on Robotics and Automation, Roma, Italy, pp. 391–396, April.

Kong, L., Liu, X., Li, Z., Wu, M., 2010. Automatic barrier coverage formation with mobile sensor networks. In: IEEE International Conference on Communications (ICC), pp. 1–5.

Kumar, S., Lai, T.H., Arora, A., 2005. Barrier coverage with wireless sensors. In: Proceedings of the 11th Annual International Conference on Mobile Computing and Networking (MobiCom), pp. 284–298.

Kumar, S., Lai, T.H., Posner, M.E., Sinha, P., 2010. Maximizing the lifetime of a barrier of wireless sensors. IEEE Trans. Mob. Comput. 9 (August (8)).

Li, W., Zhang, W., 2015. Coverage hole and boundary nodes detection in wireless sensor networks. J. Netw. Comput. Appl. 48, 819–827.

Li, L., Zhang, B., Shen, X., Zheng, J., Yao, Z., 2011. A study on the weak barrier coverage problem in wireless sensor networks. Comput. Netw. 55 (3), 711–721.

Li, Y., Vu, C., Ai, C., Chen, G., Zhao, Y., 2011. Transforming complete coverage algorithms to partial coverage algorithms for wireless sensor networks. IEEE Trans. Parallel Distrib. Syst. 22 (4), 695–703.

Li, J., Chen, J., Lai, T.H., 2012. Energy-efficient intrusion detection with a barrier of probabilistic sensors. In: Proceedings of the IEEE Conference on Computer Communications (INFOCOM '12), Orlando, FL, USA, pp. 118–126, March.

Liu, B., Dousse, O., Wang, J., Saipulla, A., 2008. Strong barrier coverage of wireless sensor networks. In: Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc).

Md, S., Malek, B., Sadik, Md.M., Rahman, A., 2016. On balanced k-coverage in visual sensor networks. J. Netw. Comput. Appl. 72, 72–86.

Saipulla, A., Liu, B., Wang, J., 2008. Barrier coverage with air-dropped wireless sensors. In: Proceedings of the 28th IEEE Military Communications Conference (MILCOM), pp 1–7.

Saipulla, A., Westphal, C., Liu, B., Wang, J., 2009. Barrier coverage of line-based deployed wireless sensor networks. In: Proceedings of the 28th IEEE Conference on Computer Communications (INFOCOM).

Saipulla, A., Liu, B., Xing, G., Fu, X., Wang, J., 2010. Barrier coverage with sensors of limited mobility. In: ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc).

Shen, J., Wang, Z., Wang, Z., 2015. Fault tolerant line-based barrier coverage formation in mobile wireless sensor networks. Int. J. Distrib. Sens. Netw.. http://dx.doi.org/10.1155/2015/930585.

Vecchio, M., Lopez-Valcarce, R., 2015. Improving area coverage of wireless sensor networks via controllable mobile nodes: a greedy approach. J. Netw. Comput. Appl. 48, 1–13.

Vu, C.T., Chen, G., Zhao, Y., Li, Y., 2009. A universal framework for partial coverage in wireless sensor networks. In: IEEE International Performance Computing and Communications Conference (IPCCC), December.

Wang, G., Cao, G., Porta, T.L., Zhang, W., 2005. Sensor relocation in mobile sensor networks. In: Proceedings of IEEE International Conference on Computer Communications (INFOCOM), Miami, FL, March.

Wang, S., Mao, X., Tang, S., Li, X., Zhao, J., Dai, G., 2011. On movement-assisted connectivity restoration in wireless sensor and actor networks. IEEE Trans. Parallel Distrib. Syst. 22 (4), 687–694.

Xiang, Y., Xuan, Z., Tang, M., Zhang, J., Sun, M., 2016. 3D space detection and coverage of wireless sensor network based on spatial correlation. J. Netw. Comput. Appl. 61, 93–101.

Yick, J., Mukherjee, B., Ghosal, D., 2008. Wireless sensor network survey. Comput. Netw. 52 (August (12)), 2292–2330.

Zhang, H., Hou, J., 2004. On deriving the upper bound of α-lifetime for large sensor networks. In: Proceedings of the ACM International Symposium on Mobile Ad-hoc Networking and Computing (MobiHoc).