A PTAS to Minimize Mobile Sensor Movement for Target Coverage Problem

Zhiyin Chen, Xiaofeng Gao[§], Fan Wu and Guihai Chen
Shanghai Key Laboratory of Scalable Computing and Systems
Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, 200240, China
Email: chenzhiyin@sjtu.edu.cn, gao-xf@cs.sjtu.edu.cn, fwu@cs.sjtu.edu.cn, gchen@cs.sjtu.edu.cn

***Abstract*—Energy consumption is a fundamental and critical issue in wireless sensor networks. Mobile sensors consume much more energy during the movement than that during the communication or sensing process. Thus how to schedule mobile sensors and minimize their moving distance has great significance to researchers. In this paper, we study the target coverage problem in mobile sensor networks. Our goal is to minimize the moving distance of sensors to cover all targets in the surveillance region. Here initially all the sensors are located at k base stations. Thus we define this problem as *k-Sink Minimum Movement Target Coverage*. To solve this problem, we propose a PTAS, named *Energy Effective Movement Algorithm* (EEMA). We can divide EEMA into two phases. In the first phase, we partition the surveillance region into some subareas. In the second phase, we select subareas and schedule sensors to the selected subareas. We also prove that the approximation ratio of EEMA is $1+\varepsilon$ and the time complexity is $n^{O(1/\varepsilon^2)}$. Finally, we conduct experiments to validate the efficiency and effectiveness of EEMA.**

I. INTRODUCTION

Wireless sensor networks have been widely applied in different fields, such as military applications, environmental applications and industrial applications [1]. Since the battery energy of sensors is usually limited and has a great impact on the lifetime of wireless sensor networks, one of the most fundamental and critical issue in wireless sensor networks is how to develop an energy efficient sensor schedule to satisfy some coverage requirements.

Many previous works save energy by scheduling the sensor state. When a sensor is idle, it will turn to sleep mode. It will be active again if needed. However, with the development of micro-electro-mechanical technologies, mobile sensors become more and more popular. A mobile sensor could detect the surveillance region periodically when moving along a predefined trajectory, which greatly reduces the number of sensors needed to monitor a region of interest and thus becomes an economical method for coverage requirements. Since for each sensor, the energy consumption during the movement is much more higher than that during the communication and sensing process, we need to develop efficient sensor scheduling strategies to save energy and extend network lifetime.

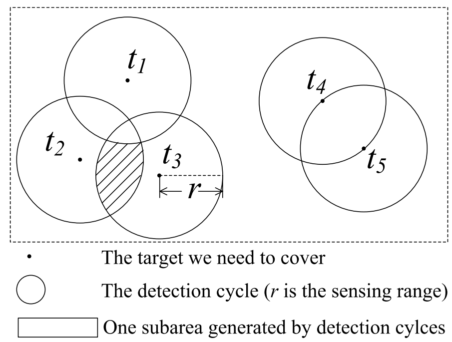


Fig. 1. Illustration of subareas

Several previous works studied the movement scheduling problem in mobile sensor networks. Ammari et al. [2] proposed a heuristic algorithm for k-coverage. By selecting and placing sensors appropriately, they tried to minimize sensor movement length and thus save energy. Wang et. al. [3] also studied this problem when k is not fixed. Liu et al. [4] and He et.al. [5] studied the movement scheduling problem for barrier coverage. Some other works [6], [7] discussed this problem in hybrid networks, which have both mobile sensors and static sensors. However, few literatures studied the problem for target coverage problem. Zorbas et al. [8] tried to prolong the network lifetime under probabilistic coverage model. Mini et al. [9] proved that this problem is NP-hard. Most of them only developed heuristic algorithms, without any performance guarantee and theoretical analysis.

In this paper, we try to minimize sensor movement for target coverage problem. Our goal is to minimize the moving distance of sensors to cover all targets in the surveillance region. Here initially all the sensors are located at k base stations and have the same sensing range. Thus we define this problem as *k-Sink Minimum Movement Target Coverage* (k-MMTC). We propose a *polynomial-time approximation scheme* (PTAS), named EEMA, to solve k-MMTC. Its approximation ratio is $1+\varepsilon$ where ε is a value that we set in EEMA.

EEMA can be divided into two phases. In the first phase, we partition the surveillance region into several subareas according to the detection cycle of targets. As shown in Fig. 1, t_i is a target and r is the sensing range of mobile sensors.

This work has been supported in part by the China 973 project (2012CB316200), National Natural Science Foundation of China (Grant number 61202024, 61472252, 61133006, 61272443, 61422208), the Opening Project of Key Lab of Information Network Security of Ministry of Public Security (The Third Research Institute of Ministry of Public Security) Grant number C15602, and the Opening Project of Baidu (Grant number 181515P005267).

[§]X.Gao is the corresponding author.

The detection cycle of t_i means if a sensor locates inside this cycle, then t_i can be successfully detected. The detection cycles for each target divide the surveillance region into many subareas, one of which is shown as the shadow area Fig. 1. To partition the objective region successfully, firstly we calculate the intersection points of different detection cycles and get a graph. Secondly we find all paths in the generated graph based on depth-first search (DFS). Finally, we can get all subareas. In the second phase, we calculate the weight of each subarea. The weight represents the minimum distance when a sensor move from a station to the subarea. Then we just need to select enough subareas with minimum weight to cover the target set.

To summarize, our contributions in this paper are as follows:

1) We define the problem *k-sink minimum movement target coverage* (k-MMTC) in mobile sensor networks and propose a novel polynomial-time approximation scheme EEMA.
2) We analyze the performance of EEMA and prove it approximation ratio of $1 + \varepsilon$. We also prove that the time complexity of EEMA is $n^{O(1/\varepsilon^2)}$ where n is the number of targets and ε is a predefined parameter.
3) We compare EEMA with the optimal solution and the algorithm in [9] to validate its effectiveness and efficiency.

In all, we are the first to design a PTAS for k-sink minimum movement target coverage in mobile sensor networks.

The rest of this paper is organized as follows: Section II discusses the related works. Section III describes the preliminary, basic model, problem assumption, problem statement and some definitions. In Section IV, we present the detailed design of EEMA. Furthermore we demonstrate the complexity and approximation ratio of EEMA. In Section V, we compare EEMA with other solutions and evaluate the performance of EEMA. Section VI summaries our work and states the future research direction.

II. Related Works

In majority scenarios, to achieve a best coverage in wireless sensor network is NP-hard. Thus we can hardly find the optimal solution for coverage problem in polynomial time. Thus, previous works usually designed heuristic algorithms or approximations to solve coverage problem.

Compared with approximation algorithms, heuristic algorithms are more popular in previous works. [10], [11] adopted simulated annealing algorithms. [10] applied simulated annealing to design periodical mobile coverage schedules. The scheduling objective is to distribute the coverage time of a mobile sensor in proportion according to the importance levels of targets when mobile sensors move around the targets. [11] also utilized simulated annealing for a coverage restoration scheme, which can find the best neighboring sensor to replace the failed sensor and extend the network lifetime. [12]–[16] adopted genetic algorithms. For area coverage maximization and energy conservation, [12] determined the speed and direction of each mobile node based on genetic algorithm. [13] proposed a new network coverage and optimization control strategy to solve 3D coverage problem in wireless sensor networks. With mobile sensors, [14] proposed a coverage holes healing algorithm. [15] proposed a sensors deployment strategy. The strategy can meet desired coverage requirement and maintain connectivity. [16] proposed an exact approach to maximize the network lifetime under a coverage ratio constraint and maximize the coverage ratio under a lifetime constraint. [16] also considered the bandwidth constraint.

Particle swarm optimization, which is applied in [17]–[19], is also a popular method. [17] combined particle swarm optimization with virtual force algorithm and proposed a dynamic sensor deployment algorithm. [18] applied particle swarm optimization to increase the coverage ratio. [19] proposed an adaptive approach based on particle swarm optimization which could achieve a good coverage solution with enhanced time efficiency. [20], [21] applied learning automata for coverage problem. In [20], a scheduling algorithm based on learning automata can schedule the sensors to detect the moving targets. [21] proposed three scheduling algorithm based on learning automata and organized sensors into several cover sets to extend the lifetime of wireless sensor networks. Artificial bee colony algorithm and artificial fish-swarm algorithm are also used for the coverage problem in wireless sensor network, such as [9], [22], [23].

However, unlike approximation algorithms, heuristic algorithms cannot give any performance guarantee. Sometimes, heuristic algorithms may have a bad performance and determining the parameters of heuristic algorithms is usually difficult. Some papers designed approximation algorithms for coverage problem in wireless sensor networks. For target coverage problem, the best approximation ratio is $4 + \xi$ [24]. [25] proposed an approximation algorithm for area coverage problem. Their algorithm could maximize the spatial-temporal coverage by scheduling the sensors activity. Its approximation ratio is 0.5. [26] studied k-coverage problem and proposed a 3-approximation algorithm. [27] designed an $O(\rho)$-approximation algorithm, where ρ is the density of sensors.

Some works designed approximation algorithms for variations of coverage problems. [28], [29] studied sweep coverage problem and proposed a 2-approximation. [30] studied camera coverage problem and gave a 2-approximation to minimize the number of cameras and maintain the connectivity of network. Though approximation has performance guarantee, it is usually difficult to design constant-factor approximation and analyze the approximation ratio. Thus some coverage problems still have no good approximation.

Nowadays, with the development of micro-electro-mechanical technologies, coverage problems with mobile sensors become more and more important. [31] gave a survey of movement strategies, including healing coverage hole, optimizing area coverage, and improving event coverage. [32] focused on minimizing movement of mobile sensors for target coverage, which was called MMTC. They reduced set cover to a special case of MMTC, and proved its NP-completeness. However, the best approximation ratio for set cover problem is $\ln n$. In this paper, we generalize the special case of MMTC and propose a new problem, k-sink minimum movement target coverage (k-MMTC) problem. Then we design a PTAS to solve the new problem for target coverage. It is the best approximation design for these series of coverage problems.

III. Preliminary and Problem Statement

In this section, we give the network model and formulate the problem that we study in this paper. Then we prove the NP-hardness of our problem for which we can hardly develop a polynomial time optimal algorithm for the problem.

A. Network model

We study the problem in homogeneous network. We model the network as $G = (T, P, S, r)$.

T represents the targets set which we need to cover. Assume that there are n targets $T = \{t_1, t_2, \cdots, t_n\}$ in the surveillance region. The surveillance region is flat and has no obstacle against sensor movement. The targets distribute uniformly and randomly in the surveillance region. Each target is static with a known location. In some papers, the target is also refer to as Point of Interest (POI).

P represents the stations set where the mobile sensors locate. Assume that there are k stations $P = \{p_1, p_2, \cdots, p_k\}$ around the surveillance region. All mobile sensors must start from a sink station in P. At the beginning, enough mobile sensors are static at the sink stations. From each sink station, we can send arbitrary number of mobile sensors to cover the targets in T.

S represents the mobile sensors set. Assume we use m sensors, $S = \{s_1, s_2, \cdots, s_m\}$, to cover all of targets in T. All sensors can move continuously in any direction, stop anywhere, and have the same sensing range.

r represents the sensing range of mobile sensors in S. Disk model is adopted for coverage. Namely, the target t_i is covered by the sensor s_j, if the distance between t_i and s_j is less than r. In addition, the distance between different targets may be less than r.

B. Problem Definition

As now well known, limited energy in sensors is a fundamental issue in wireless network and energy consumption for sensor movement is much more than that for sensing and communication. Thus we define the problem to minimize movement distance. Before we propose the problem, we give some definitions.

Definition 1. *(Movement Distance): When we schedule a mobile sensor s_i to cover a target t_j, s_i will move from a station p_k to somewhere near t_j. The movement distance of s_i, denoted as $d(s_i)$, equal to the linear distance between p_k and the destination.*

In order to minimize the sensor movement and reduce energy consumption, we define *k-Sink Minimum Movement Target Coverage* (k-MMTC) problem as follows:

Definition 2. *(k-Sink Minimum Movement Target Coverage Problem): We have k sink stations to send mobile sensors and cover all targets in T. k-MMTC is to schedule the sensor movement and minimize the sum of movement distance, denoted as d_{sum}. $d_{sum} = \sum_{i=1}^{m} d(s_i)$.*

The previous work in [32] has proved that 1-MMTC problem can be reduced to set cover problem which is NP-complete. Thus k-MMTC problem is also NP-complete. In this paper, we design a PTAS for the k-MMTC problem.

IV. Energy Effective Movement Algorithm

In this section, we proposed a PTAS, named as Energy Effective Movement Algorithm (EEMA), to solve the k-MMTC problem.

A. Overview

We divide EEMA into two phases. In the first phase, we propose a novel method to divide the surveillance region into some subareas according to the locations of targets. The sensors in the same subarea can cover the same targets set. In the second phase, we schedule the mobile sensors and move the sensors to cover all targets. Finally, we also analyze the time complexity and the approximation ratio of EEMA.

B. The First Phase

Before we introduce the detail of EEMA in the first phase, we give some definitions.

Definition 3. *(Detection-cycle): We can get a cycle, denoted as $\odot t_i$, from a target t_i. t_i is the center of $\odot t_i$. The sensing range, r, is the radius of $\odot t_i$. We define $\odot t_i$ as the detection-cycle of t_i.*

Apparently, a sensor s_j can detect t_i as long as s_j in $\odot t_i$.

Definition 4. *(Key-point): When two targets, t_i and t_j, are close enough, $\odot t_i$ and $\odot t_j$ may intersect. The intersection points on $\odot t_i$ are called as the key-points of t_i and denoted as $KP(t_i)$, where $KP(t_i) = \{kp_1(t_i), kp_2(t_i), \cdots\}$.*

For a key-point $kp_j(t_i)$, we can record the location of $kp_j(t_i)$ and the two targets which intersect and generate $kp_j(t_i)$. We find that the $KP(t_i)$ divide $\odot t_i$ into some arcs, $\{kp_1(t_i)\overset{\frown}{)}kp_2(t_i), kp_2(t_i)\overset{\frown}{)}kp_3(t_i), \cdots\}$. Those arcs divide the surveillance region and generate some subareas. We define the arcs as follows:

Definition 5. *(Curved-boundary): We can sort $KP(t_i)$ in anti-clockwise order, and get a sequence of key-points $\{kp_1(t_i) \to kp_2(t_i) \to \cdots \to kp_1(t_i)\}$. The neighboring key-points can generate an arc $kp_j(t_i)\overset{\frown}{)}kp_{j+1}(t_i)$, which is defined as curved-boundary.*

Definition 6. *(Covered-set of curved-boundary): The sensors on the same curved-boundary, $kp_j(t_i)\overset{\frown}{)}kp_{j+1}(t_i)$, can cover the same targets set. We define this targets set as covered-set of the curved-boundary $kp_j(t_i)\overset{\frown}{)}kp_{j+1}(t_i)$. We denote this covered-set as $T(t_i, j)$.*

Obviously, $t_i \in T(t_i, j)$. Besides, the distance between the sensors on $kp_j(t_i)\widehat{)kp}_{j+1}(t_i)$ and t_i is equal to r. For the other targets which are in $T(t_i, j)\backslash\{t_i\}$, the distance must be less than r. Thus we classify the targets in $T(t_i, j)$ into two categories: t_i which is marked as grey target for $kp_j(t_i)\widehat{)kp}_{j+1}(t_i)$, and $T(t_i, j)\backslash\{t_i\}$ which are marked as black targets for $kp_j(t_i)\widehat{)kp}_{j+1}(t_i)$.

Sometimes, $T(t_i, j)\backslash\{t_i\}$ may be $\varnothing$. Thus we classify $kp_j(t_i)\widehat{)kp}_{j+1}(t_i)$ into two types: grey curved-boundary when $T(t_i, j)\backslash\{t_i\} = \varnothing$, black curved-boundary when $T(t_i, j)\backslash\{t_i\} \neq \varnothing$. A sequence of closed curved-boundary generate a subarea. We give the definition of covered-set.

Definition 7. *(Covered-set of subarea): We can get some curved-boundaries from targets set T. The curved-boundaries generate some subareas. Each subarea Ω' represents a target set which named covered-set. The covered-set is denoted by $T(\Omega')$. A sensor in Ω' can cover all targets in a covered-set.*

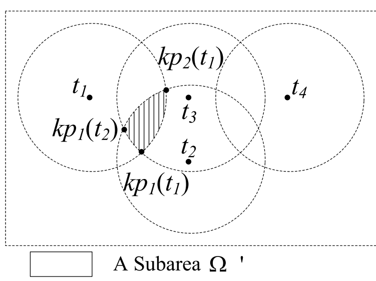


Fig. 2. Illustration of Definitions

Next, We give an instance to further illustrate the definitions above. As shown in Fig. 2, there are four targets in the surveillance region. We draw the detection-cycles of all targets, and divide the surveillance region into 11 disjoint subareas. The intersection point of $\odot t_1$ and $\odot t_3$, $kp_1(t_1)$ is the first key-point of t_1. Of course, we can also denote $kp_1(t_1)$ as $kp_1(t_3)$. A subarea Ω' is enclosed by three curved-boundaries, $kp_1(t_1)\widehat{)kp}_2(t_1)$, $kp_1(t_1)\widehat{)kp}_1(t_2)$ and $kp_1(t_2)\widehat{)kp}_2(t_1)$. We can find that $T(\Omega') = \{t_1, t_2, t_3\}$. $T(t_1, 1) = \{t_1, t_2, t_3\}$ and t_1 is a grey target in $T(t_1, 1)$.

In fact, we can easily prove that: if $kp_j(t_i)\widehat{)kp}_{j+1}(t_i)$ is a curved-boundary of a subarea Ω', then $T(\Omega') \subseteq T(t_i, j)$ and $T(t_i, j)\backslash T(\Omega') = \{t_i\}$ or $\varnothing$.

We can also infer that:

1) If $T(t_i, j)$ only contains grey target t_i, then $T(\Omega') = \{t_i\}$ or $\varnothing$.
2) If $kp_q(t_p)\widehat{)kp}_{q+1}(t_p)$ is also a curved-boundary of Ω', then $T(t_p, q)/T(t_i, j) = \{t_p\}$ or $\varnothing$.
3) $kp_{j+1}(t_i)\widehat{)kp}_{j+2}(t_i)$ and $kp_{j-1}(t_i)\widehat{)kp}_j(t_i)$ cannot be the curved-boundaries of Ω'.

Next, we give the detail to calculate the subareas according to the discussion above.

Given the set of all targets, firstly we can easily get all key-points and all curved-boundaries. Secondly we construct a new graph $G = (V, E)$ to calculate the subareas generated by all curved-boundaries. To construct G, we propose Graph Conversion (GC) algorithm which is shown in Algorithm 1. In G, each node $v(t_i, j)$ represents a curved-boundaries $kp_j(t_i)\widehat{)kp}_{j+1}(t_i)$. There is an edge between $v(t_i, j)$ and $v(t_p, q)$, if $p \neq i$, $T(t_p, q) \cap T(t_i, j) \neq \varnothing$ and $kp_j(t_i)\widehat{)kp}_{j+1}(t_i)$, $kp_q(t_p)\widehat{)kp}_{q+1}(t_p)$ are connected. We also need to calculate $T(t_i, j)$. In this paper, we select the mid-point of the curved-boundary and calculate the distance from the mid-point to the targets. If the distance is no more than r, then the target is in $T(t_i, j)$. Then a subarea corresponds to a cycle L in G which satisfies a specific condition as follows: if $v(t_i, j)$ is in L, then $\forall v(t_p, q)$ in L, $T(t_i, j)\backslash\{t_i\} \subseteq T(t_p, q)$.

Algorithm 1 Graph Conversion (GC) algorithm

Input: The set of targets, T;
Output: A new graph, $G = (V, E)$;
Calculate the key-points set of t_i, $\forall t_i \in T$.
Calculate the covered set of all curved-boundaries;
Generate $G = (V, E)$. Each node in G represents a curved-boundaries, two nodes have an edge if the corresponding curved-boundaries are connected and have different centers.
return $G = (V, E)$;

To find this kind of cycle, we propose Variant Depth First Search (VDFS) algorithm. When we need to find all subareas involved with the node $v(t_i, j)$, we start from $v(t_i, j)$ and search the graphs until we get a cycle corresponding to Ω'. At the first step of search, we move from $v(t_i, j)$ to $v(t_{p1}, q_1)$ and check weather $t_i \in T(t_{p1}, q_1)$. If $t_i \notin T(t_{p1}, q_1)$, $T(\Omega') = T(t_i, j)\backslash\{t_i\}$. Otherwise, $T(\Omega') = T(t_i, j)$. Then we move forward to the next node $v(t_{p2}, q_2)$, only if $T(\Omega') \subseteq T(t_{p2}, q_2)$ and $T(t_{p2}, q_2)\backslash\{t_{p2}\} \subseteq T(\Omega')$. When we return back to $v(t_i, j)$, we get a cycle. Obviously, a curved boundary is involved in two subareas. Thus we can only find two cycles when we start from $v(t_i, j)$ and the two cycles have no public edges.

The detail of VDFS is shown in Algorithm 2 and 3.

To further illustrate GC and VDFS, we give a simple example as shown in Fig. 3. There are two targets and four curved-boundaries in this example. Thus we can generate a new graph $G = (V, E)$ and $|V| = 4$. $v(t_1, 1)$ and $v(t_1, 2)$ do not have an edge, since they have the same center t_1. $v(t_1, 1)$ and $v(t_2, 2)$ also do not have not an edge, since $T(t_1, 1) \cap T(t_2, 2) = \varnothing$.

Then we try to find all three subareas. we can search from $v(t_1, 1)$. Obviously, $T(t_1, 1) = \{t_1\}$, then $T' = \{t_1\}$ in Algorithm 3. We add $v(t_1, 1)$ and $v(t_2, 1)$ to the path in turn. Since $T(t_1, 2)\backslash\{t_1\} \nsubseteq T'$, $v(t_1, 2)$ cannot add to the path. Finally, we get a cycle $v(t_1, 1) \to v(t_2, 1) \to v(t_1, 1)$ and the cycle represent the subarea Ω_1 such that $T(\Omega_1) = \{t_1\}$. Next,

Algorithm 2 Variant Depth First Search (VDFS) algorithm

Input: $G = (V, E)$;
Output: The paths involved with $v(t_i, j)$;
for all $v \in V$ **do**
  visited(v)=$false$;
end for
visited($v(t_i, j)$)=$true$;
if $T(t_i, j)\backslash\{t_i\} \neq \varnothing$ and the $path$ corresponding to $T(t_i, j)\backslash\{t_i\}$ is not found **then**
  EXPLORE($G, v(t_i, j), T(t_i, j)\backslash\{t_i\}$);
end if
if We have not find $path$ corresponding to $T(t_i, j)$ **then**
  EXPLORE($G, v(t_i, j), T(t_i, j)$);
end if
return ;

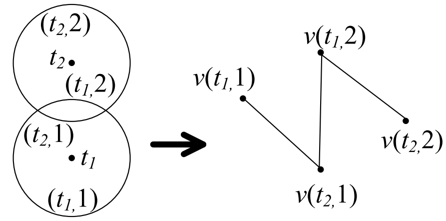
Algorithm 3 EXPLORE($G,\ v(t_i, j),\ T'$)

Input: $G = (V, E)$, $v(t_i, j)$, T';
Output: The paths involved with $v(t_i, j)$;
if T' is explored **then**
  return ;
end if
$path$.add($v(t_i, j)$);
if $T' \subseteq T(t_p, q)$ and $T(t_p, q)\backslash\{t_p\} \subseteq T'$ and $(v(t_i, j), v(t_p, q)) \in E$ and visited$v(t_p, q)$ = false **then**
  visited$v(t_p, q)$ = true;
  EXPLORE($G, t(t_p, q), T'$)
end if
if No such $v(t_p, q)$ exists **then**
  Output $path$;
  $path$=null;
  return ;
end if
return ;

we search from $v(t_2, 1)$. Since $T(t_2, 1)\backslash\{t_2\} = T(\Omega_1) = \{t_1\}$ and we have found Ω_1, we only find the subarea Ω_2 such that $T(\Omega_2) = T(t_2, 1) = \{t_1, t_2\}$. We add $v(t_2, 1)$ and $v(t_1, 2)$ to the path, and get Ω_2. In this way, we can easily find all three subareas Ω_1, Ω_2, Ω_3. $T(\Omega_1) = \{t_1\}$, $T(\Omega_2) = T(t_2, 1) = \{t_1, t_2\}$, $T(\Omega_3) = T(t_2, 1) = \{t_2\}$.

Theorem 1. *The time complexity in the first phase is $O(n^2)$.*

Proof: We have n targets in total. Each pair of targets have 2 key-points at most. Thus we have $2(n-1)$ key-points at most for each target, and we have $n(n-1)$ key-points at most in total. The time complexity of GC is $O(n^2)$. There are no more than $2n(n-1)$ curved-boundaries. Thus in the new graph that we construct, the number of vertices is less that $2n(n-1)$. Obviously, each vertex averagely has no more than 4 edges and the number of edges is less than $4n(n-1)$ in total. When we search all subareas, we visit a vertex at most twice, since a curved-boundaries is involved with two sub areas at most. Thus the time complexity of VDFS is also $O(n^2)$. In the first phase, we only apply GC and VDFS. Therefore, the time complexity in the first phase is $O(n^2)$. ■



T(t_1,2)=T(t_2,1)={t_1, t_2}; T(t_1,1)={t_1};
T(t_2,2)={t_2};

Fig. 3. Illustration for GC and VDFS

C. The Second Phase

After all the preprocessing in the first phase, we can schedule the mobile sensors in the second phase. In the scheduling, we need to select stations where the mobile sensors start and the destinations to cover the targets. Obviously, the destination must be the subareas that we find in the first phase. Since the mobile sensor in the same subarea can cover the same targets set, we just need select subareas as the destinations and calculate the minimized sensor movement distance.

Firstly, we study how to calculate the minimized movement distance when a mobile sensor move from the station p_i to the subarea Ω_j. Apparently, to minimize movement distance, the mobile sensor must stop at the curved-boundaries of Ω_j. Thus we just need to calculate the distance between the station and all curved-boundaries of Ω_j respectively. Then the minimum distance is the sensor movement distance. We calculate distance between p_i and a curved-boundaries according to Theorem 2.

Theorem 2. *We have a curved-boundary $\overset{\frown}{kp_j(t_i)kp_{j+1}(t_i)}$ and a station p_i. If the segment p_it_i intersects with the curved-boundary and the intersection point is O, the minimum distance is $|p_iO|$; otherwise, the minimum distance is $|kp_j(t_i)p_i|$ or $|kp_{j+1}(t_i)p_i|$.*

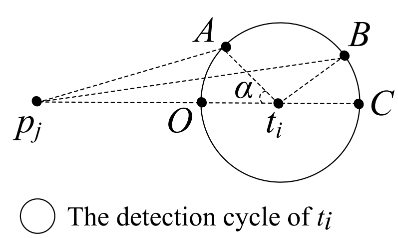


Fig. 4. The proof of Theorem 2

Proof: As shown in Fig. 4, A is a point on the detection cycle of t_i. According to Cosine Law, $|Ap_j| = |p_jt_i|^2 +$

$|At_i|^2 - 2\cos\alpha|p_jt_i||At_i|$. $|p_jt_i|$ and $|At_i|$ are constant value. Thus when A move from O to C along $\odot t_i$, $\cos\alpha$ is decreasing and $|Ap_j|$ is increasing. $|Ap_j|$ is minimum when $A = O$, $|Ap_j|$ is maximum when $A = C$.

We can assume that A is a point at the curved-boundaries, $A \neq kp_j(t_i)$, $A \neq kp_{j+1}(t_i)$, $A \neq O$, and $|Ap_j|$ is the minimum distance. Then we can move A to make α smaller, and Ap_j become smaller. This leads to a contradiction. Therefore, the theorem is proved. ∎

According to the discussion above, we can easily calculate the minimum distance between subareas and stations. When we select a subarea, we just need to ask the nearest station send a mobile sensor. Thus each subarea σ_i has a covered targets set and a weight $w(\sigma_i)$ which represents the minimum distance.

Next, we discuss about how to select subareas with minimum total weight. The main idea of the selection is partition and shifting [33].

In advance, we give a simple algorithm, called Partition algorithm. Assume that Q is a square which exactly contains all targets in the surveillance region. Firstly, we divide the square Q into a grid of squares, named as cells. The size of each cell is equal to $2mr \times 2mr$, where m is a constant and $m \in N^+$. Then we can apply brute-force search algorithm and find the optimal solution for each cell. Finally, we combine the solutions of all cells and get a solution of the original problem. The detail is shown in Algorithm 4.

Algorithm 4 Partition algorithm

Input: T; A grid of squares that contain all targets, Q;
Output: The subareas that we select to cover T;
Divide Q into cells which denoted as $cell(Q)$, and the size of each cell is $2mr \times 2mr$;
for each $e \in cell(Q)$ **do**
  Select the subareas which can cover all targets in e, such that the sum of all selected subareas weight is minimum;
end for
return All the selected subareas;

Theorem 3. *The time complexity of the Partition algorithm is* $n^{O(m^2)}$.

Proof: We analyze the time complexity for each cell e firstly. Assume that the number of targets in e is n_e. Then the number of curved-boundaries is less than n_e^2, since each target correspond to n_e curved-boundaries at most. Each subarea has at least two curved-boundaries and each curved-boundary is shared by only two neighbouring subareas. Thus the number of subareas is less than n_e^2. Note that when we select a subarea, and make a mobile sensor move in it, the sensor can cover a $\sqrt{2}r/2 \times \sqrt{2}r/2$ square. The size of e is $2mr \times 2mr$. We need to select at most $\lceil\sqrt{2}m\rceil^2$ subareas. In conclusion, the number of possible solutions for e is at most $n_e^{2\lceil\sqrt{2}m\rceil^2}$.

According to the discussion above, we can infer that the time complexity of Partition algorithm is as follows:

$$\sum_{e\in cell(Q)} n_e^{O(m^2)} \le (\sum_{e\in cell(Q)} n_e)^{O(m^2)} = n^{O(m^2)}$$

∎

Theorem 4. *The approximation ratio of the Partition algorithm is* 4.

Proof: Assume the optimal solution of selection is S^*, and the feasible solution we get from Partition algorithm is S. $S^*(e)$ is the set of subareas which are contained by S^* and $\sigma_i \in S^*(e)$ if and only if some targets in the covered-set of σ_i are distributed in the cell e. $S(e)$ is the set of subareas which are contained by S and intersect with the cell e. Obviously, $S^*(e)$ is a feasible solution for e, and $S(e)$ is the optimal solution for e. Thus $\sum_{\sigma_i\in S^*(e)} w(\sigma_i) \ge \sum_{\sigma_i\in S(e)} w(\sigma_i)$. Assume that S_k^* is the set of subareas which are contained by S^*, and $\sigma_i \in S_k^*$ if and only if the covered-set of σ_i is distributed in k cells. Apparently, $1 \le k \le 4$. Thus we can infer that:

$$\sum_{\sigma_i\in S} w(\sigma_i) = \sum_{e\in cell(Q)} \sum_{\sigma_i\in S(e)} w(\sigma_i) \le \sum_{e\in cell(Q)} \sum_{\sigma_i\in S^*(e)} w(\sigma_i)$$
$$= \sum_{k=1}^{4} \sum_{\sigma_i\in S_k^*} k \cdot w(\sigma_i) \le \sum_{k=1}^{4} \sum_{\sigma_i\in S_k^*} 4w(\sigma_i) = 4 \sum_{\sigma_i\in S^*} w(\sigma_i)$$

Then, the Theorem 4 is proved. ∎

Next, We try to optimize the algorithm by the method of shifting. We can find that the gap between the optimal solution and our solution is generated by the subareas which intersect more than one cell. Since the targets are distributed in the surveillance region randomly and uniformly, the subareas and the sensors are also evenly distributed in the square Q. As shown in Fig. 5, when the sensors stop in the area which is labeled as k $(1 \le k \le 4)$, the covered-set of subareas where the sensors stop, distributed in k cells.

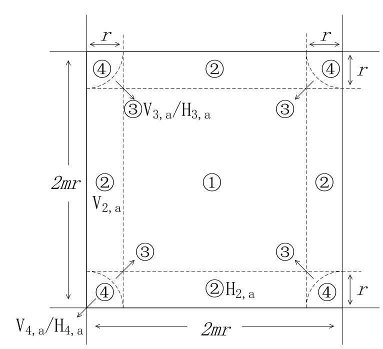


Fig. 5. Illustration for the intersection

Assume that $S_k^*(e)$ is the set of subareas which are contained by $S^*(e)$, and $\sigma_i \in S_k^*(e)$ if and only if the covered-set of σ_i is distributed in k cells. According to Fig. 5, we can

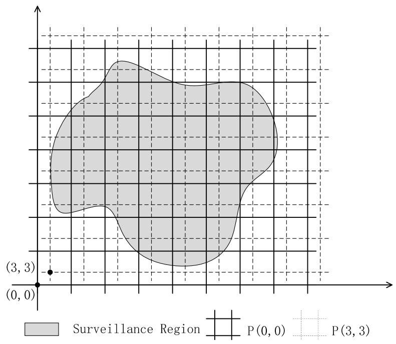


Fig. 6. Illustration for the shifting

infer that as follows:

$$|S_1^*(e)| = (m-1)^2/m^2|S^*(e)|;$$
$$|S_2^*(e)| = 2(m-1)/m^2|S^*(e)|;$$
$$|S_3^*(e)| = (4-\pi)/4m^2|S^*(e)|;$$
$$|S_4^*(e)| = \pi/4m^2|S^*(e)|;$$

Then we can conclude that $\sum_{e\in cell(Q)}|S^*(e)| = \sum_{e\in cell(Q)}\sum_{k=1}^4 k|S_k^*(e)| = (1+\frac{2}{m}+\frac{\pi}{m^2})|S^*|$. When the stations are evenly distributed in the surveillance region and we denote the average weight of subareas as $\bar{w}$, we can infer that:

$$\sum_{\sigma_i\in S} w(\sigma_i) = \sum_{e\in cell(Q)}\sum_{\sigma_i\in S(e)} w(\sigma_i) \le \sum_{e\in cell(Q)} |S^*(e)|\bar{w}$$
$$= (1+\frac{2}{m}+\frac{\pi}{m^2})|S^*|\bar{w} = (1+\frac{2}{m}+\frac{\pi}{m^2})\sum_{\sigma_i\in S^*} w(\sigma_i)$$

Thus we can conclude that the expected approximation ratio of Partition Algorithm is $1+\frac{2}{m}+\frac{\pi}{m^2}$. We can apply shifting method for derandomization.

As shown in Fig. 6, we define the partition according to the lower-left corner. $P(3,3)$ denotes the partition which has a lower-left corner at $(3,3)$. $P(0,0)$ denotes the partition which has a lower-left corner at $(0,0)$. Obviously, the partition $P(0,0)$ and the partition $P(2mr,2mr)$ is the same partition. The main idea of the shifting algorithm is shift the partition from $P(0,0)$ to $P(2r(m-1),2r(m-1))$, and apply Algorithm 4 to find a solution of k-MMTC. Finally, we chose the optimal solution and the algorithm is end. The detail of shifting algorithm is shown in Algorithm 5.

Algorithm 5 Shifting algorithm

Input: T;
Output: The subareas that we select to cover T;
1: **for** $(i=0; i<m; ++i)$ **do**
2: Apply Algorithm 4 in the partition $P(2ri,2ri)$ and get a set of selected subareas;
3: **end for**
4: Select the set with minimum total weight;
5: **return** The selected set;

Theorem 5. *The time complexity of Algorithm 5 is $mn^{O(m^2)}$.*

Proof: We invoke Algorithm 4 for m times and the time complexity of Algorithm 4 is $n^{O(m^2)}$. Apparently, the time complexity of Algorithm 5 is $mn^{O(m^2)}$. ■

Theorem 6. *The approximation ratio of Algorithm 5 is $1+3/m$.*

Proof: We assume that Sen^* denotes the location of sensors in the optimal solution. Then each sensor is assigned to a selected subarea and we can infer that $|Sen^*| = |S^*|$. S_a denotes the solution of Algorithm 4 in the partition $P(2ra,2ra)$. As shown in Fig. 5, $H_{k,a}$ denotes a subset of S^*. $SenH_{k,a}$ denotes the sensors set assigned to $H_{k,a}$. $\forall s \in SenH_{k,a}$, the detection cycle of s intersects the horizontal line of $P(2ra,2ra)$, and intersects with k cells. Similarly, we define $V_{k,a}$ and $SenV_{k,a}$. $\forall s\in SenV_{k,a}$, the detection cycle of s intersects the vertical line of $P(2ra,2ra)$, and intersects with k cells. Then we can infer that: $H_{3,a}=V_{3,a}$, $H_{4,a}=V_{4,a}$.

We define that $H_a = \sum_{k=2}^4 H_{k,a}$ and $V_a=\sum_{k=2}^4 V_{k,a}$. Then we can infer that:

$$\sum_{\sigma_i\in S_a} w(\sigma_i) = \sum_{e\in cell(Q)}\sum_{\sigma_i\in S(e)} w(\sigma_i) \le \sum_{e\in cell(Q)}\sum_{\sigma_i\in S^*(e)} w(\sigma_i)$$
$$= \sum_{\sigma_i\in S^*} w(\sigma_i) + \sum_{k=2}^4\sum_{\sigma_i\in H_{k,a}}(k-1)w(\sigma_i) + \sum_{\sigma_i\in V_{2,a}} w(\sigma_i)$$
$$= \sum_{\sigma_i\in S^*} w(\sigma_i) + \sum_{\sigma_i\in H_a} w(\sigma_i) + \sum_{\sigma_i\in V_a} w(\sigma_i) + \sum_{\sigma_i\in H_{4,a}} w(\sigma_i)$$

We can conclude that:

$$\sum_{\sigma_i\in S_a} w(\sigma_i) \le \sum_{\sigma_i\in S^*} w(\sigma_i) + 2\sum_{\sigma_i\in H_a} w(\sigma_i) + \sum_{\sigma_i\in V_a} w(\sigma_i)$$

Apparently, $\forall s\in S^*$, the detection cycle of s cannot intersect both the horizontal line of $P(2ra,2ra)$ and the horizontal line of $P(2rb,2rb)$ at the same time if $a\ne b$. Thus, we can infer that:

$$\sum_{a=0}^{m-1}\sum_{\sigma_i\in H_a} w(\sigma_i) \le \sum_{\sigma_i\in S^*} w(\sigma_i)$$

Similarly,

$$\sum_{a=0}^{m-1}\sum_{\sigma_i\in V_a} w(\sigma_i) \le \sum_{\sigma_i\in S^*} w(\sigma_i)$$

Therefore, we can infer that: $\sum_{a=0}^{m-1}\sum_{\sigma_i\in S_a} w(\sigma_i) \le \sum_{a=0}^{m-1}(\sum_{\sigma_i\in S^*} w(\sigma_i) + 2\sum_{\sigma_i\in H_a} w(\sigma_i) + \sum_{\sigma_i\in V_a} w(\sigma_i))$.

Simplify the equation above, we can get that:

$$\sum_{a=0}^{m-1}\sum_{\sigma_i\in S_a} w(\sigma_i) \le (m+3)\sum_{\sigma_i\in S^*} w(\sigma_i)$$

$$\frac{1}{m}\sum_{a=0}^{m-1}\sum_{\sigma_i\in S_a} w(\sigma_i) \le (1+\frac{3}{m})\sum_{\sigma_i\in S^*} w(\sigma_i)$$

The average value of all solutions that we get from the partitions $P(0,0), P(2r,2r), \cdots, P(2(m-1)r, 2(m-1)r)$

is $(1+\frac{3}{m})\sum_{\sigma_i \in S^*} w(\sigma_i)$. In Algorithm 5, we select the solution with minimum value. Thus the approximation ratio of Algorithm 5 is less than $1+3/m$. Then Theorem 6 is proved. ■

In summary, the detail of EEMA is shown in Algorithm 6.

Algorithm 6 EEMA algorithm

Input: k stations which can send mobile sensors to cover the targets in the surveillance region, P; T;

Output: The schedule of mobile sensors to cover T;

1: Apply Algorithm 1 to preprocess the targets T;
2: Apply Algorithm 2 to get all subareas generated by T;
3: Calculate the weight of all subareas according to the position of P;
4: Apply Algorithm 5 to select subareas;
5: Choose sensors from P to the selected subareas;
6: **return** The solution of k-MMTC;

According to Theorem 5 and Theorem 6, we can conclude that $\forall \varepsilon > 0$, EEMA can be a $(1+\varepsilon)$-approximation algorithm for k-MMTC problem that runs in time $n^{O(1/\varepsilon^2)}$, when we set $m = \lceil 3/\varepsilon \rceil$.

V. Evaluation

In this section, we use C++ and Matlab to conduct some simulations and evaluate the performance of the proposed algorithm EEMA. To confirm the effectiveness and efficiency of EEMA, we compare the numerical results with the optimal solution and TV-greedy [32].

TABLE I
Simulation Parameters

Parameter	Surveillance Region	$\|T\|$	$\|P\|$	r	m
Simulation A	$50m \times 50m$	20	10	$1m$	3
Simulation B	$500m \times 500m$	50-230	20-400	$1m$	9

A. Comparison with the optimal solution

The detailed parameters in the evaluation are listed in the simulation A of Table I. The surveillance region is $50m \times 50m$. The number of targets is 20. The number of stations which can send sensors is 10. The sensing range of sensors is $1m$. In Algorithm 4, we set the value of m as 3.

Since the initial deployment of sensors and targets is random, we run the algorithm for 1000 times with different initial deployment of sensors and targets and the numerical results is shown as Fig. 7.

If $m = 3$, we can infer that the approximation ratio of EEMA is 2. In the Fig. 7, we can find that the experiment results is consistent with Theorem 6. Besides, we also find that in the most cases the approximation ratio is between 1.4 and 1.6. We can conclude that EEMA works well for random small scale inputs.

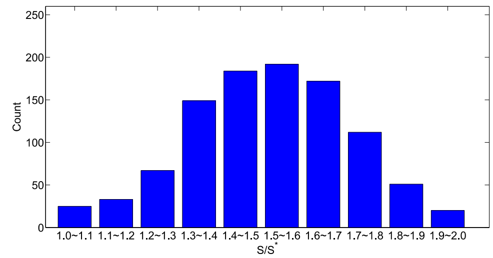


Fig. 7. Comparison with the optimal solution

B. Comparison with TV-greedy

We can hardly calculate the result for large scale inputs. Thus we compare EEMA with TV-greedy [32]. The detailed parameters in the evaluation are listed in the simulation B of Table I.

In this subsection, we also run EEMA and TV-greedy for 1000 times with the same number of targets and stations. Then we calculate the average value. The result is shown in Fig. 8 and Fig. 9.

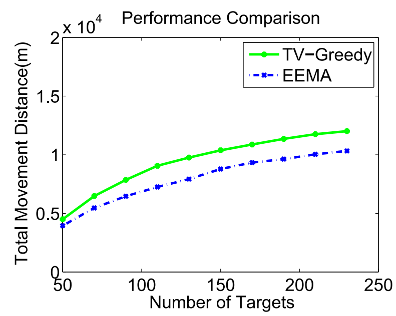


Fig. 8. Comparison with TV-Greedy with different targets density

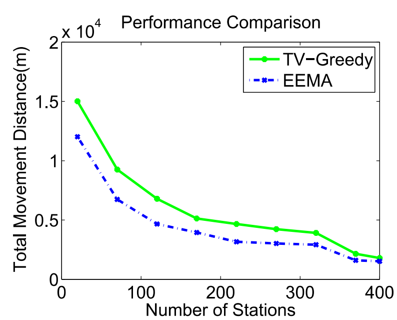


Fig. 9. Comparison with TV-Greedy with different stations density

Firstly, we set $|P| = 100$ and change the number of targets from 50 to 230. The numerical result is shown as Fig. 8. In the Fig. 8, we can see that EEMA works more effectively than TV-Greedy. Moreover, EEMA can save more energy than TV-Greedy especially when the number of targets is large.

Then we set $|T| = 100$ and change the number of stations from 20 to 400. The numerical result is shown as Fig. 9. In the Fig. 9, we can see that EEMA still works well and is more energy efficient than TV-Greedy. However, when the number of stations becomes large, EEMA save less energy than TV-Greedy. The reason is that when the number of stations is large, each target has many stations nearby we can easily get an energy efficient schedule.

VI. Conclusion

In this paper, we consider a variation of target coverage problem in mobile sensor network named *k-Sink Minimum Movement Target Coverage* (k-MMTC). To solve this problem, we propose a polynomial-time approximation scheme (PTAS), named *Energy Effective Movement Algorithm* (EEMA). EEMA can be divided into two phrases. In the first phrase, we partition the objective region into subareas according to the detection cycle of each target. Then in the second phrase, we select some subareas and schedule sensors to cover them respectively. The approximation ratio of EEMA is $1 + \varepsilon$ and the time complexity of EEMA is $n^{O(1/\varepsilon^2)}$. We also provide several numerical experiments to compare the results of EEMA with the optimal solution and one of previous works. The simulation results validate the effectiveness and efficiency of EEMA. In all, EEMA is the first PTAS for sensor movement scheduling to achieve target coverage requirement.

In the future work, we will study the problem where each station p_i can only send q_i ($q_i > 0$) sensors.

References

[1] B. Wang, *Coverage control in sensor networks*. Springer Science & Business Media, 2010.

[2] H. M. Ammari, "On the problem of k-coverage in mission-oriented mobile wireless sensor networks," *Computer Networks (COMNET)*, vol. 56, no. 7, pp. 1935–1950, 2012.

[3] Y. C. Wang and Y. C. Tseng, "Distributed deployment schemes for mobile wireless sensor networks to ensure multilevel coverage," *IEEE Transactions on Parallel & Distributed Systems (TPDS)*, vol. 19, no. 9, pp. 1280–1294, 2007.

[4] B. Liu, O. Dousse, P. Nain, and D. Towsley, "Dynamic coverage of mobile sensor networks," *IEEE Transactions on Parallel & Distributed Systems (TPDS)*, vol. 24, no. 2, pp. 301–311, 2011.

[5] S. He, J. Chen, X. Li, X. Shen, and Y. Sun, "Cost-effective barrier coverage by mobile sensor networks," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2012, pp. 819–827.

[6] T. Wimalajeewa and S. K. Jayaweera, "A novel distributed mobility protocol for dynamic coverage in sensor networks," in *IEEE International Conference on Global Telecommunications Conference (GLOBECOM)*, 2010, pp. 1–5.

[7] P. Sahoo and W.-C. Liao, "Hora: A distributed coverage hole repair algorithm for wireless sensor networks," *IEEE Transactions on Mobile Computing (TMC)*, vol. 14, no. 7, pp. 1397–1410, July 2015.

[8] D. Zorbas and T. Razafindralambo, "Prolonging network lifetime under probabilistic target coverage in wireless mobile sensor networks," *Computer Communications (CC)*, vol. 36, no. 9, pp. 1039–1053, 2013.

[9] S. Mini, S. K. Udgata, and S. L. Sabat, "Sensor deployment for probabilistic target k-coverage using artificial bee colony algorithm," in *Swarm, Evolutionary, and Memetic Computing*, 2011, pp. 654–661.

[10] D. K. Yau, N. K. Yip, C. Y. Ma, N. S. Rao, and M. Shankar, "Quality of monitoring of stochastic events by periodic and proportional-share scheduling of sensor coverage," *ACM Transactions on Sensor Networks (TOSN)*, vol. 7, no. 2, pp. 2019–2021, 2010.

[11] M. P. Habib, S.J., "A coverage restoration scheme for wireless sensor networks within simulated annealing," in *International Conference On Wireless And Optical Communications Networks (ICWOCN)*, 2010, pp. 1–5.

[12] C. Sahin, M. Uyar, S. Gundry, and E. Urrea, "Self organization for area coverage maximization and energy conservation in mobile ad hoc networks," *Transactions on Computational Science XV*, vol. 15, pp. 49–73, 2012.

[13] F. Lin, Z. Sun, and T. Qiu, "Genetic algorithm-based 3d coverage research in wireless sensor networks," in *International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS)*, 2013, pp. 623–628.

[14] H. P. Corporation, "Level set based coverage holes detection and holes healing scheme in hybrid sensor network," *International Journal of Distributed Sensor Networks (JDSN)*, vol. 41, no. 4, pp. 128–134, 2013.

[15] Y. Li, Y.-q. Song, Y.-h. Zhu, and S. Ren, "Deploying wireless sensors for differentiated coverage and probabilistic connectivity," in *IEEE Wireless Communications and Networking Conference (WCNC)*, 2010, pp. 1–6.

[16] R. Andreacute, S. Alok, and M. Sevaux, "Column generation algorithm for sensor coverage scheduling under bandwidth constraints," *Networks*, vol. 60, no. 3, pp. 141–154, 2012.

[17] W. Xue, S. Wang, and J. jie Ma, "An improved co-evolutionary particle swarm optimization for wireless sensor networks with dynamic deployment," *Sensors*, vol. 2007, no. 3, pp. 354–370, 2007.

[18] W. Ismail, W.Z., and S. Manaf, "Study on coverage in wireless sensor network using grid based strategy and particle swarm optimization," in *IEEE International Conference on Circuits and Systems (APCCAS)*, 2010, pp. 1175–1178.

[19] T.-J. Su, M.-Y. Huang, and Y.-J. Sun, "An adaptive particle swarm optimization for the coverage of wireless sensor network," *Communications in Computer and Information Science (CCIS)*, pp. 386–391, 2011.

[20] M. Esnaashari and M. Meybodi, "A learning automata based scheduling solution to the dynamic point coverage problem in wireless sensor networks," *Computer Networks (COMNET)*, vol. 54, no. 14, pp. 2410–2438, 2010.

[21] H. Mohamadi, A. S. Ismail, and S. Salleh, "Solving target coverage problem using cover sets in wireless sensor networks based on learning automata," *Wireless Personal Communications (WPC)*, vol. 75, no. 1, pp. 447–463, 2014.

[22] O. C, K. D, and G. B., "Probabilistic dynamic deployment of wireless sensor networks by artificial bee colony algorithm," *Sensors*, vol. 11, no. 6, pp. 6056–6065, 2011.

[23] D. Tao, S. Tang, and L. Liu, "Constrained artificial fish-swarm based area coverage optimization algorithm for directional sensor networks," *IEEE International Conference on Mobile Ad-Hoc and Sensor Systems (MASS)*, vol. 411, no. 6, pp. 304–309, 2013.

[24] L. Ding, W. Wu, W. J., L. Wu, Z. Lu, and W. Lee, "Constant-approximation for target coverage problem in wireless sensor networks," *IEEE International Conference on Computer Communications (INFOCOM)*, vol. 131, no. 5, pp. 1584–1592, 2012.

[25] C. Liu and G. Cao, "Spatial-temporal coverage optimization in wireless sensor networks," *IEEE Transactions on Mobile Computing (TMC)*, vol. 10, no. 4, pp. 465–478, 2011.

[26] X. Xu and M. Song, "Restricted coverage in wireless networks," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2014, pp. 558–564.

[27] J. Chen, J. Li, Lai, and T.H., "Energy-efficient intrusion detection with a barrier of probabilistic sensors: Global and local," *IEEE Transactions on Wireless Communications (TWC)*, vol. 12, no. 9, pp. 4742–4755, 2013.

[28] G. Barun and M. P. Sarathi, "Point and area sweep coverage in wireless sensor networks," in *International Symposium on Modeling Optimization in Mobile, Ad Hoc Wireless Networks (WiOpt)*, 2013, pp. 140–145.

[29] B. Gorain and P. S. Mandal, "Approximation algorithms for sweep coverage in wireless sensor networks," *Journal of Parallel and Distributed Computing (JPDC)*, vol. 74, pp. 2699–2707, 2014.

[30] H. Huang, C.-C. Ni, X. Ban, A. Jie Gao, Schneider, and S. Lin, "Connected wireless camera network deployment with visibility coverage," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2014, pp. 1204–1212.

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

[32] Z. Liao, S. Zhang, J. Cao, W. Wang, and J. Wang, "Minimizing movement for target coverage in mobile sensor networks," in *IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2012, pp. 194–200.

[33] D.-Z. Du, K.-I. Ko, and X. Hu, *Design and analysis of approximation algorithms*. Springer Science & Business Media, 2012, vol. 62.