Minimizing Movement for Target Coverage and Network Connectivity in Mobile Sensor Networks

Zhuofan Liao, Jianxin Wang, *Senior Member, IEEE,* Shigeng Zhang, Jiannong Cao, *Senior Member, IEEE,* and Geyong Min, *Member, IEEE*

Abstract—Coverage of interest points and network connectivity are two main challenging and practically important issues of Wireless Sensor Networks (WSNs). Although many studies have exploited the mobility of sensors to improve the quality of coverage and connectivity, little attention has been paid to the minimization of sensors' movement, which often consumes the majority of the limited energy of sensors and thus shortens the network lifetime significantly. To fill in this gap, this paper addresses the challenges of the Mobile Sensor Deployment (MSD) problem and investigates how to deploy mobile sensors with minimum movement to form a WSN that provides both target coverage and network connectivity. To this end, the MSD problem is decomposed into two sub-problems: the Target COVerage (TCOV) problem and the Network CONnectivity (NCON) problem. We then solve TCOV and NCON one by one and combine their solutions to address the MSD problem. The NP-hardness of TCOV is proved. For a special case of TCOV where targets disperse from each other farther than double of the coverage radius, an exact algorithm based on the Hungarian method is proposed to find the optimal solution. For general cases of TCOV, two heuristic algorithms, i.e., the *Basic* algorithm based on clique partition and the *TV-Greedy* algorithm based on Voronoi partition of the deployment region, are proposed to reduce the total movement distance of sensors. For NCON, an efficient solution based on the Steiner minimum tree with constrained edge length is proposed. The combination of the solutions to TCOV and NCON, as demonstrated by extensive simulation experiments, offers a promising solution to the original MSD problem that balances the load of different sensors and prolongs the network lifetime consequently.

Index Terms—Wireless sensor networks, target coverage, connectivity, mobile sensors, energy consumption.

1 Introduction

Wireless sensor networks (WSNs) are currently used in a wide range of applications including environmental monitoring [1] and object tracking [2]. Target coverage and connectivity are two main challenging and practically important issues of WSNs. Target coverage aims to cover a set of specified points of interest in the deployment region of a WSN. It characterizes the monitoring quality of the network [3]. Connectivity is necessary for sensors in a WSN to collect data and report data to the sink node. However, WSNs formed by randomly distributed wireless sensor nodes often cannot provide satisfactory coverage quality and cannot guarantee the connectivity of the network. In recent years, sensor mobility has been exploited to improve the coverage quality and connectivity in randomly deployed WSNs by relocating some mobile sensors to new positions to enhance the coverage quality and the connectivity of the network [4]–[8].

In this paper, we addresses a practically important problem of minimizing sensors' movement to achieve both target coverage and network connectivity in mobile sensor networks. As sensors are usually powered by energy-limited batteries and thus severely power-constrained, energy consumption should be the top consideration in mobile sensor networks. Specially, movement of sensors should be minimized to prolong the network lifetime because sensor movement consumes much higher energy than sensing and communication do [6], [9]. However, most of the existing studies aimed at improving the quality of target coverage, e.g., detecting targets with high detection probabilities, lowering false alarm rate and detection delay. Little attention has been paid to minimization of sensor movement. To fill in this gap, this study focuses on moving sensors to cover discrete targets and form a connected network with minimum movement and energy consumption.

To this end, we first formulate the Mobile Sensor Deployment (MSD) problem with the aim of deploying mobile sensors to provide target coverage and network connectivity with minimum movement. The MSD problem is then decomposed into two sub-problems: Target COVerage (TCOV) and Network CONnectivity (NCON). Combining the solutions to the two sub-problems, we achieve an efficient solution to the MSD problem. The main contributions of this paper are summarized as follows:

1) We prove the NP-hardness of the TCOV problem. For a special case of TCOV in which targets disperse from each other by more than double of the coverage radius, an exact algorithm based on the

- *Zhuofan Liao, Jianxin Wang, and Shigeng Zhang are with the School of Information Science and Engineering, Central South University, Changsha, China. E-mail: {liaozf, jxwang, sgzhang}@csu.edu.cn. Zhuofan Liao is also with the School of Computer and Communication Engineering, Changsha University of Science and Technology, ChangSha, China.*
- *Jiannong Cao is with the Department of Computing, the Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong. E-mail: csjcao@comp.polyu.edu.hk.*
- *Geyong Min is with the Department of Computing, University of Bradford, U.K. E-mail:g.min@brad.ac.uk.*

extended Hungarian method is proposed to find the optimal solution to TCOV.

2) For the general case of TCOV, two heuristic algorithms are proposed: the Basic algorithm based on clique partition, and the TV-Greedy algorithm based on Voronoi partition diagram of target points. The Basic algorithm reduces the total movement distance by minimizing the number of sensors to be moved. The TV-Greedy algorithm minimizes the total movement distance by grouping and dispatching sensors according to their proximity to targets in the Voronoi diagram.
3) For the NCON problem, first an edge length constrained Steiner tree is constructed to determine the Steiner points that are needed to connect the coverage sensors and the sink, then the extended Hungarian method is used to find the the optimal sensors to move to these points.
4) Extensive simulation experiments are conducted to evaluate the performance of the proposed algorithms. The results demonstrate that the combination of the solutions to TCOV and NCON offers a promising solution to the original MSD problem, as well as balances the load of different sensors and prolongs the network lifetime consequently.

The remainder of this paper is organized as follows. Section 2 reviews related work. The system model and problem description are given in Section 3. In Sections 4 and 5, the solutions to the TCOV problem and the NCON problem are presented, respectively. Theoretical analyses on the performance of the proposed solutions are given in Section 6. Section 7 presents the simulation results and investigates the impact of network parameters on the performance of the proposed solutions. Section 8 discusses the extension of the proposed solutions to be applied to scenarios when the free mobility model is not applicable. Finally, Section 9 concludes this paper.

2 Related Work

With the emergence of mobile sensors, extensive researches have been promoted on target coverage of WSNs. According to different application scenarios, the existing studies can be classified into three categories: (1) route patrol for collecting data from fixed targets [10]–[12], (2) detection of mobile targets [4], [5], [13], and (3) target coverage in dynamic environments [14], [15]. In these studies, mobile sensors move actively to improve the surveillance quality, but the optimization of sensor movement is not explicitly considered. Reactive mobility is exploited to improve the quality of target detection in [6], but the movement of sensors is not considered as the primary optimization objective. In [7] mobile sensors are scheduled to replace failed static sensors in order to guarantee coverage ratio with minimum movement distance. But each sensor concerned in [7] can cover only one target and the maximum moving distance for each mobile sensor is limited. In [16], an optimal velocity schedule is proposed to minimize energy consumption in movement when the road condition is uniform.

Many research efforts have also been made to improve the area coverage with mobile sensors with the aim of maximizing the covered area. In [17], Voronoi diagrams are used to detect coverage holes. After that, sensors are dispatched to cover these holes. As a result, the area coverage ratio is improved. Further, a multiplicatives weighted Voronoi diagram is used to discover the coverage holes corresponding to different sensors with different sensing ranges [18]. However, Voronoi diagrams in these studies are constructed according to the position of mobile sensors, and thus need to be recomputed after each round of sensor movement. In [19], mobile sensors are used to improve energy efficiency of sensors in area coverage. In this work, when destinations have been determined, mobile sensors are designed to move along the shortest path to minimize the energy consumption. Given designated destinations, k-coverage is studied in [20]. In this work, a competition scheme is proposed to minimize energy consumption in movement. In order to balance the energy consumption, sensors are dispatched in a relay-style [21], or following the matching results of an energy-weighted graph [22]. In these studies, destinations of mobile sensors are given in advance, and the energy efficiency is considered in the path finding process.

Mobility of sensors could also be exploited to enhance network connectivity after the coverage stage is completed. In [23], a triangular deployment strategy is proposed to dispatch sensors to connect the network after deploying mobile routers to maximize the coverage area. In the proposed strategy, sensors move along the shortest path to the corresponding triangular vertices in order to save energy. In [24], the authors considered a hybrid network consisting of both static and mobile sensors. It first divides the static sensors into groups as large as possible, and then seeks the minimum number of mobile sensors to connect these static sensor groups. In [25], a sensor node relocation approach is proposed to maintain connectivity between a region of interest and a center of interest outside the deployment region where a particular event happens.

The originality of this study and difference from existing work. (1) In this work, sensors move reactively and each sensor can cover more than one target, which is more general in practice, but also makes the problem more complicated. (2) The Voronoi diagram of targets is adopted to find the nearest sensor, which avoids blind competition among mobile sensors. Besides, because our solution generates the Voronoi diagram according to the position of targets, it does not require re-computation of the Voronoi diagram as the targets are static. This contributes to the lower complexity of the proposed solution. (3) Destinations of mobile sensors are unknown, which should be computed by our algorithms. When mobile sensors move to these destinations, both target coverage and network connectivity are satisfied. (4) In

order to investigate the impact of network parameters on the performance of our algorithms, analyses and evaluations are given according to the simulation experiment results, which provides a reference for practical engineering and theoretical basis for mobile sensor networks design.

3 System Model and Problem Statement

3.1 System Model

In the system model addressed in this study, there are m targets $T = \{t_1, \ldots, t_m\}$ with known locations to be covered, and n mobile sensors $S = \{s_1, \ldots, s_n\}$ randomly deployed in the task area. The system model works as follows:

1) Every mobile sensor knows its own position via a mounted GPS unit or a localization service in the network. We also assume that, there is a control center, e.g., a sink, which collects sensors' location information and broadcasts movement orders to mobile sensors.
2) The task area is free of obstacles against movement. For the case with obstacles, a sensor is able to choose an appropriate shortest path to the destination to bypass the obstacles on the way. In this work, we focus on determining WHICH sensors should move and WHERE they should move to in order to guarantee both target coverage and network connectivity.
3) *Network model:* Disk model [26] is adopted for both sensing and communication of sensors with the sensing radius r_s and the communication radius r_c, respectively. Each target can be covered by more than one sensor, and each sensor can cover more than one target. A target is said *covered* if and only if there is at least one sensor in the disk of radius r_s centered at the target. The disk is defined as the target's *coverage disk*, and the circle of the coverage disk is called the target's *coverage circle*.
4) *Mobility model:* The free mobility model [6] is adopted. In this model, sensors are able to move continuously in any direction and stop anywhere. The distance that a sensor moves is used to present the sensor's energy consumption incurred in the movement. The movement distance of sensor s to cover target t is $dist(s,t) - r_s$, where $dist(s,t)$ is the Euclidean distance between s and t. Similarly, the movement distance of sensor s_i to connect with sensor s_j is $dist(s_i,s_j) - r_c$, where dist(s_i,s_j) is the distance between s_i and s_j. In the obstacle-free scenario, in order to minimize the movement distance of a sensor to a target, the sensor should move along the straight line from its initial position to the target until it reaches the target's coverage circle. Fig. 1 illustrates an example of this straight line movement pattern.

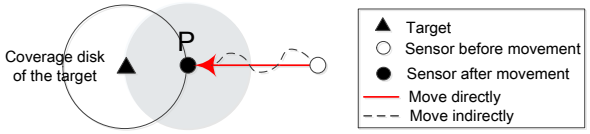


Fig. 1. A sensor should move along the straight line between its initial position and the target to minimize the movement distance. In this example, the destination of the mobile sensor is P.

3.2 Problem Statement

3.2.1 Problem Definition

With the aforementioned system model, the formal definition of the MSD problem can be given as follows.

Definition 1. *Mobile Sensor Deployment (MSD) problem: Given m targets with known locations and n mobile sensors deployed randomly in the task area, the MSD problem seeks the minimum movement of mobile sensors such that the following objectives are achieved after mobile sensors reach their new positions:*

1) *Every target is covered by at least one mobile sensor.*
2) *The network formed by all the moved sensors is connected.*

The MSD problem concerns two issues, namely target coverage and network connectivity, thus we divide it into two sub-problems and conquer them one by one. First, we focus on deploying mobile sensors to cover targets with minimum movement. These mobile sensors are called *coverage sensors*. Next, we deploy the rest sensors to provide connectivity between coverage sensors and the sink. The definitions of the two sub-problems are given below.

Definition 2. *Target COVerage (TCOV) problem: Given m targets with known locations and n mobile sensors deployed randomly in the task area, move sensors to new positions such that all the targets are covered and the total movement of sensors is minimized.*

Definition 3. *Network CONnectivity (NCON) problem: Given a sink, the set of coverage sensors, and the rest mobile sensors after the TCOV problem is solved, NCON seeks the deployment of the rest mobile sensors to connect coverage sensors and the sink with minimum movement.*

Theorem 1. *In TCOV with free mobility model, in order to minimize the movement distance of mobile sensors, the number of potential positions to which sensors can move is finite.*

Proof: Assume that there are m targets and n sensors in the network. We first consider the simple case in which one sensor covers exactly one target. Under the free mobility model, if one sensor wants to cover a target, it should move along the straight line connecting the sensor and the target and stop at the intersection of the line and the target's coverage circle, as shown in Fig. 1. As there are totally m targets and for each target there

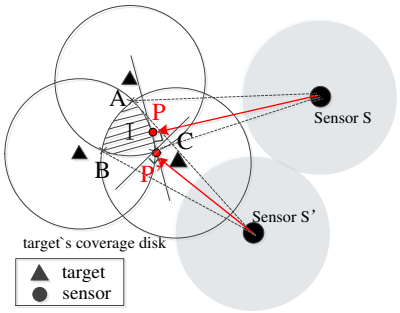


Fig. 2. Targets A, B, and C could be simultaneously covered by a single sensor (S or S'). Among all the points in I, P is closest to S, and thus it should be the destination position of S if S is despatched to cover the three targets. Similarly, P' should be the destination position of S' if S' is despatched to cover the targets.

is exactly one optimal destination, the total number of potential destination points for the sensor is m.

We then consider the general case in which a sensor can cover more the one target. In this case, because several targets could be covered by one sensor simultaneously, their coverage disks intersect with each other. Denote the intersection part of these coverage disks as I. As the coverage disks are convex, the intersection I is also convex [27]. For a sensor outside of I, there exists a unique point closest to the sensor on the boundary of I. For instance, as shown in Fig. 2, the coverage disks of targets A, B, and C intersect with each other. For sensor S, P is the closest point to S in I and thus should be the destination of S if S is dispatched to cover the three targets. Similarly, P' is the unique closest point to S' in I. For k ($1 \leq k \leq m$), the maximum number of k target combinations that could be covered by a single sensor is C_m^k. Thus the possible positions for a sensor to move to cover k targets is at most C_m^k. The total number of potential positions for a sensor to cover multiple targets is thus bounded by $\sum_{k=2}^{m} C_m^k$.

From the above analysis, to cover m targets, the number of potential positions a sensor can move to is upper bounded by $1 + m + \sum_{k=2}^{m} C_m^k = 2^m$, where the term 1 corresponds to the case that the sensor stays at its original position. As there are n sensors, the total number of potential positions the mobile sensors can move to is at most $n*2^m$, which is finite as n and m are both finite. □

According to Theorem 1, when a sensor moves to one of its potential positions, a subset of targets are covered. Thus each potential position of a sensor corresponds to a subset of targets. For all the n sensors, the total number of potential positions is limited by $n*2^m$, which is finite. This conclusion is critical to help prove the hardness of the TCOV problem in the next section.

3.2.2 Hardness of the Problem

To show the hardness of the TCOV problem, we define a special case of TCOV, namely TCOV*, and prove that it is NP-hard. This naturally induces the NP-hardness of the original TCOV problem.

The TCOV* problem is a special case of TCOV. In TCOV*, all the mobile sensors are initially deployed at the same location, which means that sensors start to move at the same point. If there exists a solution to TCOV, then the corresponding TCOV* problem will be solved by deploying mobiles sensors at the same initial position, but the converse does not hold.

Theorem 2. *The TCOV* problem is NP-hard.*

Proof: Denote the power set of T (i.e., the set of all the targets) by $P(T)$. Recall that there are totally $n * 2^m$ potential positions for the n mobile sensors to move to in the TCOV* problem. Each potential position corresponds to a subset of T, i.e., an element in $P(T)$. For each potential position, we assign a weight W to its corresponding element in $P(T)$, which is defined as the movement distance between the mobile sensor and that potential position. Then the decision version of the TCOV* problem can transformed into the following set cover problem: Given the universal set of targets T and a finite number (no more than $n*2^m$) of weighted subsets of T whose union comprises the universe, determine whether there are some subsets whose total weight is less than or equal to W, such that the union of these subsets contains all the elements in T?

The decision version of TCOV* is equivalent to that of the weighted set cover problem [28], which is NP-complete. Therefore, TCOV* is NP-hard. □

Remark 1. *As TCOV* is a special case of TCOV, TCOV is also NP-hard.*

4 Solutions to the TCOV Problem

Although the TCOV problem is NP-hard, there exists a special case that can be solved in polynomial time. In this section, on one hand, we analyze a special case of TCOV and transform it into an *assignment problem* [29] and find the optimal solution; on the other hand, for the general case of TCOV, we propose two heuristic algorithms to solve it: the Basic algorithm based on clique partition, and the TV-Greedy algorithm based on the Voronoi partition diagram of targets.

4.1 Exact Solutions to A Special Case of TCOV

For a special case of TCOV in which the distance between any pair of targets is greater than $2r_s$, an exact solution based on the extended Hungarian algorithm is proposed in our previous work [30]. In this special case, as targets disperse from each other by more than double of the coverage radius, each sensor can cover at most one target. Thus different target needs to be covered by different mobile sensor. The TCOV problem in this scenario could be transferred to the *assignment problem* [29] that is to assign exactly one agent to each task in such a way that the total cost of the assignment

is minimized. However, in the traditional assignment problem the number of agents equals the number of tasks ($n = m$), while in our TCOV problem the number of sensors is usually larger than the number of targets, i.e., $n > m$. To deal with this issue, we extended the Hungarian algorithm proposed in [31] by extending the cost matrix to a $n \times n$ matrix as follows

$$[c_{i,j}]_{n\times n} = \left(\begin{array}{ccc|ccc} c_{1,1} & \cdots & c_{1,m} & 0 & \cdots & 0 \\ \vdots & & \vdots & \vdots & & \vdots \\ c_{n,1} & \cdots & c_{n,m} & 0 & \cdots & 0 \end{array}\right), \quad (1)$$

where $c_{i,j}$ $(1 \le i \le n, 1 \le j \le m)$ is set as the movement distance of moving sensor s_i to cover target t_j, i.e.,

$$c_{i,j} = \begin{cases} dist(s_i, t_j) - r_s & \text{if } dist(s_i, t_j) > r_s, \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

Here $dist(s_i, t_j)$ is the Euclidean distance between s_i and t_j. With this extended cost matrix, the optimal solution to TCOV in the special case could be found in polynomial time by using the Hungarian algorithm [31]. More details on the extended Hungarian algorithm can be found in our previous work [30].

Algorithm 1: The Basic Algorithm

Input: $T = t_1, t_2, ..., t_m$;// Positions of targets
$S = s_1, s_2, ..., s_n$;// Initial positions of sensors
r_s: // The coverage radius
$Boundary = \{f_1, ...f_a\}$;// The boundaries of the task area
Output: tmc;// The total moving cost

```
tmc ← 0; T_initcov ← ∅; S_initcov ← ∅;
for each t_i (1 ≤ i ≤ m) do
    S_tmp ← ∅;
    for each s_j (1 ≤ j ≤ n) do
        if dist(t_i, s_j) ≤ r_s then
            T_inticov = T_inticov ∪ {t_i};
            S_tmp = S_tmp ∪ {s_i};
        S_inticov = S_inticov ∪ {s_j | dist(t_i, s_j) ≤
        dist(t_i, s_i), s_i ∈ S_tmp }
T_needcov = T \ T_initcov; S_rest = S \ S_inticov;
Movecost= GEOCP(T_needcov, S_rest, r_s, Boundary);
(P_dest, S_move, tmc) = extended-Hungarian(T_needcov,
S_rest, Movecost);
return tmc;
```

4.2 Heuristic Solutions to the General Case of TCOV

For the general case of the TCOV problem, we propose two heuristic algorithms to find near optimal solutions.

4.2.1 The Basic Algorithm

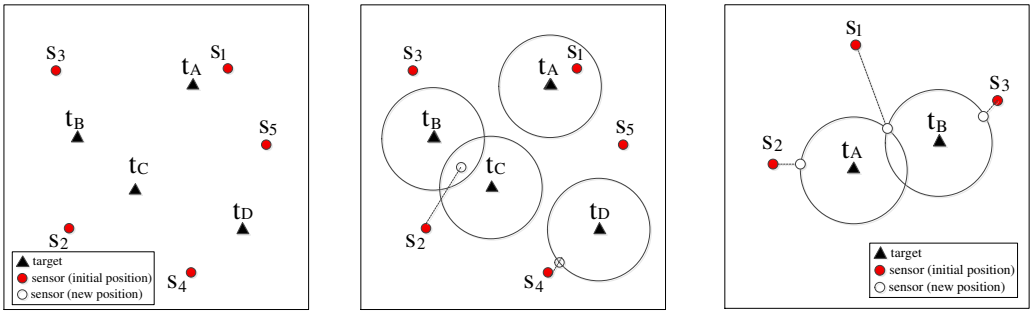
A simple heuristic to minimize the movement distance of sensors is to minimize *the number of sensors that need to move*. Actually, after the sensors are deployed, some targets may have already been covered. Denote the set of targets that have already been covered by $T_{initcov}$, and denote the set of uncovered targets by $T_{needcov}$. Then we have $T_{needcov} = T \setminus T_{inticov}$. In order to minimize the number of mobile sensors that need to move, we first construct a graph of targets representing whether targets can be simultaneously covered, then find the destinations of mobile sensors by using clique partition.

The graph is constructed as follows. For every target in $T_{needcov}$, there is a vertex in the graph. There is an edge between two vertices if and only if the corresponding targets could be simultaneously covered by the same sensor. After the graph is constructed, we find a minimum clique partition of the constructed graph. Each partitioned clique represents a subset of targets that can be covered by the same sensor. Thus, for targets belonging the same clique, we need to dispatch only one mobile sensor to cover them. With this method, the number of mobile sensors that need to move is minimized. After the clique partition is obtained, the extended Hungarian algorithm is used to determine which sensor should be dispatched to cover the targets in each clique.

The pseudo code of the Basic algorithm is presented in Algorithm 1. The algorithm first finds out the set of targets to be covered ($T_{needcov}$) and the set of mobile sensors to be moved (S_{rest}). It then finds a minimum clique partition on the graph of targets in $T_{needcov}$. For every clique and every sensor in S_{rest}, the potential destination and corresponding movement distance for the sensor to cover targets in that clique is computed. The extended Hungarian algorithm is then used to find which sensor should be move and move to which potential destination to cover all the targets in $T_{needcov}$.

Fig. 3 demonstrates the execution of the Basic algorithm. As shown in figure 3(a), there are four targets t_A, t_B, ,t_C, and t_D, and five mobile sensors s_1, s_2, s_3, s_4, s_5. Initially target t_A is covered by sensor s_1, thus $T_{needcov} = \{t_B, t_C, t_D\}$ and $S_{rest} = \{s_2, s_3, s_4, s_5\}$. After clique partition, targets in $T_{needcov}$ is divided into two groups: $\{t_B, t_C\}$ and $\{t_D\}$. According to the output of the extended-Hungarian algorithm, sensor s_2 is moved to cover both t_B and t_C, and sensor s_4 is moved to cover t_D. The result is shown in figure 3(b).

However, although the Basic algorithm minimizes the number of sensors to move, it may increase the total movement distance of sensors. For example, as shown in Fig. 3(c), target A and B could be covered by one single sensor. According to the Basic algorithm, s_1 should be moved to cover them because it is closest to the intersection of the coverage disks of A and B among the three sensors. However, if we move two s_2 and s_3 to cover t_A and t_B respectively, we can further reduce the total movement distance, although the number of moved sensors is not minimized. In the next section, we further propose a Voronoi diagram based algorithm to minimize the movement of sensors rather than the number of sensors to move.



(a) Initial positions (b) Result of Basic (c) Suboptimality of Basic

Fig. 3. Illustration of the Basic algorithm: (a) Initial positions of targets and sensors; (b) The results of the Basic algorithm, in which two sensors need to move; (c) Suboptimality of the Basic algorithm: moving least sensors may induce longer total movement distance.

4.2.2 The Target-Based Voronoi Greedy Algorithm

In this section, we present a target based Voronoi greedy algorithm (TV-Greedy) to minimize the total movement distance of sensors to cover targets.

Basic Idea and Definitions

The basic idea of TV-Greedy is to deploy the nearest sensor to cover the targets that are uncovered. Since sensors located in a target's Voronoi polygon are closer to this target than to others, we use Voronoi diagrams of targets to group sensors according to their proximity to the corresponding target.

For the sake of clarity, the definitions and notations that will be used in the algorithm description is presented below:

1) If a sensor is located in a target's Voronoi polygon, the sensor is defined as a *server* to this target, and the target is regarded as a *client* of its servers. The set of a target's servers is called that target's *own server group* (OSG). The sensor in a target's OSG that is nearest to the target is called the *chief server* of that target, and other sensors are called *non-chief servers* of the target.
2) Two targets are *neighbors* if their Voronoi polygons share an edge. For two neighboring targets A and B, the sensor in A's OSG that is closest to B is called an *aid server* to B.
3) A target's *candidate server group* (CSG) is the union of its own chief server and aid servers from neighbors. For a target, only sensors in its CSG will be dispatched to cover it.

For instance, as shown in Fig. 4, the own server group of target t_B is $OSG_B = \{s_4, s_5, s_6\}$, in which s_4 is the chief server. For other sensors in OSG_B, s_6 is the aid server for t_A, and s_5 is the aid server for t_C. Meanwhile, t_B has an aid server from t_C, which is s_2. Thus the candidate server group of t_B is $CSG_B = \{s_4, s_2\}$. Note that there is no sensor in target t_C's Voronoi polygon, and thus there is no aid server for t_B from t_C even though t_B and t_C are neighbors.

Algorithm Description

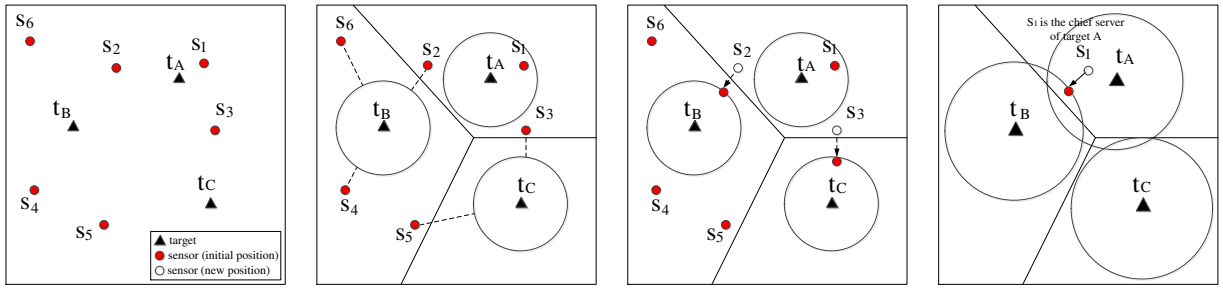
TV-Greedy starts from the generation of targets' Voronoi diagrams, which divides sensors into independent groups for each target. With assistance of targets' Voronoi diagrams, we can construct a sensor group for each target, which includes sensors in proximity to this target. Then, the nearest sensor to each target is selected from the target's group and its neighbors' group. After that, the selected sensor moves to the corresponding target. Details of the algorithm are described as follows, and its pseudo code is given in Algorithm 2.

First, the Voronoi diagram of targets is generated by using the coordinate information of targets which is known to sensors. Base on the vertices information of Voronoi polygons, the neighbors of each target are determined (Steps 1-2 in Algorithm 2).

Second, the own server group OSG of each target is determined. In each OSG, the own servers (sensors in the OSG) is sorted by their distances to the client (the target of the OSG) in ascending order, according to which the chief server is identified as the first in the sorted list. For the rest own servers, we identify the aid server for each neighbor of the client via distance comparison and sorting, as shown in Fig. 4(b) (Steps 3-6 in Algorithm 2).

Third, for each target, if it is covered initially, sensors in its OSG stand by and wait for orders (Steps 7-9 in Algorithm 2). If the target is not covered initially, then its CSG will be formed, which is a logical server group merged with the chief server of the target and all the aid servers from its neighbors.

Then, if the CSG of a target is not empty, the nearest sensor is selected from the CSG to move (shown in Fig. 4(c), Steps 10-15 in Algorithm 2). If the CSG is empty, it means that there is no sensor located in the target's Voronoi polygon. In this case, if there exit neighbors of the target that can share their chief server with the target, the nearest chief server moves to the nearest new position which is in the coverage disk of t_i' (shown in Fig. 4(d), Steps 16-18 in Algorithm 2); otherwise, the CSG of t_i is regenerated by searching aid server of the 2nd order neighbor of t_i's (i.e., neighbors of neighbors)



(a) Initial position of targets and sensors. (b) Voronoi graph of targets and groups of sensors. (c) Moving decisions are made. (d) Sharing a chief server.

Fig. 4. Illustration of the TV-Greedy algorithm.

Algorithm 2: The TV-Greedy Algorithm

Input: $T = t_1, t_2, ..., t_m$;//The position of all targets
$S = s_1, s_2, ..., s_n$;//The position of sensors
r_s;//The coverage radius
Output: tmc;//The total moving cost

```
1  Generate the Voronoi diagram (VD) of targets;
2  Determine neighbors for each target according to
   their Voronoi polygon;
3  Determine the OSG for each target according to S
   and VD;
4  for each OSG_i do
5      Determine the chief server ;
6      Identify the aid server for t_i's neighbor;
7  for each t_i do
8      if t_i has already been covered then
9          Return cost(t_i)=0;
10     else
11         Produces CSG_i of t_i;
12         if CSG_i ≠ ∅ then
13             Move the nearest server to cover t_i;
14             return cost(t_i)= moving distance;
15         else
16             if there exit neighbors' chief servers that
               could be shared then
17                 move the nearest chief server to cover
                   t_i;
18                 Return cost(t_i)= moving distance;
19             else
20                 Regenerate the CSG of t_i by searching
                   aid servers of the t_i's 2nd or higher
                   order neighbors;
21                 Move the nearest aid server to cover
                   t_i;
22                 Return cost(t_i)= moving distance ;
23     tmc=tmc+cost(t_i)
24 return tmc
```

or higher order neighbor. After that, the nearest aid server moves to the nearest new position which is in t_i's coverage disk (Steps 19-22 in Algorithm 2).

5 Solutions to the NCON Problem

The sensors that are used to cover targets in the TCOV problem are referred to as *coverage sensors*. After the TCOV problem is solved, all the targets are covered by at least one coverage sensor. Besides the coverage of targets in the first stage, another important requirement for a WSN is the connectivity of sensors and the sink, which promises the data transmission. If the sink and the coverage sensors are initially connected, then the connectivity problem is solved; otherwise, we need to study the NCON problem, i.e., how to connect the sink and the coverage sensors.

The basic idea of providing connectivity is to relocate the rest mobile sensors to some locations where they can connect coverage sensors and the sink. Consider a tree-topology, where the sink is the root and all the coverage sensors are the leaf nodes, the goal of NCON is to relocate mobile sensors to new positions as intermediate node to connect the sink and coverage sensors, and the movement of sensor is minimized. From the above analysis, the NCON problem can be solved in two steps.

- First, we construct an edge length constrained Steiner tree spanning all the coverage sensors and the sink, such that each tree edge length is no longer than r_c. The Steiner tree is required to minimize the number of sensors that need to move.
- Second, we relocate the rest mobile sensors to the generated Steiner points to connect the coverage sensors and the sink. As for the second step, it is actually the special case of TCOV in which the Steiner points are regarded as "target"s and the coverage radius is zero. Then for each target we need to dispatch a dedicated sensor to cover it.

The key point to solve the NCON problem is to solve the first step: seeking an edge length constrained Steiner tree T spanning coverage sensors and the sink. Since the Steiner tree problem is NP-hard, we propose an approximate algorithm as follows: (1) constructing an Euclidean minimum spanning tree, and (2) separating

Algorithm 3: The ECST Algorithm

Input: $S = s_1, s_2, ..., s_n$;//The set of mobile sensors
S_{cov};//The set of coverage sensors
$sink(x, y)$;//The location of the sink
r_c;//The communication radius
Output: SP;//The set of Steiner points

```
V = S_cov;
Construct a complete graph G = (V, E);
Construct an Euclidean minimum spanning tree T_ems of G with the sink as the root;
for each v_i ∈ V and its parent v_p^i do
    Separate the edge e(v_i, v_p^i) into ⌈‖e(v_i,v_p^i)‖ / r_c⌉ parts;
    SP(x_i, y_i) ← each separating point;
return SP;
```

Algorithm 4: The ECST-H Algorithm

Input: $S = s_1, s_2, ..., s_n$;//The set of mobile sensors
S_{cov};//The set of coverage sensors
$sink(x, y)$;//The location of the sink
r_c;//The communication radius

```
ECST (S, S_cov, sink(x, y), r_c);
extended-Hungarian (SP, S/S_cov); //Move the rest mobile sensors to the Steiner points
return movement cost and the deployment orders;
```

each edge of the spanning tree into the sections with length not larger than r_c. Because the sum of edge length in an Euclidean minimum spanning tree is minimum, the number of section points on all edges is minimum. The EuClidean minimum Spanning Tree (ECST) algorithm is listed in Algorithm 3.

With the output SP of the ECST algorithm, the next step is to assign the rest mobile sensors one-by-one to each point in SP with the minimum movement. Since it is actually an assignment problem, it can be solved using the extended Hungarian method described in Section 4.1. Algorithm 4 gives the ECST-Hungarian (ECST-H) algorithm to solve the NCON problem.

6 Theoretical Analyses

6.1 Complexity of Algorithms

This section investigates the complexity of the proposed algorithms to provide a theoretical guidance on whether the solutions are applicable in practice. Given m targets and n mobile sensors, we use the Merge sort as our sorting algorithm, which takes $O(n \log n)$ time and $O(n)$ space in the worst case o sort n data items [32].

6.1.1 Complexity of the Algorithms for TCOV

Recall that the number of fixed targets is m and the number of mobile sensors is n. In most scenarios, $m < n$. As for the extended-Hungarian method, a $n \times n$ cost matrix is constructed. Hence the time complexity of the extended-Hungarian method is $O(n^3)$ [31].

TABLE 1
Time Complexity of Different Algorithms.

Algorithm	Ex-Hungarian	Basic	TV-Greedy	ECST-H
Complexity	$O(n^3)$	$O(n^3)$	$O(n \log n)$	$O(n^3)$

The Basic algorithm consists of three parts. The first part is a nested loop from Step 2 to Step 8, whose complexity is $O(mn)$. The second part is the computation of potential positions of sensors and corresponding costs (GEOCP[1], Step 10 in Algorithm 1). This part focuses on the generation of target adjacent graph, mainly on sorting targets by their distances to the boundary of the area. The time complexity of GEOCP is $O(m^3)$ that was proven in our previous work [33]. The third part is the extended-Hungarian method used to dispatch sensors to partitioned cliques. As the number of cliques must be no larger than n, this part takes at most $O(n^3)$ time. Thus the total time complexity of the Basic algorithm is $O(m^3+n^3)$. When there are less targets than sensors, the total time complexity of the Basic algorithm is $O(n^3)$.

As for the TV-Greedy algorithm shown in Algorithm 2, the main operations are generating Voronoi diagrams of targets in Step 1 and sorting by distance to determine OSG and CSG in Steps 2-3. In Step 1, generating Voronoi diagrams of m points in a plane takes $O(m \log m)$ time [34]. Hence, the time and space complexity of TV-Greedy are actually determined by Steps 4-22, whose functions are grouping and sorting sensors by their distances to the corresponding target. In Steps 4-6, it require $O(n \log n)$ time in the worst case to sort sensors in each OSG and there are totally m OSGs. Thus the total time complexity of Steps 4-6 is $O(mn \log n)$. In Steps 7-22, the worst case happens in Steps 20-22. That is, when targets are distributed along the diagonal line of the area, each of the first $(m-1)$ target's Voronoi polygon has only one sensor and the rest $(n-m+1)$ sensors are in the m-th target's Voronoi polygon. In such a case, the algorithm starts at the first target to the mth one, and Steps 20-22 take $O(m^2(n-m)\log(n-m))$ time. In conclusion, the time complexity of TV-Greedy is $O(m^2 n \log n)$. If the number of targets is a constant, the time complexity of the TV-Greedy is $O(n \log n)$.

6.1.2 Complexity of the Algorithm for NCON

As for the ECST-H algorithm shown in Algorithm 4, computations consist of two parts: the ECST part and the extended-Hungarian part. For the ECST part presented in Algorithm 3, Step 1 runs in a constant time, Step 2 runs in $O(n^2)$ time to generate a complete graph of n vertices, Step 3 needs $O(n \log n)$ time [35] to find an Euclidean minimum spanning tree, and Steps 4-6 run in $O(n)$ time to sequentially traverse all the edges of generated spanning tree. The time complexity of the

1. The detailed implementation of GEOCP was given in [33] and is omitted in this paper due to space limitation.

ECST algorithm is thus $O(n^2)$. The time complexity of extended-Hungarian algorithm is $O(n^3)$. In conclusion, the total time complexity of ECST-H is $O(n^3)$.

As a summary, Table 1 lists the complexity of solutions for the TCOV problem and for the NCON problem. It can be concluded from Table 1 that, TV-Greedy has much lower time complexity than its counterparts. Hence, the combination of TV-Greedy and ECST-H would be a promising choice to solve the MSD problem in operation efficiency matters.

6.2 Communication Overhead

In the proposed algorithms, communications mainly consist of two parts: the collection of mobile sensors' initial positions (from sensors to the sink), and the dissemination of moving orders (from the sink to sensors), i.e., which sensors should move and the destinations they need to move to.

For the first part, there are two difference cases for a sensor to report its position to the sink: a) the sensor could communicate with the sink directly, and b) the sensor need to use some relay sensors to communicate with the sink. In the former case, the communication cost is $O(1)$. In the latter case, the communication cost is at most $O(n)$ as the number of relay nodes is at most n. Thus the communication cost in the first part is $O(n)$.

For the second part, there are also two difference cases as in the first part. Assume that the total number of sensors (both coverage sensors and Steiner sensors) that need to move is K. If the sensor can directly communicate with the sink, it takes $O(1)$ time for the sink to send the moving order to the sensor. Otherwise, it takes at most $O(n)$ time for the sink to send the moving order to the sensor. Thus the overall communication cost in the second part is at most $O(Kn)$.

Combining the communication overhead in the two parts, the total communication overhead of the proposed algorithms is $O(Kn)$, where $K \ll n$ in general cases.

7 Simulation Experiments

7.1 Simulation Settings and Performance Metrics

To evaluate the performance of the proposed algorithms, we conduct a set of simulations by using Matlab. We first investigate the performance of the three solutions to the TCOV problem, namely Ex-Hungarian, Basic, and TV-Greedy, then study how their combinations with the ECST-H algorithm perform in solving the MSD problem. In the simulations, the targets and mobile sensors are randomly generated in a 400m×400m area. The default coverage radius and communication radius are $r_s = 10$m and $r_c = 15$m, respectively. For each combination of network parameters, we randomly generate 20 instances of the network and report the mean performance result.

The primary concerned metric is the total movement distance of sensors. We consider two network parameters that may impact sensors' movement distance: the number of targets (m) and the number of mobile sensors (n). As the TV-Greedy algorithm performs the best among the three algorithms, we also investigate the performance gap between TV-Greedy and the optimal solution in a small network to get an impression of how close TV-Greedy approaches the optimal solution.

7.2 An Illustration of Sensors Deployment

To get an intuitive impression of how our algorithms work, we demonstrate the generated tree topologies of the three different solutions, namely Ex-Hungarian+ECST-H, Basic+ECST-H, and TV-Greedy+ECST-H, in Fig. 5. There are 20 targets and 150 sensors in the network. We have the following observations.

The first observation is that Basic uses the least number of coverage sensors and TV-Greedy uses the most number of coverage sensors, while Ex-Hungarian uses exactly the same number of coverage sensors as targets because it moves sensors to targets in a one-to-one manner. Different choices of the coverage sensors also impact the choice of Steiner sensors in the NCON problem. As shown in Table 2, different solutions use different number of Steiner sensors to provide connectivity. As for the total number of both coverage sensors and Steiner sensors, it is still that Basic+ECST-H uses the least and TV-Greedy+ECST-H uses the most.

The second observation is that TV-Greedy incurs much shorter movement distance than the other two solutions in the coverage stage. Although TV-Greedy uses more coverage sensors than Ex-Hungarian does (25 vs 19), its movement distance is much shorter (236.8m vs 424.3m). This owes to TV-Greedy's smart strategy in choosing coverage sensors. It groups sensors according to their proximity to the targets, and uses the nearest sensor to cover a target. This effectively reduces the movement distance to cover all the targets.

It could also be observed that the differences in the solutions to the TCOV problem affect the performance of ECST-H in solving the NCON problem, which consequently affect the overall performance of solutions to the MSD problem. ECST-H performs the best when Ex-Hungarian is used, slightly better than when TV-Greedy is used. However, the overall performance of TV-Greedy+ECST-H is best among the three combinations.

7.3 Performance of Different Algorithms to TCOV

7.3.1 The Impact of Mobile Sensor Number

We first study how the number of mobile sensors affects the performance of the three solutions to the TCOV problem when the number of targets is fixed. Two scenarios are considered. In the first scenario, targets are scattered sparsely that the distance between any two targets is greater than $2*r_s$. In this case, the Ex-Hungarian method can find the optimal solution and thus can be used as the benchmark to evaluate the performance of Basic and

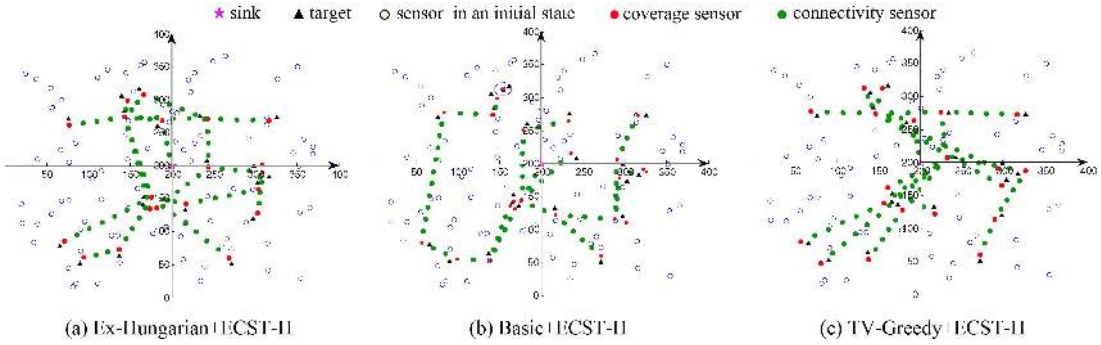


Fig. 5. Network topologies generated by different combinations of algorithms. There are 20 targets and xxx sensors in the network. Basic+ECST-H moves the least number of sensors, but TV-Greedy+ECST-H results in the shortest movement distance, as shown in Table 2.

TABLE 2
Performance data of different combinations of algorithms in Fig. 5.

Algorithm Combination	Coverage/Steiner sensors	TCOV/NCON movement(m)	Total sensors/movement(m)
Ex-Hungarian+ECST-H	20/74	424.3/**580.3**	94/1004.6
Basic+ECST-H	**19**/**57**	493.1/948.1	**82**/1440.2
TV-Greedy+ECST-H	25/83	**236.8**/654.4	102/**891.2**

TV-Greedy. In the second scenario, targets are scattered randomly and densely. This represents the general case of TCOV in which the distances between targets might be less than $2 * r_s$.

Fig. 6 (a) shows the performance of the three algorithms in the first scenario when n varies from 100 to 400. It shows that when there are more sensors, all the three algorithms incur less movement distance. The reason is obvious: With more sensors, each target can be covered by a closer sensor, which reduces the total movement distance. Ex-Hungarian performs best in this scenario, TV-Greedy follows, and Basic performs worst. Because in this case targets disperse from each other more than double of communication radius, Ex-Hungarian can find the optimal solution. Compared to the optimal solution found by Ex-Hungarian, TV-Greedy and Basic incur about 25% more distance and 36% more distance, respectively. Obviously, TV-Greedy performs better than Basic does.

The performance of the three algorithms in the general scenario is given in Fig. 6 (b). The trend is similar, i.e., all the three algorithms incur shorter movement distance with larger number of sensors. The difference from the first scenario is that TV-Greedy and Basic perform better than the Ex-Hungarian algorithm. This owes to the fact that in TV-Greedy and Basic different targets can be covered by the same sensor, while in Ex-Hungarian every target needs to be covered by distinct sensors. Thus TV-Greedy and Basic could use less sensors to achieve coverage which contributes to the reduction in movement distance. TV-Greedy performs nearly the same as Basic when the number of sensors is relatively small (e.g., $n \leq 100$), but outperforms Basic significantly when there are more sensors to dispatch (e.g., $n \geq 150$). Overall, TV-Greedy is superior to both Ex-Hungarian and Basic in reducing movement sensors in general case of TCOV.

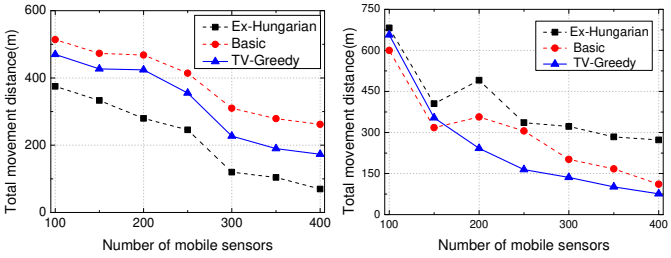


Fig. 6. Impact of the mobile sensor number (n) on the movement distances of the three algorithms when $m = 30$: (a) When the distances between targets are greater than $2 * r_s$; and (b) when targets are scattered randomly.

7.3.2 *The Impact of Target Number*

The impacts of the target number on the movement distance of the proposed algorithms are also studied in the same two scenarios. In both scenarios, as shown in Fig. 7, the movement distance increases when m increases. The reason is that when m increases, more targets need to be covered, which requires more sensors to be moved and consequently incurs longer movement distance. Fig. 7 (a) shows the movement distance of different algorithms in the first scenario, i.e., when targets are spacing greater than $2 * r_s$. In this scenario, Ex-Hungarian is optimal and performs best. TV-Greedy and Basic perform very close to each other and need about 35% more movement than EX-Hungarian. The results in the general scenario is shown in Fig. 7 (b). Again, TV-Greedy outperforms the other two algorithms, incurs about xx% less movement distance than Basic and yy% less movement distance than Ex-Hungarian.

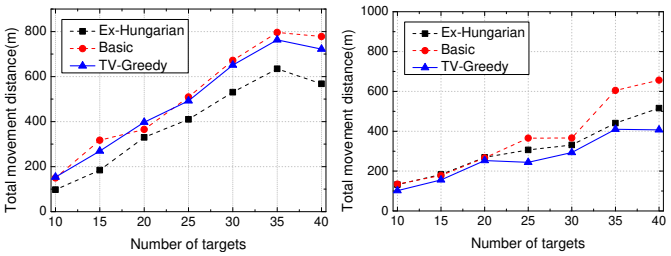


Fig. 7. Impact of the target number (m) on the movement distances of the three algorithms when $n = 300$: (a) When targets are spacing greater than $2 * r_s$; and (b) when targets are scattered randomly.

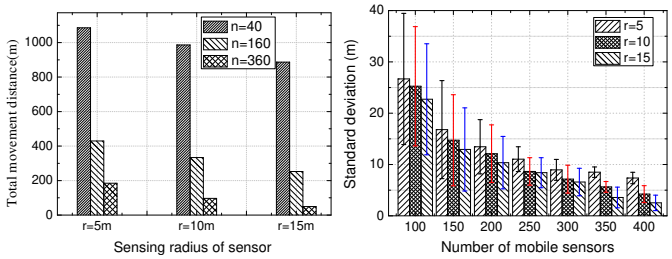


Fig. 8. The impact of network parameters on the performance of TV-Greedy: (a) Total movement distance when n and r_s changes; and (b) The mean and standard deviation of sensor's movement distance (only moved sensors are counted).

7.4 Further Investigation of TV-Greedy

7.4.1 Impact of Different Network Parameters

As TV-Greedy performs best among the three algorithms in the general case of TCOV, we take it as an example to further investigate how the network parameters, e.g., the sensing radius and number of sensors, influence the performance of our solutions to the TCOV problem.

Fig. 8 (a) plots the total movement distance of TV-Greedy for different combinations of n and r_s when the target number is fixed at 30. It can be observed that the total movement distance decreases along with the increase of both mobile sensors and sensing radius. However, the impact of increasing mobile sensors is more significantly than enlarging sensing radius. When r_s is fixed and n increases in the equivalent case, there is a nearly 60% decrease in the total movement distance. In contrast, when n is fixed and r_s increases in the equivalent case, there is only about 10% decrease in the total movement distance. This shows that sensors' density has much greater effects on the moving cost than the sensing radius. This can be explained as follows. When there are more mobile sensors, TV-Greedy has more choices in selecting coverage sensors for a target, and it will select the sensor closest to the target to cover that target. In other words, when there are more sensors TV-Greedy might select a different set of sensors as coverage sensors. In contrast, when mobile sensors are unchanged and only r_s increases, it is very possible that the same set of sensors are selected as coverage sensors, but with each sensor move shorter distance due to larger r_s. Thus increasing mobile sensors can reduce total movement distance more significantly than increasing sensing radius.

Fig. 8 (b) plots the mean and standard deviation of sensors' movement with different number of mobile sensors. Three different sensing radii are considered, namely $r_s = 5, 10, 15$. We can see that both the mean and standard deviation of sensors' movement distance decrease when there are more mobile sensors, which means that TV-Greedy achieves better balance of energy consumption among different nodes. Similar as for the total movement distance, a larger r_s also results in shorter mean movement distance. However, the impact of sensing radius r_s is lighter than the impact of mobile sensor numbers.

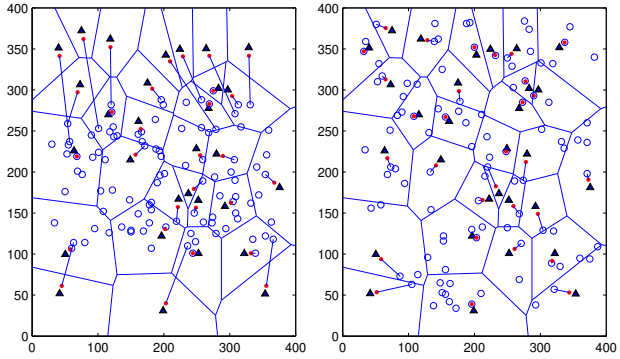


Fig. 9. The impact of sensors' initial positions on TV-Greedy's performance (targets and sensors are marked as triangles and open circles, respectively): (a) when sensors are initially deployed densely; and (b) when sensors are initially deployed randomly. Random deployment of sensors results in shorter total movement distance.

The initial deployment of sensors also impacts the performance of TV-Greedy. Fig. 9 shows TV-Greedy's output to cover the same set of targets when sensors are deployed with different initial positions. There are 30 targets and 150 mobile sensors. In Fig. 9 (a), sensors are initially deployed densely. In this case, 22 sensors need to move and the total movement distance is $725.8m$. In Fig. 9 (b), sensors are randomly scattered. In this case, 16 sensors need to move with total movement distance $237.2m$. In the network shown in Fig. 9 (a), sensors are apart from targets. In this case, most sensors have to move from one Voronoi polygon to another to cover the targets that are far way, which results more sensors to be moved and larger total movement distance. In contrast, sensors in Fig. 9 (b) are randomly scattered, which is similar to the distribution of targets. In this case, sensors just need move slightly to cover targets.

7.4.2 Performance Gap Between TV-Greedy and the Optimal Solution

We study how the solution provided by TV-Greedy approaches the optimal solution by using a small network. As shown in Fig. 10, the network consists of 5 targets

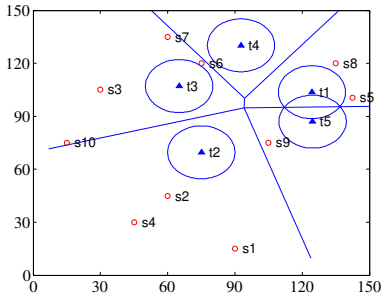


Fig. 10. The network used to study the performance gap between TV-Greedy and the optimal solution, in which there are 5 targets and 10 sensors.

and 10 mobile sensors that are randomly deployed in a 150m×150m area. The network is small so that the optimal solution could be found by using brute force searching, which should be $\{s_5 \rightarrow t_1 + t_5, s_2 \rightarrow t_2, s_7 \rightarrow t_3, s_6 \rightarrow t_4\}$. The total movement distance in the optimal solution is **31.5m**. The solution found by TV-Greedy is $\{s_8 \rightarrow t_1, s_2 \rightarrow t_2, s_6 \rightarrow t_3, s_7 \rightarrow t_4, s_9 \rightarrow t_5\}$, with total movement distance of **59.4m**. TV-Greedy incurs about 28m more distance than the optimal solution. As a comparison, the solution found by the extended Hungarian method is $\{s_6 \rightarrow t_1, s_s \rightarrow t_2, s_7 \rightarrow t_3, s_8 \rightarrow t_4, s_9 \rightarrow t_5\}$ with total movement distance of **81.2m**, which uses about 50m longer distance than the optimal solution to cover all the targets.

7.5 Performance of ECST-H

With the positions of coverage sensors generated by the above three algorithms, the next is to relocate the rest mobile sensors to connect coverage sensors and the sink to provide a complete solution to the original MSD problem. In this part, we test the performance of ECST-H based on the positions of coverage sensors generated by EX-Hungarian, Basic and TV-Greedy, respectively. The sink is placed in the center of the area, as shown in Fig. 5.

Fig. 11 (a) gives the movement distance of ECST-H when $m = 30$ and n varies from 200 to 400 with increment 50. In Fig. 11 (a), ECST-H(E) represents the result of ECST-H when the Ex-Hungarian method is used to obtain coverage sensors in the first stage. Similarly, ECST-H(B) and ECST-H(T) represent the result of ECST-H when Basic and TV-Greedy are used in the first stage, respectively. As indicated in the figure, the larger number of sensors, the less movement of sensors for connectivity. ECST-H(T) outperforms both ECST-H(E) and ECST-H(B), especially when the number of sensors exceeds 150. But with the increase of sensor number, all of the three curves drop, which implies that movement distance of sensors for connectivity would become smaller when the sensor density rises.

Fig. 11 (b) shows the movement of sensors in ECST-H when $n = 300$ and m varies from 10 to 40 with increment 5. It can be observed that when the number of targets increases, movement of sensors grows. The reason is

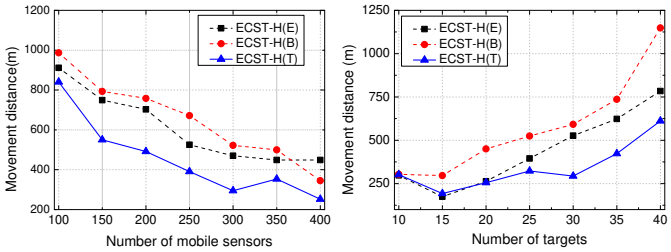


Fig. 11. Movement distance in ECST-H with different parameters: (a) m is fixed at 30, n varies; and (b) n is fixed at 100, m varies.

that more targets generally need more coverage sensors to cover them. This results in a steiner tree in ECST-H with more leaves and branches. Therefore, the increase of Steiner points leads to more sensors' movement to connect coverage sensors and the sink. Fig. 11 (b) indicates that, ECST-H(T) requires the minimum movement, and increases gently compared to its counterparts. ECST-H(B) soars when targets increase. This is because the more targets, the more cliques partitioned in Basic (including isolated targets), which leads to more coverage sensors that need to be connected.

8 Discussions and Future Works

8.1 When Free Mobility Model Is Not Applicable

Although we build our work on the free mobility model in this paper, the proposed solutions could be extended to be applicable to more generalized cases when the free mobility model is not applicable. We consider two cases in which the mobile sensor cannot move freely: (a) there are obstacles in the task area, and (b) the mobile sensors are limited in moving directions or/and step length.

For the first case, the shortest path between the mobile sensor and the target might not be the straight line connecting them. However, the shortest path can be identified according to the position of obstacles using the algorithms like Ant Colony Optimization Algorithms [36]. After the shortest path is obtained, each sensor knows how long it needs to move in order to cover a target. The proposed solutions can be then applied.

For the second case, consider an example in which the mobile sensors could move in four directions only (e.g., north, south, west, and east) and the length of each step is limited to be d meters. We can divide the task area into grids of size dm×dm. The mobile sensors can move along the grid edges only. In this model, the shortest path between the mobile sensor and the target, in the form of a series of segments, can be calculated. As a result, our solutions can also be applied after every mobile sensor obtains the distance to be moved to cover a target.

8.2 Future Works

In the future, we plan to extend our work into distributed algorithms. A distributed solution to the MSD problem is very attractive because it takes advantage

of robustness when facing network changes and sensor failures. The main challenge is that, in the distributed manner, mobile sensors can communicate only with sensors in proximity. Similarly, the moving decisions need to be made locally. However, localized algorithms face the potentially complicated relationship between local behavior and global behavior. Algorithms that are locally optimal may not perform well in a global sense. It remains our future work to design a distributed variant of our algorithms to solve the MSD problem.

9 Conclusion

In this work, we studied the Mobile Sensor Deployment (MSD) Problem, aiming at deploying mobile sensors to provide target coverage and network connectivity with minimum sensors' movement, in mobile sensor networks. We divided this problem into two sub-problems, Target COVerage (TCOV) problem and Network CONnectivity (NCON) problem. For the TCOV problem, we prove it is NP-hard. For a special case of TCOV, an extended Hungarian method is provided to get an optimal solution; for general cases, two heuristic algorithms are proposed based on clique partition and Voronoi diagram, respectively. For the NCON problem, we first propose an edge constrained Steiner tree algorithm to find destinations of mobile sensors, then use the extended Hungarian to dispatch rest sensors to connect the network. Theoretical analysis and simulation results show that, compared with extended Hungarian algorithm and Basic algorithm, solutions based on TV-Greedy have low complexity and are very close to the optimum.

Acknowledgement

This work is supported in part by the National Science Foundation of China (Grant No.61073036, 61103203), the Fund for Creative Research Groups of China (Grant No.70921001), the Program for Changjiang Scholars and Innovative Research Team in University of China under (Grant No.IRT0661).

References

[1] Renjie Huang, Wen-Zhan Song, Mingsen Xu, Nina Peterson, Behrooz Shirazi, and Richard LaHusen. Real-world sensor network for long-term volcano monitoring: Design and findings. *IEEE Transactions on Parallel and Distributed Systems*, 23(2):321–329, 2012.

[2] Gaddi Blumrosen, Bracha Hod, Tal Anker, Danny Dolev, and Boris Rubinsky. Enhancing rssi-based tracking accuracy in wireless sensor networks. *ACM Transactions on Sensor Networks*, 9(3):1–29, 2013.

[3] Mingming Lu, Jie Wu, Mihaela Cardei, and Minglu Li. Energy-efficient connected coverage of discrete targets in wireless sensor networks. *Networking and Mobile Computing*, pages 43–52, 2005.

[4] Benyuan Liu, Olivier Dousse, Philippe Nain, and Don Towsley. Dynamic coverage of mobile sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 24(2):301–311, 2013.

[5] Jren-Chit Chin, Yu Dong, Wing-KaiHon, and David K. Y. Yau. On intelligent mobile target detection in a mobile sensor network. In *Proc. of the 4th IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS)*, pages 1–9, 2007.

[6] Rui Tan, Guoliang Xing, Jianping Wang, and Hing Cheung So. Exploiting reactive mobility for collaborative target detection in wireless sensor networks. *IEEE Transactions on Mobile Computing*, pages 317–332, 2010.

[7] Zhixin Fu and Keyou You. Optimal mobile sensor scheduling for a guaranteed coverage ratio in hybrid wireless sensor networks. *International Journal of Distributed Sensor Networks*, 2013, 2013.

[8] Ren C. Luo and Ogst Chen. Mobile sensor node deployment and asynchronous power management for wireless sensor networks. *IEEE Transactions on Industrial Electronics*, 59(5):2377–2385, May 2012.

[9] Arun A. Somasundara, Aditya Ramamoorthy, and Mani B. Srivastava. Mobile element scheduling with dynamic deadlines. *IEEE Transactions on Mobile Computing*, 6(4):395–410, 2007.

[10] Tai lin Chin, Parameswaran Ramanathan, Kewal K. Saluja, and Kuang ching Wang. Exposure for collaborative detection using mobile sensor networks. In *Proc. of the 2rd IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS)*, pages 743–750, 2005.

[11] Nabhendra Bisnik, Alhussein A.Abouzeid, and Volkan Isler. Stochastic event capture using mobile sensors subject to a quality metric. *IEEE Transactions on Robotics*, 23(4):676–692, 2007.

[12] Liang He, Jianping Pan, and Jingdong Xu. A progressive approach to reducing data collection latency in wireless sensor networks with mobile elements. *IEEE Transactions on Mobile Computing*, 12(7):1308–1320, 2013.

[13] Xiao-Yang Liu, Kai-Liang Wu, Yanmin Zhu, Linghe Kong, and Min-You Wu. Mobility increases the surface coverage of distributed sensor networks. *Computer Networks*, 57(11):2348–2363, 2013.

[14] Ji Luo, Dan Wang, and Qian Zhang. Double mobility: coverage of the sea surface with mobile sensor networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 13(1):52–55, 2009.

[15] Shibo He, Jiming Chen, Xu Li, Xuemin Shen, and Youxian Sun. Cost-effective barrier coverage by mobile sensor networks. In *Proc. of the 31st Annual IEEE International Conference on Computer Communications (Infocom)*, pages 819–827. IEEE, 2012.

[16] Guiling Wang, Mary Jane Irwin, Piotr Berman, Haoying Fu, and Tom La Porta. Optimizing sensor movement planning for energy efficiency. In *Proc. of the 2005 international symposium on Low power electronics and design (ISLPED)*, pages 215–220, 2005.

[17] Hamid Mahboubi, Kaveh Moezzi, Amir G. Aghdam, and Kamran Sayrafian-Pour. Self-deployment algorithms for field coverage in a network of nonidentical mobile sensors. In *Proc. of IEEE International Conference on Communications (ICC)*, pages 1 – 6, 2011.

[18] Hamid Mahboubi, Kaveh Moezzi, Amir G Aghdam, and Kamran Sayrafian-Pour. Distributed deployment algorithms for efficient coverage in a network of mobile sensors with nonidentical sensing capabilities. *Accepted to appear in IEEE Transactions on Vehicular Technology*, PP(99):1–1, 2014.

[19] Yinying Yang, Mirela I. Fonoage, and Mihaela Cardei. Improving network lifetime with mobile wireless sensor networks. *Computer Communications*, 33(4):409 – 419, 2010.

[20] You-Chiun Wang and Yu-Chee Tseng. Distributed deployment schemes for mobile wireless sensor networks to ensure multilevel coverage. *IEEE Transactions on Parallel and Distributed Systems*, 19(9):1280–1294, 2008.

[21] Guiling Wang, Guohong Cao, T. La Porta, and Wensheng Zhang. Sensor relocation in mobile sensor networks. In *Proc. of the 24th Annual IEEE International Conference on Computer Communications (Infocom)*, volume 4, pages 2302–2312, 2005.

[22] You-Chiun Wang, Wen-Chih Peng, in Hsien Chang, and Yu-Chee Tseng. Exploring load-balance to dispatch mobile sensors in wireless sensor networks. In *Proc. of of 16th International Conference on Computer Communications and Networks (ICCCN)*, pages 669–674, 2007.

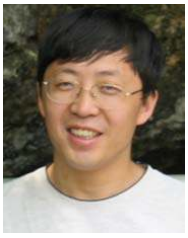
[23] Emi Mathews and Ciby Mathew. Deployment of mobile routers ensuring coverage and connectivity. *International Journal of Computer Networks & Communications*, 4(1), 2012.

[24] Shu Zhou, Min-You Wu, and Wei Shu. Finding optimal placements for mobile sensors: wireless sensor network topology adjustment. In *Proc.of the IEEE 6th Circuits and Systems Symposium on Emerging Technologies: Frontiers of Mobile and Wireless Communication*, volume 2, pages 529–532, 2004.

[25] Ines EI Korbi and Sherali Zeadally. Energy-aware sensor node relocation in mobile sensor networks. *Ad Hoc Networks*, 16(1):247 – 265, 2014.

[26] Xiaole Bai, Santosh Kumar, Dong Xuan, Ziqiu Yun, and Ten H. Lai. Deploying wireless sensors to achieve both coverage and connectivity. In *Proc. of the 7th ACM international symposium on Mobile ad hoc networking and computing (Mobihoc)*, pages 131–142. ACM, 2006.
[27] M.L.J. van de Vel. *Theory of convex structures*, volume 50. North Holland, 1993.
[28] Richard M. Karp. *Reducibility among combinatorial problems*. Springer, 1972.
[29] Eugene L. Lawler. The quadratic assignment problem. *Management science*, pages 586–599, 1963.
[30] Zhuofan Liao, Shigeng Zhang, Jiannong Cao, Weiping Wang, and Jianxin Wang. Minimizing movement for target coverage in mobile sensor networks. In *Proc. of 32nd International Conference on Distributed Computing Systems Workshops (ICDCSW)*, pages 194–200. IEEE, 2012.
[31] Harold W. Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.
[32] Richard Cole. Parallel merge sort. *SIAM Journal on Computing*, 17(4):770–785, 1988.
[33] Zhuofan Liao, Jianxin Wang, Shigeng Zhang, and Jiannong Cao. Clique partition based relay placement in wimax mesh networks. In *Proc. of IEEE Global Communications Conference (Globecom)*, pages 2566–2571, 2012.
[34] Mark de Berg, Marc van Kreveld, Mark Overmars, and Otfried Cheong Schwarzkopf. *Computational geometry*. Springer, 2000.
[35] Michael Ian Shamos. Geometric complexity. In *Proc. of the 7th Annual ACM symposium on Theory of computing (STOC)*, pages 224–233. ACM, 1975.
[36] Marco Dorigo and Mauro Birattari. Ant colony optimization. In Claude Sammut and Geoffrey I. Webb, editors, *Encyclopedia of Machine Learning*, pages 36–39. Springer, 2010.

Zhuofan Liao received the B.Sc. degree in Mathematics and Applied Mathematics from Hunan Normal University, China, in 2003, and Ph.D degree in Computer Science from Central South University, China, in 2012. She is currently an Assistant Professor in the School of Computer and Communication Engineering at Changsha University of Science and Technology, China. Her research interests include deployment and controlled mobility in wireless sensor networks and relay deployment in mobile Internet.



Jianxin Wang received his B.S. and M.S. degree in Computer Science from Central South University of Technology, P. R. China, and his PhD degrees in Computer Science from Central South University. Currently, Jianxin Wang He is the vice dean and a professor in School of Information Science and Engineering, Central South University, Changsha, Hunan, P.R. China. Jianxin Wang is currently serving as executive editor of International Journal of Bioinformatics Research and Applications and serving in the editorial board of International Journal of Data Mining and Bioinformatics. He has also served as a program committee member for many international conferences. He was a program committee co-chair for the 7th and 8th International Symposium on Bioinformatics Research and Applications (ISBRA 2011 and ISBRA2012), will be a program committee co-chair for the 8th International Frontiers of Algorithmics Workshop (FAW2014) and 10th International Symposium on Bioinformatics Research and Applications (ISBRA2014). His current research interests include algorithm analysis and optimization, parameterized algorithm, bioinformatics and computer network. He has published more than 200 papers in various International journals and refereed conferences. He is a senior member of the IEEE.

Shigeng Zhang received the BSc, MSc, and DEng degrees, all in Computer Science, from Nanjing University, China, in 2004, 2007, and 2010, respectively. He is currently an Assistant Professor in School of Information Science and Engineering at Central South University, China. His research interests include Cloud Computing, Internet of Things, wireless sensor networks, and RFID systems. He is a member of IEEE and CCF.

Jiannong Cao received the BSc degree in Computer Science from Nanjing University, China, in 1982, and the MSc and PhD degrees in Computer Science from Washington State University, Pullman, Washington, in 1986 and 1990, respectively. He is currently a chair professor in the Department of Computing at Hong Kong Polytechnic University, Hung Hom. He is also the director of the Internet and Mobile Computing Lab in the department. Before joining Hong Kong Polytechnic University, he was a faculty member in the Department of Computer Science at James Cook University and University of Adelaide in Australia, and City University of Hong Kong. His research interests include parallel and distributed computing, networking, mobile and wireless computing, fault tolerance, and distributed software architecture. He has published more than 300 technical papers in the above areas. His recent research has focused on mobile and pervasive computing systems, developing testbed, protocols, middleware and applications. He has served as a member of editorial boards of several international journals, a reviewer for international journals/conference proceedings, and also as an organizing/program committee member for many international conferences. He is a senior member of the IEEE, including Computer Society and the IEEE Communication Society, a senior member of the China Computer Federation, and a member of the ACM. He is also a member of the IEEE Technical Committee on Distributed Processing, IEEE Technical Committee on Parallel Processing, IEEE Technical Committee on Fault Tolerant Computing.

Geyong Min received the BSc degree in computer science from the Huazhong University of Science and Technology, China, in 1995 and the PhD degree in computing science from the University of Glasgow, United Kingdom, in 2003. He is a professor of computer science in the Department of Computing, University of Bradford, United Kingdom. His research interests include next-generation Internet, wireless communications, multimedia systems, information security, ubiquitous computing, modeling and performance engineering. His recent research was supported by UK EPSRC, Royal Society, Nuffield Foundation, and European FP. He has published more than 200 research papers in prestigious international journals, including IEEE Transactions on Computers, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Communications, IEEE Transactions on Wireless Communications, and IEEE Transactions on Multimedia, and in reputable international conferences, such as ICDCS, IPDPS, GLOBECOM, and ICC. He is an editorial board member of nine international journals. He served as the guest editor for 17 international journals and was the chair or vice chair of 30 international conferences/workshops. He received the Outstanding Leadership Awards from IEEE International Conferences CIT'2010/ScalCom'2010/ICESS'2010, ScalCom'2009, HPCC'2008.