Efficient Scheduling Strategies for Mobile Sensors in Sweep Coverage Problem

Zhiyin Chen*, Xudong Zhu*, Xiaofeng Gao*§, Fan Wu*, Jian Gu† and Guihai Chen*
*Shanghai Jiao Tong University, Department of Computer Science and Engineering, P.R.China
†The Third Research Institute of Ministry of Public Security, Testing Center, P.R.China
Email: chenzhiyin@sjtu.edu.cn, xudongzhu42@gmail.com,
gao-xf@cs.sjtu.edu.cn, fwu@cs.sjtu.edu.cn, gujian@mctc.gov.cn, gchen@cs.sjtu.edu.cn

***Abstract*—Nowadays, with the development of micro-electro-mechanical technologies, sweep coverage are more and more popular in wireless sensor networks, which is also applied widely in other scenarios, such as message ferrying and data routing in the ad-hoc network. In order to reduce the sweep cycle and the number of required mobile sensors, we propose the *Distance-Sensitive-Route-Scheduling* (DSRS) problem, which is the first to consider the effect of sensing range. We prove that DSRS is NP-complete, and consider two different scenarios: the single kissing-point case and the general case. The former case requires a mobile sensor to change its moving direction after visiting a target. Correspondingly, we propose an approximation ROSE to schedule the routes of mobile sensors efficiently. For the latter general case, we present another approximation G-ROSE based on ROSE. We further characterize the non-locality property and design a distributed sweep algorithm D-ROSE, cooperating sensors to guarantee the required sweep requirements with the best effort. Our algorithms is scalable to different sweep coverage problems involving route schedules. We compare our algorithms with several previous algorithms, and the simulation results show that our algorithms greatly outperform other works especially with large sensing range, which can be improved up to 45%.**

I. Introduction

Coverage problem is a critical and fundamental issue in wireless sensor networks. It has a great impact on the energy consumption, network performance, and sensor deployment. In the past, many literatures have researched the coverage problem. However, mobile sensors are more and more popular [1], and the popularity of mobile sensors generates a new kind of coverage problem, *sweep coverage* [2], [3].

In traditional coverage problems, we usually require a large number of redundant sensors to keep the coverage ratio and communication connectivity, which is undesirable due to the financial constraints. In sweep coverage, we just need to schedule the mobile sensors to move around the targets and detect them periodically. Thus the number of required sensors is decreased greatly. Sweep coverage problems can also be applied to many other fields, such as message ferrying and data routing in ad-hoc network [4]–[8]. Thus, the research on sweep coverage problem is of great importance.

A typical sweep coverage problem asks to cover a set of interested targets by a limited number of mobile sensors. Correspondingly, for many real-world applications, majority of previous works consider the sweep coverage problem from three aspects: determining the minimum number of required mobile sensors; calculating the minimum visit cycle T of all targets; and finding the minimum required velocity of mobile sensors. In all literatures, the route of mobile sensors is critical and we always expect to find a shortest route. However, it is difficult to achieve this goal and we usually try to find a next-best feasible solution, which is referred to as the *route scheduling problem*. Some of the previous works only solved this problem by heuristics without any performance bound, while some other works convert this problem into a Traveling-Salesman-Problem (TSP) and implement several related approximations to solve it.

No matter what methods are applied, all of those works neglected the impact of sensing range (detailed definition is represented in Section III-A) and required sensors to touch each target in the surveillance area. In fact, such strategy brings much redundant trajectory length. Actually, if we guarantee that the distance between a target and a sensor route is no more than the sensing range of the mobile sensor, then the target can be detected successfully in each period. In this way we may decrease the route length greatly and improve the system performance, especially when the sensing range is large.

Fig. 1 is an example to illustrate the importance of sensing range effect. Here $t_1, \cdots, t_5$ are targets to be detected (communicated), while the dotted circles denote their sensing ranges (communication ranges). Route 1 (dotted polygon) is a sensor route regardless of sensing range effect, while Route 2 (solid polygon) is the shortest route considering sensing range effect. Easy to see, the total length of the first route is much longer than the second one.

In this paper, to find the shortest route for mobile sensors with large sensing ranges, we propose three route scheduling algorithms, ROSE, G-ROSE, and D-ROSE respectively. According to our investigation, we find that a best sensor route usually satisfies a special property, named *kissing-property*, which brings great benefit to find a feasible solution. To simplify the problem, we first study a restricted scenario with

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.

978-1-5090-1732-4/16/$31.00 ©2016 IEEE

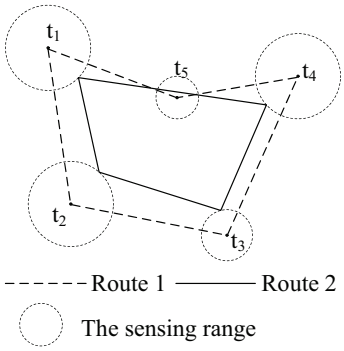


Fig. 1: Comparison of two mobile routes

only one sensor, and consider a special case: single kissing-point case. We propose an approximation named ROSE for this special case. Next, based on ROSE, we propose a generalized approximation G-ROSE without single kissing-point constraint. Finally, we consider the route scheduling problem for large scale networks with multiple sensors and propose a distributed algorithm D-ROSE. Inspired by divide-and-conquer technique, D-ROSE clusters the targets into several groups, each with a unique route but trajected by several mobile sensors cooperatively. We finally evaluate our design by mass numerical experiments. In the evaluation, we compare G-ROSE with CSWEEP [2], [3] and ITSP [9]. The simulation results show that G-ROSE outperforms the previous works greatly especially with large sensing range.

To summarize, our contributions in this paper are:

1) We are the first to consider the impact of sensing range for the route scheduling problem and prove the NP-completeness of the new problem. We also find the key property of a best solution, named *kissing-property*.
2) We propose three algorithms ROSE, G-ROSE, and D-ROSE to solve the route scheduling problem efficiently. ROSE deals with a restricted scenario with single sensor and single kissing-point constraints, while G-ROSE is for single sensor multiple kissing-points case. D-ROSE is a distributed algorithm for the generalized version.
3) We compare our algorithms with several previous works and find that our algorithms outperform them greatly, especially when the sensing ranges are large.

The rest of this paper is organized as follows: Section II discusses the related works. In Section III, we formulate the route scheduling problem for single sensor case, prove its NP-completeness, and propose ROSE and G-ROSE. In Section V, we present the design of D-ROSE for large scale networks. We conduct the performance evaluation of G-ROSE in Section VI and finally, we conclude this work in Section VII.

II. Related Work

In recent years, many literatures have studied the sweep coverage problem, and we can briefly categorize the application of sweep coverage into three kinds:

1) Message ferrying [4]–[7], also called data gathering. In this scenario, a mobile robot sweeps around some stationary sites, which will produce valuable data. These sites are usually far apart and cannot communicate with each other directly. Thus, to keep these sites connective or send the valuable data to a sink node, a mobile robot will visit each site at least once in a predefined access cycle, T. Such kind of scenario usually appears in ad-hoc network and the Internet of Things.

2) Sensing coverage [2], [3], [9]–[13]. This scenario usually appears in wireless sensor network. When static sensors cannot sense all of the targets, we can use mobile sensors to detect the targets periodically. After sensing a target, the sensor will keep moving to the next target nearby. In this way, all targets can be detected in turn and each target can be sensed at least once in a predefined access cycle.

3) Devices control [14]. In this scenario, some static devices, such as machine tools, work at different positions. A mobile device sweeps around these devices and send instructions periodically. The devices will work according to the instructions.

Due to the limited battery energy and excessive producing cost of the sensors (as well as the robots), we need to find an energy efficient way to achieve sweep coverage by minimum number of sensors. Thus, related literatures usually focused on these problems: finding the minimum velocity for covering all targets with some sensors (mini-velocity-problem) [9], [11], [15], [16]; finding the minimum number of sensors to be deployed with a fixed sensor velocity (mini-sensors-problem) [2], [3], [10], [12], [13]; finding the minimum period T of all targets (mini-T-problem) [4], [5], [17], [18]. Surprisingly, although these problems are different, their solutions are very similar. Based on our study, we find that most of the related researches solved these problems from three aspects:

1) Route scheduling: Apparently, we need to find a shortest sensor route to visit the targets, if we want to save energy and reduce the number of mobile sensors. The route scheduling problem has always been reduced to an NP-complete problem Euclidean Traveling-Salesman-Problem (TSP). [19], [20] formulated the Integer-Linear-Programming. [6], [12], [19], [20] gave solutions by heuristic algorithms without performance guarantee. [2], [3], [9], [10], [21] designed constant-factor approximations with ratio greater than 2. All of these works [2], [3], [9]–[11], [21] adopted Prime algorithm for their satisfactory TSP path. Moreover, [10], [11] proved that mini-sensor-problem cannot be approximated within a factor of 2.

2) Targets clustering: Since different targets may have different access cycle constraints, if sensors move on the TSP path directly, they will visit some targets more frequently than necessary. Thus, [13], [15], [16] partitioned the targets according to their position and the access cycle of the targets. [4], [5], [9] split a TSP-based path into several loops directly.

3) Speed control: Some papers assumed that the velocity of sensors did not remain the same. Thus speed control was involved with energy and latency. [18] tried to maximize the plateau speed until we have a tight interval. However, no one discussed the relationship between energy and speed so far.

Compared with the previous works, we are the first to consider the sensing range effect (as well as communication range) in the route scheduling. Our algorithms are scalable for various sweep coverage problems mentioned above.

III. Problem Formulation and Discussion

In this section, we give some assumptions and definitions for the route scheduling problem for sweep coverage. Then, we prove its NP-completeness, for which we can hardly develop a polynomial time optimal algorithm.

A. Preliminary and Assumptions

Sometimes, due to the sensing delay constraint between a sensor and a target, in order to prevent sensing error, we have to maintain the detection between them for a period λ' in each access cycle. Assume that the sensing delay of all targets are the same and the velocity of sensors is a constant v. Then we just need to ensure that the total length of the route where the sensor can detect the target is greater than $v \cdot \lambda'$. Since the sensing delay cannot be too large, to simplify the problem, we guarantee that the minimum distance between the sensor and each target is smaller than r, where $r = r_o - v \cdot \lambda'/2$. Here r_o is the sensing radius of the sensors. Correspondingly, the sensing delay constraint will be satisfied. According to the value of r, we can define the detection-cycle and sensing range as follows.

Definition 1. *(Detection-circle): For each target t_i, we can draw a circle centered at t_i with radius r as the detection-circle of t_i (denoted as $\odot t_i$).*

Definition 2. *(Sensing Range): The region in $\odot t_i$ is named as the sensing range of t_i. In this paper, we declare that the sensing range is r, if the radius of t_i equal to r.*

Since we just need to schedule the sensor passing the overlap area when two detection circles intersect, we can easily reduce the situation when some detection circles intersect to the situation when no detection cycles intersect. Thus, assume that all targets is far apart and no detection circles intersect.

B. Problem Formulation

We study the problem in homogeneous network. Assume that there are n targets $T = \{t_1, t_2, \cdots, t_n\}$ in the surveillance region. As mentioned in Section III-A, the distance between each pair of targets is larger than $2r_o$, where r_o is the sensing radius of the sensors. A mobile sensor will sweep around the targets periodically. The sensing delay of the mobile sensor is λ' which means that we will maintain detection between the mobile sensor and every target for a moment λ' in each period. We assume that the velocity of the mobile sensor is a constant, v.

To measure the efficiency of sweep coverage, we give some definitions as follows.

Definition 3. *(Timely λ-sweep coverage): A target is said to be timely λ-sweep covered by a mobile sensor iff. the target is covered by the sensor at least once every λ time units and the detecting duration of the target is λ' time units every time.*

If a target is λ-sweep covered, the time interval λ is called the sweep period of the target. In practice, different targets may have different sweep periods. Thus we give the definition of global λ-sweep coverage.

Definition 4. *(Global λ-sweep coverage): A set of targets are globally sweep covered by a mobile sensor iff every target is timely λ-sweep covered.*

Our target is to design a schedule for sensor movement and get the minimum λ. Since the velocity of mobile sensors is a constant, we can get a global λ-sweep coverage with minimum λ, when the route of the sensor is the shortest in each sweep period. As mentioned in Section III-A, we can simplify the problem and just make sure that the minimum distance between the sensor and each target is smaller than r, where $r = r_o - v \cdot \lambda'/2$. We define the Distance-Sensitive-Route-Scheduling (DSRS) problem as follows:

Definition 5. *(Distance-Sensitive-Route-Scheduling problem): Given the locations of the targets T, the sensing delay λ', the sensing range r_o, and the velocity of the sensor v, the problem is to decide a shortest path on which the mobile sensor can sweep around the targets, such that the distance between each target and the path is less than r, where $r = r_o - v \cdot \lambda'/2$.*

Theorem 1. *The DSRS problem is NP-complete.*

Proof: We consider a special case when $r = 0$. Then the DSRS problem requires to find the shortest path for the mobile sensor to visit all targets. Obviously, it is a well known Euclidean Traveling-Salesman-Problem (TSP) which is NP-complete. The TSP is a special case of the DSRS problem. Thus the DSRS problem is NP-complete. ■

Many approximation algorithms can be used to find the TSP path. [22] proposed a well known polynomial time approximation scheme, PTAS, for Euclidean TSP. However, we cannot neglect the sensing range and apply the TSP path directly. Since the gap between the TSP path and the shortest path will be great with large sensing range. What is more, the approximation ratio is infinitely great if we consider the sensing range and apply TSP path directly.

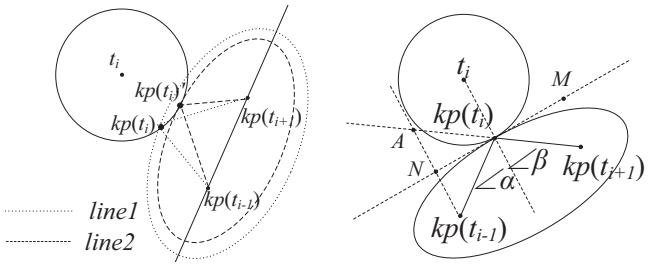
To find the solution of the DSRS problem, we just need to find a closed path for the mobile sensor to visit all targets in one period. We call this path as a sensor route. The mobile sensor can repeatedly move on the route periodically, then we can get a solution. In this paper, we schedule the route according to a property, named kissing-property, which is described as follows.

Definition 6. *(Kissing-property): The route of the mobile sensor has to intersect with the detection circle of every target and we need to make the route as short as possible.*

Since detection circles and the route have intersection points, we give the definition of kissing-point as follows:

Definition 7. *(Kissing-point): For each target t_i, the route of the mobile sensor and $\odot t_i$ may intersect. We define the intersection points as the kissing-points of the target, denoted as $KP(t_i)$.*

According to our observation, we give a lemma about the kissing-point as follows.



(a) Proof of Lemma 1 (b) Proof of Theorem 2

Fig. 2: Proof of Lemma 1 and Theorem 2

Lemma 1. $|KP(t_i)| \geq 1$.

Proof: According to the kissing-property, for each target t_i, the route has to intersect with $\odot t_i$. Then the route has at least one intersection point with $\odot t_i$. Thus, $|KP(t_i)| \geq 1$. ■

IV. Route Scheduling Algorithms for Single Mobile Sensor

In this section, we propose algorithms for single mobile sensor to solve the DSRS problem. To simplify the problem, we firstly consider the special scenario when $|KP(t_i)| = 1$ for each target t_i. Then we generalize this special case and get a solution for the general case when $|KP(t_i)| \geq 1$.

A. Single Kissing-point Case

1) Overview: We propose an approximation algorithm, named Efficiently Route Scheduling Algorithm (ROSE) for single kissing-point case. In this case, every target only has one kissing-point, namely $KP(t_i) = \{kp(t_i)\}$. Thus we can sort the targets, find the kissing-point of each target and connect the kissing-points in order. Here we implement TSP algorithm to get the order of kissing-points. Assume that the TSP path is: $t_1 \to t_2 \to \cdots \to t_n \to t_1$. Then we just need to find the positions of the kissing-points. Then we can get a sensor route: $kp(t_1) \to kp(t_2) \to \cdots \to kp(t_n) \to kp(t_1)$.

2) Characterizing the kissing-points: To decide the positions of kissing-points, we characterize kissing-points firstly.

Lemma 2. *If we draw an ellipse from three adjacent kissing-points $kp(t_{i-1}), kp(t_i), kp(t_{i+1})$, the focal points of the ellipse are $kp(t_{i-1}), kp(t_{i+1})$. $kp(t_i)$ is a point on the ellipse curve. Then the length of the route $kp(t_{i-1}) \to kp(t_i) \to kp(t_{i+1})$ is shortest when the ellipse curve is tangent to $\odot t_i$.*

Proof: Suppose that the ellipse curve is not tangent to $\odot t_i$ as shown in Fig. 2(a). Then the ellipse curve and $\odot t_i$ have two intersection points. We can narrow the ellipse curve until the narrowed ellipse curve and the circle have only one intersection point $kp(t_i)'$. Then, the narrowed ellipse curve has the same focal points, $kp(t_{i-1}), kp(t_{i+1})$. Obviously, the major axes of the narrowed ellipse is shorter than the former major axes. The length of path, $kp(t_{i-1}) \to kp(t_i)' \to kp(t_{i+1})$ which is marked as line 2, equals to the major axes of the narrowed ellipse. The length of the path, $kp(t_{i-1}) \to kp(t_i) \to kp(t_{i+1})$ which is marked as line 1, equals to the major axes of the former major axes. Thus, the path, $kp(t_{i-1}) \to kp(t_i)' \to kp(t_{i+1})$, is shorter than $kp(t_{i-1}) \to kp(t_i) \to kp(t_{i+1})$. Then $kp(t_i)$ is not the suitable kissing-point of the target t_i. ■

Lemma 2 serve as a basis for proving the following theorem.

Theorem 2. *The line l_i, which passes the point t_i and the point $kp(t_i)$, is the bisector of the angle $\angle kp(t_{i+1})kp(t_i)kp(t_{i-1})$. As shown in Fig. 2(b), where $\angle\alpha = \angle\beta$.*

Proof: In Fig. 2(b), We draw an ellipse from three adjacent kissing-points $kp(t_{i-1}), kp(t_i), kp(t_{i+1})$ as mentioned in Lemma 1. We have proved that the ellipse curve is tangent to $\odot t_i$. MN is the tangent line of the the ellipse curve, passing $kp(t_i)$. Apparently, MN is also the tangent line of $\odot t_i$. Furthermore, $Akp(t_{i-1}) \perp MN$. The point N is the intersection point of the line MN and $Akp(t_{i-1})$.

We prove that $\angle Nkp(t_i)kp(t_{i-1}) = \angle Mkp(t_i)kp(t_{i+1})$ firstly. Suppose the ellipse curve is expressed as Equation 1:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1; \quad (a > b > 0) \tag{1}$$

The coordinate of $kp(t_i)$ is $(a\cos\theta, b\sin\theta)$. Then we can get the line MN denoted by Equation (2):

$$\frac{x}{a}\cos\theta + \frac{y}{b}\sin\theta = 1; \tag{2}$$

The line $kp(t_{i+1})kp(t_i)$ can be expressed as Equation (3):

$$y = \frac{b\sin\theta}{a\cos\theta - a}x + \frac{ab\sin\theta}{a\cos\theta - a}; \tag{3}$$

The line $kp(t_{i-1})kp(t_i)$ can be expressed as Equation (4):

$$y = \frac{b\sin\theta}{a\cos\theta + a}x + \frac{ab\sin\theta}{a\cos\theta + a}; \tag{4}$$

Let k_j denote the slope of the line j. Then we can get Equation (5):

$$\begin{cases} k_{MN} = & \dfrac{-b\cos\theta}{a\sin\theta} \\ k_{kp(t_{i+1})kp(t_i)} = & \dfrac{b\sin\theta}{a\cos\theta - a} \\ k_{kp(t_{i-1})kp(t_i)} = & \dfrac{b\sin\theta}{a\cos\theta + a} \end{cases} \tag{5}$$

According to Equation (5), we can get:

$$\tan\angle Mkp(t_i)kp(t_{i+1}) = \frac{k_{MN} - k_{kp(t_{i+1})kp(t_i)}}{1 + k_{MN} * k_{kp(t_{i+1})kp(t_i)}} = \frac{b}{c\sin\theta};$$

$$\tan\angle Nkp(t_i)kp(t_{i-1}) = \frac{k_{kp(t_{i-1})kp(t_i)} - k_{MN}}{1 + k_{MN} * k_{kp(t_{i-1})kp(t_i)}} = \frac{b}{c\sin\theta};$$

$$\therefore \angle Mkp(t_i)kp(t_{i+1}) = \angle Nkp(t_i)kp(t_{i-1})$$

$$\because t_i kp(t_i) \perp MN$$

$$\therefore \angle Mkp(t_i)kp(t_{i+1}) + \angle\beta = \angle Nkp(t_i)kp(t_{i-1}) + \angle\alpha = \pi/2$$

Therefore, we have $\angle\alpha = \angle\beta$. Theorem 2 is proved. ■

3) Finding the kissing-points: Obviously, when $kp(t_i)$ is on the left of the best position, $\angle\alpha < \angle\beta$. When $kp(t_i)$ is on the right of the best position, $\angle\alpha > \angle\beta$. We propose a novel algorithm, named as Access-Kissing-Points (AKP) algorithm to find the best kissing-points. In AKP, we set the precision of solution as λ_{pre}. AKP can fix $kp(t_i)$ by Theorem 2, if we

get its adjacent kissing-points $kp(t_{i+1}), kp(t_{i-1})$. The details of AKP is shown in the Algorithm 1.

Algorithm 1 Access-Kissing-Points algorithm

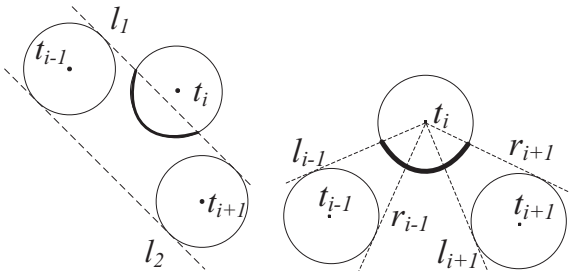
Input: t_i, $kp(t_{i-1})$, $kp(t_{i+1})$, v, r_o, λ', λ_{pre};
Output: The position of $kp(t_i)$;

1: Calculate the minimum distance between the sensor and the targets in each period, $r = r_o - v * \lambda'/2$, and get $\odot t_i$;
2: Calculate the intersection point of the line $t_i kp(t_{i-1})$ and $\odot t_i$, A, and the intersection point of the line $t_i kp(t_{i+1})$ and $\odot t_i$, B;
3: Calculate the mid-point of $\overset{\frown}{AB}$, denoted as O, and assume that O is the kissing-point;
4: If the length of $\overset{\frown}{AB}$ is smaller than λ_{pre}, or $\angle\alpha = \angle\beta$, goto step 7;
5: If $\angle\alpha < \angle\beta$, set O as A and goto step 3;
6: If $\angle\alpha > \angle\beta$, set O as B and goto step 3;
7: **return** The position of O;

Next, we explain the correctness and efficiency of AKP. If we set $\lambda_{pre} = 2^{-k}$, then the complexity of AKP is $O(k \lg r)$ where r is the radius of $\odot t_i$. When we set $k = 100$, the precision will be high enough and the complexity of AKP is low. Thus AKP is effective and efficient.

AKP will be used latter, since getting the accurate position of the adjacent kissing-points is still difficult. However, we can get the the curve where each kissing-point may be. The curve which contains $kp(t_i)$ is denoted by $\overset{\frown}{kp(t_i)}$. Then we gradually prune $\overset{\frown}{kp(t_i)}$ to approach $kp(t_i)$ according to Theorem 2.

At the beginning, we only know that the kissing-point must be a point on $\odot t_i$, namely $\overset{\frown}{kp(t_i)} = \odot t_i$. We prune the curve to get the kissing-point from two aspects:



(a) Situation 1 (b) Situation 2

Fig. 3: Prune the range of the kissing-point

1) The range of the kissing-point can be narrowed as shown in Fig 3.

In situation 1, as shown in Fig 3(a), l_1 and l_2 are the common tangent lines of $\overset{\frown}{kp(t_{i-1})}$ and $\overset{\frown}{kp(t_{i+1})}$. $\overset{\frown}{kp(t_{i-1})}$ and $\overset{\frown}{kp(t_{i+1})}$ are between l_1 and l_2. If $\overset{\frown}{kp(t_i)}$ intersects with l_1 or l_2, $kp(t_i)$ must be between l_1 and l_2 apparently. Thus we can prune $\overset{\frown}{kp(t_i)}$ and set $\overset{\frown}{kp(t_i)}$ as the bold arc, as shown Fig 3(a).

In situation 2, as shown in Fig 3(b), l_{i-1} and r_{i-1} which pass t_i, are the tangent lines of $\overset{\frown}{kp(t_{i-1})}$. l_{i+1} and r_{i+1} which pass t_i, are the tangent lines of $\overset{\frown}{kp(t_{i+1})}$. Choose two tangent lines and make sure that the adjacent targets, t_{i-1}, t_{i+1}, and the segment $t_{i-1}t_{i+1}$ in the angle generated by two chose tangent lines. Apparently, $kp(t_i)$ must be in the angle generated by these two lines. As shown in Fig 3(b), we choose the tangent lines l_{i-1} and r_{i+1} and set $\overset{\frown}{kp(t_i)}$ as the bold arc.

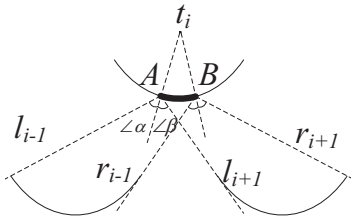


Fig. 4: Illustration for pruning process

2) If we cannot prune $\overset{\frown}{kp(t_i)}$ by the first method, we can continue to prune $\overset{\frown}{kp(t_i)}$ as showed in Fig. 4. l_{i-1}/l_{i+1} is the left tangent line of $\overset{\frown}{kp(t_{i-1})}$/$\overset{\frown}{kp(t_{i+1})}$ which passes a point A in $\overset{\frown}{kp(t_i)}$. Obviously, when A moves to the left, $\angle\alpha$ will become smaller, and $\angle\beta$ will become bigger. According to Theorem 2 we can prove that $kp(t_i)$ cannot be on the left of A, if the line At_i is the bisector of the angle generated by l_{i-1} and l_{i+1}, namely $\angle\alpha = \angle\beta$. Similarly, we can decide the position of B and $kp(t_i)$ cannot be on the right of B. Thus we can prune $\overset{\frown}{kp(t_i)}$ and set $\overset{\frown}{kp(t_i)}$ as the bold arc, as shown Fig 4. We can also easily find that, when $\overset{\frown}{kp(t_{i-1})}$ or $\overset{\frown}{kp(t_{i+1})}$ is pruned, we can continue to prune $\overset{\frown}{kp(t_i)}$.

According to the discussion above, we propose an algorithm, named ROSE, to schedule the route of the sensor. ROSE decides the position of kissing-points and connects the kissing-points one by one. We set two thresholds, th_{time} and th_{range}, in ROSE. th_{time} represents the limit of the number of the prune loops. th_{range} represents the limit of the minimum prune range. The details of ROSE are shown in Algorithm 2. In Algorithm 2, if we want get a better solution, we can execute step 6 and step 7 continuously.

Theorem 3. *The time complexity of ROSE is* $O(n^{\frac{1}{\epsilon^2}})$.

Proof: The time complexity of ROSE is dominated by step 2 and step 4. The time complexity of the approximation algorithm [22] is $O(n^{\frac{1}{\epsilon^2}})$, when the approximation ratio is $O(1+\epsilon)$. The pruning loop is iterated at most th_{time} times, which is a constant. The time complexity of each pruning loop is $O(n)$. Thus the time complexity of ROSE is $O(n^{\frac{1}{\epsilon^2}} + n)$. We set $\epsilon < 1$ usually. Then the time complexity is $O(n^{\frac{1}{\epsilon^2}})$. ■

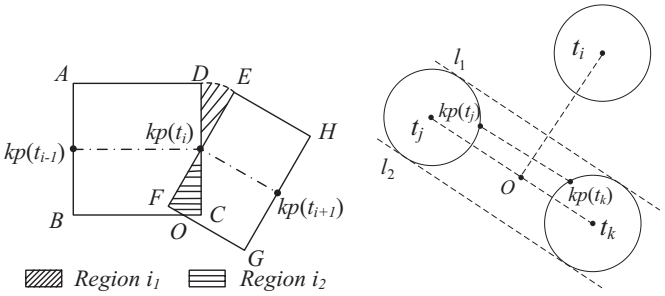
Theorem 4. *The approximation ratio of ROSE is* $3.55 + \epsilon$.

Proof: Assume L^* is the shortest route in the DSRS problem. L_{tsp} is the best solution of the TSP problem. Then we can get that $L^* > L_{tsp} - 2nr$. The total length of the route in ROSE is L. Obviously, $L < (1+\epsilon)L_{tsp}$. Thus the approximation ratio is $L/L^* < (1+\epsilon)L_{tsp}/L^* < (1+\epsilon)(L^* + 2nr)/L^* = 1 + \epsilon + 2nr/L^*$. Furthermore, since no cycles of targets intersects, the route length and the number of targets

Algorithm 2 ROSE for single kissing-point case

Input: v, r_o, λ', T;
Output: The minimum access period, λ;

1: Calculate the minimum distance between the sensor and the targets in each period, $r = r_o - v * \lambda'/2$;
2: Calculate the TSP path of the targets by an approximation algorithm [22];
3: Mark the targets as $t_0, t_1, \cdots, t_{n-1}$ in order according to the TSP path;
4: Prune the range where the kissing-points may be;
5: Set the kissing-points $kp(t_0), kp(t_2), kp(t_4), \cdots$ on the middle of the curve
6: Calculate $kp(t_1), kp(t_3), kp(t_5), \cdots$ by AKP;
7: Update $kp(t_0), kp(t_2), kp(t_4), \cdots$ by AKP;
8: Connect the kissing-points. Then get a route of the sensor. The total length of the route is L; Calculate the access period, $\lambda = L/v$;
9: **return** λ;



(a) Illustration of Theorem 4 (b) Proof of Lemma 3

Fig. 5: Proof of Lemma 3 and Theorem 4

have a relationship. S_r denotes the area which is less than $2r$ from the route. The area of all $\odot t_i (t_i \in T)$ is $n\pi r^2$. Since all targets are covered by the route, we can infer that $S_r \geq n\pi r^2$. Next, we calculate the value of S_r.

As shown in Fig 5(a), $kp(t_{i-1}) \rightarrow kp(t_i) \rightarrow kp(t_{i+1})$ is a part of the route. $kp(t_i)C = kp(t_i)D = kp(t_i)E = \overset{\frown}{kp(t_i)F} = 2r$. $ABCD$ and $EFGH$ are two rectangles. $\overset{\frown}{DE}$ is an arc centred on O. S_{i_1} denotes the area of region i_1 and S_{i_2} denotes the area of region i_2. Obviously, $S_{i_2} > S_{i_1}$. Then we can infer that $S_r = 4rL + \sum_{i=1}^{n}(S_{i_1} - S_{i_2}) < 4rL$.

Then $nr/L < 4/\pi$. Since $L^* < L$, we can get that $1 + \epsilon + 2nr/L^* < 1 + \epsilon + 2nr/L < 3.55 + \epsilon$. Thus the approximation ratio of ROSE is $3.55 + \epsilon$. ■

B. General Case

We proposed another algorithm, called as General and Efficiently Route Scheduling Algorithm (G-ROSE), to solve the DSRS problem in the general case.

Theorem 5. *If a target t_i has more than one kissing-points, $KP(t_i) = \{kp_1(t_i), kp_2(t_i), \cdots, kp_m(t_i)\}$ $(m > 1)$. Then $\forall kp_j(t_i) \in KP(t_i)$, $kp_j(t_i)$ must be on the segment connecting two adjacent kissing-points of other targets. In other words, the targets which have more than one kissing-points have no impact on the route of the mobile sensor.*

Proof: We prove Theorem 5 by contradiction. Assume $kp_j(t_i)$ is not on the segment connecting two adjacent kissing-points of other targets. We can delete $kp_j(t_i)$ from the route of mobile sensor, and set the segment connecting two adjacent kissing-points of other targets as the route. Then we will get a shorter route. Thus the theorem is proved. ■

According to Theorem 5, choose the targets which have only one single kissing-point and apply ROSE to those targets. Then we get the route of the mobile sensor in the general 2-D case. This is the main idea of G-ROSE.

Lemma 3. *If the distance between one target t_i and the line connecting any other two targets is more than $2r$, then the target t_i has only one kissing-point.*

Proof: We prove Lemma 3 by contradiction. As shown in Fig. 5(b), select any other two targets, $t_j, t_k (j \neq i, k \neq i)$. Draw two lines l_1, l_2 that both are tangent to the two circles $\odot t_j$, $\odot t_k$. l_1, l_2 are both parallel to the line t_j, t_k. The distance between the target t_i and $t_j t_k$ is more than $2r$. The distance between the tangent line (l_1 or l_2) and $t_j t_k$ is r. Then the distance between the target t_i and the tangent line is more than r. Thus the circle $\odot t_i$ cannot intersect with $kp(t_j)kp(t_k)$. Suppose the target t_i has more than one kissing-points, then the circle $\odot t_i$ intersects with the segment connecting two kissing-points, which leads to a contradiction. ■

At the beginning, select some targets which have only one kissing-point according to Lemma 3. Apparently, other targets may also have only one kissing-point. We apply ROSE to the selected targets and get a route. Some targets may be far away from the route, and the route cannot intersect the detection circles of these targets. Thus we select the target which is farthest away from the route and update the route. The algorithm will terminate until all targets can be covered.

The detail of G-ROSE is as Algorithm 3.

Algorithm 3 G-ROSE for general 2D case

Input: T, v, r_o, λ';
Output: The minimum access period, λ;

1: Calculate the minimum distance between the sensor and the targets in each period, $r = r_o - v * t'/2$;
2: Choose the targets that have only one kissing-point. If there are not such kind, then choose two targets which have the largest distance;
3: Apply ROSE to the selected targets, and get a route;
4: If some detection cycles do not intersect the route, select the target farthest from the route, and go to step 3;
5: Connect the kissing-points and get a route of which is L. Calculate the access period, $\lambda = L/v$;
6: **return** λ;

Theorem 6. *The time complexity of G-ROSE is $O(n^{1+\frac{1}{\epsilon^2}})$.*

Proof: The time complexity of G-ROSE is dominated by step 3 and step 4. From Section IV-A3, we know the time complexity of ROSE is $O(n^{\frac{1}{\epsilon^2}})$ when $\epsilon < 1$. Step 4 will execute at most $O(n)$ times. Thus the time complexity of G-

ROSE is $O(n^{1+\frac{1}{\epsilon^2}})$. ■

Actually, to decrease the complexity of G-ROSE, we can sacrifice the approximation ratio and apply some other approximation algorithm to decide the order of targets. In our experiment, we adopt the algorithm based on minimum spanning tree to find the solution of TSP.

Theorem 7. *The approximation ratio of G-ROSE is* $1+\rho$.

Proof: Similar with Theorem 4, we can easily get a solution L by G-ROSE such that $L < (1+\epsilon)L_{tsp}$ where L_{tsp} is the best solution of the TSP problem. Since $L^* < L_{tsp} - 2nr$, we can get that $L/L^* < (1+\epsilon)L_{tsp}/(L_{tsp} - 2nr)$. When $\rho = (\epsilon + 2nr/L_{tsp})/(1 - 2nr/L_{tsp})$, we can conclude that the approximation ratio of G-ROSE is $1+\rho$. The theorem is proved. ■

V. Distributed Route Scheduling Algorithm for Large Scale Networks

One mobile sensor is not enough for large scale networks, since the visit cycle can be too long. We need an appropriate amount of sensors to achieve sweep coverage and reduce the length of visit cycle. In this situation, we propose another distributed route scheduling algorithm, called D-ROSE.

D-ROSE is divided into two phases. In the first phase, we divide the surveillance area into several subregions and detect the positions of targets. In the second phase, we cluster targets and generate a closed route for each cluster. Then we allocate appropriate amount of sensors on each route.

A. The First Phase

At the beginning of the first phase, we divide the surveillance area into big square grids, since the scale of networks is too large. We call this division as first division. After first division, we can select a sink node for each big square grid and process each big square girds independently. To get the position of the targets, we make the second division in each big square grid. The big square grid is divided into some square grids. The mobile sensor will move to the grids, so that each grid which overlaps with the surveillance area has one mobile sensor. However, the mobile sensors have no idea about whether any targets locates in the grids. The mobile sensors need to travel along the Peano Curve in the square grids and get the positions of all targets in their grids. The sensors will store the positions of targets in a table and wait communication sensors. The communication sensors will move around the surveillance area and collect the information about the positions of the targets. Then these information are sent to the sink node. Finally, The sink node combines different position information tables into a new table.

B. The Second Phase

In the second phase, the sink node cluster all targets into several groups according to their locations. we propose a novel clustering algorithm to group the targets, named Targets Clustering (TC) algorithm.

In TC, we construct a graph $G = (V, E)$ and group the targets according to G. Each target represent a vertex in V and $E = \varnothing$ initially. Then according to the location of targets, we calculate the distances among all pairs of targets, and store the distances in an array $D[n(n+1)/2]$. Next, We sort the distances in ascending order. From $D[0]$ to $D[n]$ we add the corresponding edges to G. Before $D[i] = e(u, v)$ is added to E, we need to judge whether u and v is in the same connected component. If not, we union the connected component with u and the connected component with v after $D[i]$ is added to E. In this step, we can apply the Union-Find algorithm [23]. When a new connected component is generated, we can look on each connected component as a group. Then we get a new scheme of targets clustering and apply G-ROSE to calculate the minimal number of sensors needed. Finally, all targets are connected in the same group. We compare all clustering scheme that we get, and choose the best clustering scheme with minimal necessary number of sensors. The detail of Targets Clustering algorithm is shown in Algorithm 4.

Algorithm 4 Targets Clustering algorithm

Input: T, v, r_o, λ', t_d;
Output: Targets clustering result;
1: Calculate the distance between t_i, t_j ($\forall t_i, t_j \in T, i \neq j$), and store the distances in an array $D[n(n+1)/2]$;
2: Sort all distances in ascending order such that $D[0] \leq D[1] \leq \cdots \leq D[n(n+1)/2]$;
3: **for** each $i = 0 \rightarrow n(n+1)/2$ **do**
4: **if** $D[i] == e(u, v)$ && find(u)≠find(v) **then**
5: Union(u, v) and generate a new connected component;
6: View each connected component as a group and get a new clustering scheme;
7: Apply G-ROSE to the new generated group and calculate the minimal number of sensors needed in this scheme;
8: **end if**
9: **end for**
10: Select the best clustering scheme with minimal number of sensors needed;
11: **return** The selected clustering scheme;

After we get the clustering scheme and divide the targets into k groups, the sink node will calculate the distance between each sensor and each group. We set the distance between each sensor and each group as the minimum distance between the sensor and the targets in the group. Then we consider each group as a point, and apply Kuhn-Munkres algorithm [24] to decide that which group the sensors move to. Thus we can assign enough mobile sensors to each group. The sink node sends the result of Kuhn-Munkres algorithm to the communication sensors, and the mobile sensors know where they should go through the communication sensors.

The detail of D-ROSE is shown in Algorithm 5.

C. Performance Analysis

Theorem 8. *The time complexity of the sink node is* $O((n+m)nm + n^{2+\frac{1}{\epsilon^2}})$ *where* n *is the number of targets in each big square grid and* m *is the number of mobile sensors*

Algorithm 5 D-ROSE for large scale networks

Input: t_d, T, v, r_o, λ', S, $Region$;
Output: The minimum number of sensors needed;

1: Divide the surveillance area in two rounds and assign one mobile sensor to one grid.
2: Let the mobile sensors travel around the grids. The communication sensors collect the targets position information and send the information to the sink node;
3: The sink node divides the targets into several groups by the Targets Clustering algorithm;
4: The sink node assigns some mobile sensor to one group by Kuhn-Munkres algorithm.
5: For each group, we apply G-ROSE to get a route in each group, and use minimum sensors to sweep the targets;
6: **return** The minimum number of required sensors;

Proof: We invoke G-ROSE for n times in Targets Clustering algorithm. The time complexity of Targets Clustering algorithm is $O(n^{2+\frac{1}{\epsilon^2}})$ where n is the number of targets in each big square grid. The time complexity of Kuhn-Munkres algorithm is $O((n+m)nm)$ where m is the number of mobile sensors in each big square grid. Thus the time complexity of the sink node is $O((n+m)nm + n^{2+\frac{1}{\epsilon^2}})$. ■

To decrease the time complexity of the Targets Clustering algorithm, we can find the minimum number of necessary sensors in each clustering scheme according to the TSP route instead of the route decided by G-ROSE. In this way, the time complexity of TC can be $O(n^2)$ when we find a TSP route by minimum spanning tree.

Theorem 9. *G-ROSE is $(2.5+\gamma)$-approximation if we find the TSP route based on minimum spanning tree.*

Proof: We assume that: In the optimal solution OPT, there are m groups, $G_1, G_2, \cdots, G_n$. In G_i, the length of optimal DSRS route is denoted as $C_i(1 \le i \le m)$. The length of optimal TSP route is denoted as $L_i(1 \le i \le m)$. The length of spanning tree is $S_i(1 \le i \le m)$. Then we can infer that:

$OPT = \sum_{i=1}^{m} \lceil C_i/len \rceil$, where $len = vt_d$.

$C_i \ge L_i - 2n_i r \ge S_i - 2n_i r$, where $1 \le i \le m$ and n_i is the number of targets in G_i.

When we get m groups by the Targets Clustering algorithm, in each group TG_i, the length of spanning tree is $TS_i(1 \le i \le m)$. Then we can easily prove that $\sum_{i=1}^{m} TS_i \le \sum_{i=1}^{m} S_i(1 \le i \le m)$. we can get a TSP route TL_i for each group TG_i in polynomial time such that $TL_i \le 1.5TS_i$ [25]. If the solution we get from the Targets Clustering algorithm is S, we can infer that:

$$
\begin{aligned}
S &\le \sum_{i=1}^{m} \lceil TL_i/len \rceil \le \sum_{i=1}^{m} TL_i/len + m \\
&\le \sum_{i=1}^{m} 1.5TS_i/len + m \le \sum_{i=1}^{m} 1.5S_i/len + m \\
&\le \sum_{i=1}^{m} 1.5L_i/len + m \le \sum_{i=1}^{m} 1.5(C_i + 2n_i r)/len + m \\
&= 1.5OPT + 3nr/len + m
\end{aligned}
$$

Thus we can conclude that $S/OPT \le 1.5 + 3nr/(len \cdot OPT) + m/OPT$. When $3nr/(len \cdot OPT) = \gamma$, the approximation ratio is $2.5 + \gamma$. ■

VI. Performance Evaluation

In this section, we conduct extensive simulation experiments to evaluate our algorithms. Since D-ROSE is based on G-ROSE, we only implement ROSE and G-ROSE. The simulation result is presented via Mathematica. We compare our algorithm with CSWEEP [2], [21] and ISTP [9].

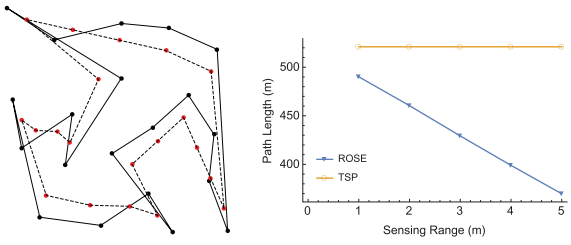
A. Simulation Setup

To test the performance of our algorithms in different situation, we generate different scenarios with different number of targets, and different sensing range. In addition, even our algorithms are applied to the mini-T-problem in this paper, our algorithms can also be applied to the mini-sensor-problem and mini-velocity-problem which are mentioned in section II, when the algorithms are changed a little. Thus we can compare our algorithms with the previous work easily.

Furthermore, we propose another simple algorithm, named as CenRS for comparison with our algorithms. CenRS also considers the impact of sensing range. We design CenRS as follows: Firstly, we calculate the center of all targets; Secondly, close the targets to the center and the displacement distance is r; Finally, apply TSP-algorithm directly and find a solution.

B. ROSE VS TSP-algorithm

Since G-ROSE is based on ROSE, we evaluate ROSE firstly. We implement ROSE by C++ language and compare ROSE and TSP-algorithm. In this experiment, we deal with the targets with more than one kissing-point in a simple way. When $\odot t_i$ is on the line $t_{i-1}t_{i+1}$, we just need to set $kp(t_i)$ as the midpoint of $kp(t_{i-1})kp(t_{i+1})$. We apply ROSE and TSP algorithm to these targets, then we can find two route scheduling solutions. When we set the sensing range as 2, we can get an result as shown in Fig. 6(a). The black points denote the targets and the red points denote the kissing-points. Then We set the sensing range from 1 to 5, experiment 2000 times for different sensing range, and calculate the average length of the route as shown in Fig. 6(b). Apparently, the improvement of ROSE is proportional to the sensing range.



(a) An example when $r = 3$ (b) Average route length with different r

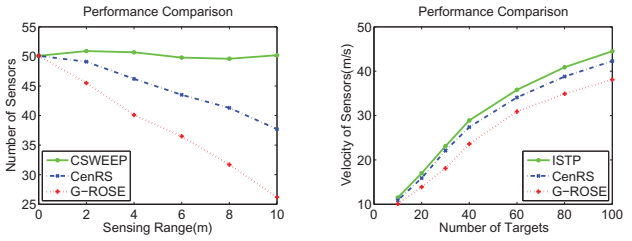
Fig. 6: Comparison between ROSE and TSP algorithm

C. G-ROSE VS CSWEEP

In this subsection, G-ROSE is applied to mini-sensors-problem and we compare among G-ROSE, CenRS and C-SWEEP. We change the sensing range from 0 to 10, and calculate the average number of required sensors. Finally, we get the result as shown in the Fig. 7(a). We find that G-ROSE is more better with larger sensing range. When the sensing range is 10, the performance is improved more than 45%.

D. G-ROSE VS ISTP

In this subsection, G-ROSE is applied to mini-velocity-problem and we compare among G-ROSE, CenRS and ISTP. We change the number of targets from 10 to 100, and calculate the the average minimum velocity needed. Finally, we get the result as shown in the Fig. 7(b). We find that when the number of targets becomes larger, the improvement is more noticeable.



(a) Under the same region with different sensing range

(b) Under the same region with different target density

Fig. 7: Performance Comparison

In summary, all the above experiments show that G-ROSE has a better performance than the other algorithms especially with large sensing range and great density of targets.

VII. Conclusion

In this paper, we consider the impact of sensing range in sweep coverage problems to shorten the trajectory length of mobile sensors. Firstly, we consider a restricted version with single kissing-point constraint for each target and propose an approximation algorithm, named as ROSE. We prove that ROSE has an approximation ratio of $3.55 + \epsilon$. Based on ROSE, we proposed another algorithm, named as G-ROSE, for the general case without any restriction. We further design a distributed sweep algorithm, named as D-ROSE to solve sweep coverage problem with multiple sensors. The simulation results show that G-ROSE outperforms the previous works especially with large sensing range and great density of targets. In all, we are the first to consider the sensing range effect (as well as communication range) in route scheduling, which could greatly reduce the trajectory length for mobile sensors.

References

[1] W. Bang, L. H. Beng, and M. Di, "A survey of movement strategies for improving network coverage in wireless sensor networks," *Computer Communications (COMCOM)*, vol. 32, no. 13, pp. 1427–1436, 2009.

[2] W. Cheng, Z. Li, K. Liu, Y. Liu, X. Li, and X. Liao, "Sweep coverage with mobile sensors," in *IEEE International Symposium on Parallel and Distributed Processing (SPDP)*, 2008, pp. 1–9.

[3] J. Du, Y. Li, H. Liu, and K. Sha, "On sweep coverage with minimum mobile sensors," in *IEEE International Conference on Parallel and Distributed Systems (ICDPS)*, 2010, pp. 283–290.

[4] R. Moazzez-Estanjini and I. C. Paschalidis, "Improved delay-minimized data harvesting with mobile elements in wireless sensor networks," in *International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (SMOAW)*, 2011, pp. 49–54.

[5] R. Moazzez-Estanjini and C. Paschalidis, I, "On delay-minimized data harvesting with mobile elements in wireless sensor networks," *Ad Hoc Networks*, vol. 10, no. 7, pp. 1191–1203, 2012.

[6] W. Zhao, M. Ammar, and E. Zegura, "Controlling the mobility of multiple data transport ferries in a delay-tolerant network," in *IEEE Conference of the IEEE Computer and Communications Societies (CCCS)*, 2005, pp. 1407 – 1418.

[7] B. Tariq, M. Mukarram, M. Ammar, and E. Zegura, "Message ferry route design for sparse ad hoc networks with mobile nodes," in *IEEE International Symposium on Mobile Ad Hoc Networking and Computing (SMANC)*, 2006, pp. 37–48.

[8] M. Xi, K. Wu, Y. Qi, J. Zhao, Y. Liu, and Z. Li, "Run to potential: Sweep coverage in wireless sensor networks," in *International Conference on Parallel Processing (ICPP)*, 2009, pp. 50–57.

[9] D. Zhao, H. Ma, and L. Liu, "Mobile sensor scheduling for timely sweep coverage," in *IEEE Conference on Wireless Communications and Networking (CWCN)*, 2012, pp. 1771–1776.

[10] B. Gorain and P. S. Mandal, "Point and area sweep coverage in wireless sensor networks," in *International Symposium on Modeling & Optimization in Mobile, Ad Hoc & Wireless Networks (SMOMAW)*, 2013, pp. 140–145.

[11] G. Barun and M. Partha, Sarathi, "Approximation algorithms for sweep coverage in wireless sensor networks," *Journal of Parallel and Distributed Computing (JPDC)*, vol. 74, pp. 2699–2707, 2014.

[12] S. Li, W. Wei, L. Feng, L. Zhonghao, and Z. Jiliu, "A sweep coverage scheme based on vehicle routing problem," *Journal of Electrical Engineering (JEE)*, vol. 11, no. 4, pp. 2029–2036, 2013.

[13] L. Shu, K. Cheng, X. Zhang, and J. Zhou, "Periodic sweep coverage scheme based on periodic vehicle routing problem," *Journal of Networks*, vol. 9, no. 3, pp. 726–732, 2014.

[14] J. Roh and S. Jin, "Device control protocol using mobile phone," in *International Conference on Advanced Communication Technology (CACT)*, 2014, pp. 355–359.

[15] O. Tiwari, Shiv and S. Kumar Yadav, "Data harvesting with mobile elements in wireless sensor networks," *Journal of Electronics and Communication Engineering (JECE)*, vol. 9, pp. 104–114, 2014.

[16] Y. Gu, D. Bozda, R. W. Brewer, and E. Ekici, "Data harvesting with mobile elements in wireless sensor networks," *Computer Networks (CN)*, vol. 50, no. 17, pp. 3449–3465, 2006.

[17] R. Sugihara and R. Gupta, "Optimizing energy-latency trade-off in sensor networks with controlled mobility," in *IEEE International Conference on Computer Communications (ICCC)*, April 2009, pp. 2566–2570.

[18] R. Sugihara and K. Gupta, Rajesh, "Optimal speed control of mobile node for data collection in sensor networks," *IEEE Transactions on Mobile Computing (TMC)*, vol. 9, no. 1, pp. 127–139, 2010.

[19] A. Somasundara, Arun, A. Ramamoorthy, and B. Srivastava, Mani, "Mobile element scheduling with dynamic deadlines," *IEEE Transactions on Mobile Computing (TMC)*, vol. 6, no. 4, pp. 395–410, 2007.

[20] A. A. Somasundara, A. Ramamoorthy, and M. B. Srivastava, "Mobile element scheduling for efficient data collection in wireless sensor networks with dynamic deadlines," in *IEEE Conference on Real-time Systems Symposium (CRSS)*, 2004, pp. 296–305.

[21] M. Li, W. Cheng, K. Liu, Y. He, X. Li, and X. Liao, "Sweep coverage with mobile sensors," *IEEE Transactions on Mobile Computing (TMC)*, vol. 10, no. 11, pp. 1534–1545, 2011.

[22] A. Sanjeev, "Polynomial time approximation schemes for euclidean tsp and other geometric problems," in *Journal of the ACM*, 1996, pp. 2–11.

[23] T. H. Cormen and C. E. Leiserson, "Introduction to algorithms, second edition," *Mit Press*, 2001.

[24] H. Zhu, M. Zhou, and R. Alkins, "Group role assignment via a kuhn-munkres algorithm-based solution," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans (TSMC)*, vol. 42, no. 3, pp. 739–750, May 2012.

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