Trajectory Segment Selection with Limited Budget in Mobile Crowd Sensing

Yueyue Chen[a], Pin Lv[b,*], Deke Guo[c], Tongqing Zhou[a], Ming Xu[a]

[a] *College of Computer, National University of Defense Technology, Changsha, China*
[b] *School of Computer Electronics and Information, Guangxi University, Nanning, China*
[c] *College of Information System and Management, National University of Defense Technology, Changsha, China*

Abstract

Mobile Crowd Sensing is an emerging paradigm, in which a large number of participants are involved to complete a sensing task under a certain incentive mechanism. Hence, when the budget used to pay participants is limited, how to choose the most appropriate participants becomes a critical problem. Most of existing works aim to select a subset of participants to maximize the coverage, without considering redundancy. There are two kinds of redundancy in the existing literature, one is brought by the incomplete coverage assessment, while the other one is brought by the traditional participant selection process. Since paying for redundant data leads to budget waste, existing works can not solve the participant selection problem commendably under limited budget. To address such issues, we first propose a coverage assessment considering both uniform coverage and maximum coverage, then design a trajectory segment selection scheme. Rather than choosing the whole trajectory of a participant, our scheme selects certain segments. Both offline and online algorithms are proposed in this paper. Two benchmarks are implemented and we carry out extensive experiments based on a real dataset. The evaluation results prove the effectiveness and the advantage of our algorithms in terms of the coverage quality.

Keywords: mobile crowd sensing, segment selection, coverage

1. Introduction

Smart devices, such as mobile phones, smart vehicles, wearable devices and so on, have entered a new prosperous period. Besides their communication functions, a rich set of embedded sensors on these smart devices enable new applications across a wide variety of domains, including indoor localization [1] [2], environmental monitoring [3] and social networks [4]. With the more and more powerful sensing and computing capacities of the smart devices, Mobile Crowd Sensing (MCS) has become a new paradigm that has attracted much attention [5]. MCS enables large-scale information collection and crowd intelligence extraction with the data contributed by ordinary users. Those data can be sensed or generated from their smart devices and then aggregated in the cloud [6].

As demonstrated in Fig. 1, there are usually three parts in a MCS framework: task publishers, MCS servers, and users with smart devices [7]. The MCS servers are responsible for data collection, processing and providing service for users. The numerous users can be either data source or data consumer, which means users can provide data for computing center as well as request data service from it. To get enough sensing data and mine more significant information, the MCS servers usually release sensing tasks to public. Ordinary users are encouraged to

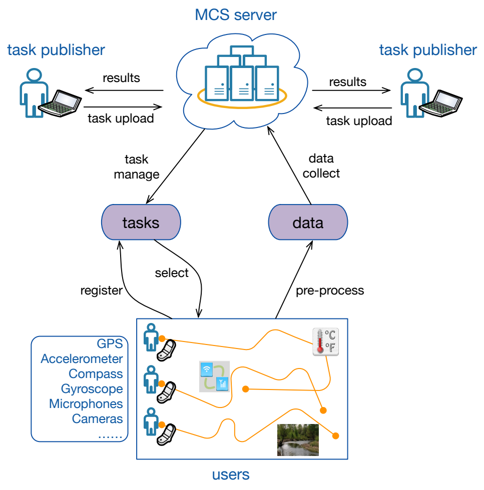


Figure 1: The framework of mobile crowd sensing

involve in the tasks and upload their data to the MCS servers. In this paradigm, the MCS server is referred to as a recruiter, and the users interested in the tasks are called participants.

In order to encourage participants to make effective contribution for sensing tasks, a recruiter usually utilizes some incentive mechanisms as reward for the participants [8][9]. However, since the budget of a task is usually limited, the recruiter prefers paying participants who provide higher quality data, rather than paying all of them. Hence, how to achieve a tradeoff between limited budget and crowd sensing quality becomes a challenge. To choose high quality data from all participants is quite crit-

*Corresponding author
Email addresses: yueyuechen@nudt.edu.cn (Yueyue Chen), lvpin@gxu.edu.cn (Pin Lv), dekeguo@nudt.edu.cn (Deke Guo), zhoutongqing@nudt.edu.cn (Tongqing Zhou), xuming@nudt.edu.cn (Ming Xu)

ical to solve the problem. The crowd sensing quality includes many factors, and a very important one is the spatial coverage [10]. The location information is a fundamental tag of participants and tasks, and different tasks may require different types of coverage. For instance, some tasks need a full coverage of interested area, others may need a credible coverage of the POIs (Points Of Interest). However, most of existing works aim to select a subset of participants to maximize the coverage, without considering redundancy. There are two kinds of redundancy in the existing literatures, one is brought by the incomplete coverage assessment, while the other one is brought by the traditional participant selection process.

On one hand, most of related works focus on how to gain the maximum coverage of interested area within the budget [11][12], and only a few researches pay attention to how to gain better spatial representative data with limited money [13]. For example, the distributions of participants in urban area and suburb are different, thus it is important to get both urban and suburb sensing data. Hence, uniform coverage should be considered in the participant selection process, while in fact did not. The difference between uniform coverage and maximum coverage can be seen in Fig. 2. With the same budget, which means the same number of participants should be selected. Fig. 2(a) is the selection results under uniform coverage, while Fig. 2(b) is that under maximum coverage. Maximum coverage algorithms prioritize the number of acquired samples over the variance of the acquired sample set. They only consider the sample number and ignore the relationship between samples. For instance, there usually exhibit high similarity between two samples with smaller distance, and it is more easier to deduce one of the two samples from the other. In this scenario, the similarity between any pair of samples is redundancy in some way. To reduce such redundancy, collected samples should be different enough with each other under limited budget.

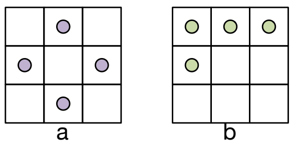


Figure 2: The difference between uniform coverage and maximum coverage with the same limited budget

On the other hand, existing recruitment approaches usually focus on how to select participants. Participant selection scheme can not guarantee removing such trajectory overlaps between different participants. An example is shown in Fig. 3. Suppose we want to monitor the environment condition of a city. After issuing the scope of monitored area to the crowd, three participants are willing to execute the task. Suppose the trajectories of participants can be predicted, which can be achieved by many ways, such as navigation [12], periodic movement [14], or just uploaded by participant themselves. The three participants move along the same road at first, and then turn to different directions. Suppose every trajectory can be divided into two parts, as shown in Fig. 3, and the cost of each part is 1. If the budget is 4, which means we can only recruit two of them, data redundancy can not be avoided no matter which two are chosen, because the spacial coverage of the first half of their trajectories are almost the same. Hence, the prior participant selection methods remain inapplicable due to the data redundancy.

To remove data redundancy and improve the coverage quality, we propose to select different trajectory segments of different participants to payback, instead of recruiting a subset of participants. For instance, we can choose the whole trajectory (A_1 *and* A_2) of participant A, the B_2 part trajectory of participant B, and the C_2 part trajectory of participant C. The total cost is also 4, which is equal to that of choosing two participants, yet more representative samples across different regions are collected. Segment selection algorithm can encourage participants to sense those sparsely-populated areas. Because there are more data redundancy in populous areas and it is not easy to get payback among so many competitors. Hence, people will tend to taking sensing tasks in underpopulated areas for more rewards.

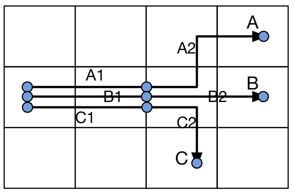


Figure 3: The data redundancy of participant selection

Our contributions are summarized as follows:

- We quantify two aspects of coverage quality criteria, including the coverage percentage and the uniform degree, and formulate the trajectory segment selection problem based on predicted trajectories in the MCS framework.
- We propose a trajectory segment selection scheme aiming at maximizing the coverage quality, both offline and online selection algorithms are discussed.
- We evaluate our algorithm based on a real dataset and compare our algorithms with other two popular selection algorithms, including a greedy participant selection algorithm and a genetic segment selection algorithm. The results show that our algorithms outperform existing works.

The rest of this paper is organized as follows. In Section 2, the system model of the trajectory segment selection problem is described and formulated. In Section 3, offline and online segment selection algorithms are developed respectively to solve the formulated problem. Two benchmarks are introduced in Section 4. Experiment results and detailed explains are demonstrated in Section 5. Section 6 lists related works, and the paper is concluded in Section 7.

2. Problem Formulation

The objective of this paper is to achieve high coverage quality in the MCS framework under the constraint of limited budget.

To achieve this goal, the MCS server needs to select the most representative samples among the area of interest to get the maximum benefits. We propose to select appropriate trajectory segments of different participants rather than simply choose a subset of participants. The system model and the Trajectory Segment Selection (TSS) problem are described in the following subsections.

2.1. System model

Suppose we want to monitor the temperature of a city. Given the sensing task, we assume that there are in total m participants, who are interested in the task, denoted as a vector $V=\{v_1, \cdots, v_m\}$. The sensing task needs to be conducted for a period of time T. The period T can be further expressed as a vector $T=\{t_1, ..., t_n\}$, where t_i $(1 \leq i \leq n)$ is a specific moment within T. The partition granularity of T can be adjusted according to various sensing requirements. In each time slice, a sampling is conducted at a certain moment $\tau \in [t_i, t_{i+1}]$ $(i \in [1, n-1])$ to represent the sensing result during this time slice. The system model is illustrated in Figure 4 (b), and Figure 4 (a) is the illustration of related trajectories.

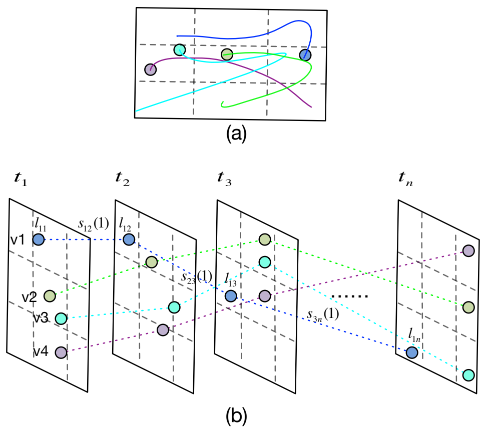


Figure 4: Illustration of the system model

Definition 1: (Trajectory Matrix) Based on the participant vector V and the time vector T, participant trajectories can be formulated as an $m \times n$ matrix:

$$L = \begin{pmatrix} l_{11} & l_{12} & \cdots & l_{1n} \\ l_{21} & l_{22} & \cdots & l_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ l_{m1} & l_{m2} & \cdots & l_{mn} \end{pmatrix}, \tag{1}$$

where l_{ki} represents the location of a participant v_k at t_i, $k \in [1, m]$ and $i \in [1, n]$. Each row vector indicates the trajectory of one participant during the task period, and each column vector shows the different locations of different participants at a specific time.

Definition 2: (Trajectory Set) With the trajectory matrix aforementioned, the trajectory of a participant is divided into multiple segments in accordance of the time slices. The trajectory segment of participant v_k during time slice $[t_i, t_j]$ can be denoted by $s_{ij}(k)$. Hence, the whole set of predicted trajectories can be denoted as:

$$\bigcup_{k,i} s_{i(i+1)}(k), \tag{2}$$

where $k \in [1, m]$, $i \in [1, n-1]$.

Definition 3: (Segment Cost) The payment for participants from recruiter is based on the sensing cost of smart devices. We assume the sensing cost is determined by the length of a trajectory segment. Consider that participants usually vary in their speeds, thus the lengths of trajectory segments for different participants during the same time slice are not equal. Thus, the cost for participant v_k during the period $[t_i, t_j]$ can be computed as:

$$c(s_{ij}(k)) = |s_{ij}(k)| \times \Delta c \tag{3}$$

where $|s_{ij}(k)|$ represents the length of segment $s_{ij}(k)$, and Δc is the unit cost.

Definition 4: (Constraint) The main constraint of a sensing task is that the total cost of selected segments can not exceed the limited budget. Suppose the total budget is denoted as B, and the set of segments selected is denoted as S'. Therefore, the constraint can be computed as:

$$C = \sum_k \sum_i c(s_{i(i+1)}(k)) \leq B, \;\; s_{i(i+1)}(k) \in S' \tag{4}$$

2.2. Coverage assessment

In order to select proper participants to execute the sensing task, we need to estimate the sensing quality of participants. As discussed in the introduction section, the coverage constraint is an important factor. Therefore, we measure two coverage metrics in this subsection: the coverage percentage and the uniform degree.

Since the total time period is divided into multiple time slices as shown in Figure 4, the coverage percentage of each time slice is computed. The average coverage percentage of all time slices is set as the average coverage percentage of the participants, as discussed in Definition 5. The average uniform degree is obtained in the same way, which is discussed in Definition 6.

To compute the two coverage metrics, the trajectory matrix L introduced in Definition 1 is transformed into the coverage matrix H. Suppose the sensing area is divided into multiple equal boxes, $A=\{A_1, \cdots, A_q\}$. S' is the selected segments, and L' is the related location matrix of selected segments. If the location l_{ki} in the trajectory matrix L is within a box A_x, the box is defined to be covered by the segment. According to the element l_{kj}, the relevant box number is determined, and the elements of the box matrix is A_x $(1<x<q)$.

$$H = \begin{pmatrix} h_{11} & h_{12} & \cdots & h_{1n} \\ h_{21} & h_{22} & \cdots & h_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ h_{m1} & h_{m2} & \cdots & h_{mn} \end{pmatrix}, \tag{5}$$

Definition 5: (Average Coverage Percentage) The coverage percentage of each time slice indicates the ratio of the covered boxes by selected segments to the total number of boxes. Based on the defined coverage matrix, the coverage percentage in a time slice t_i can be computed according to the following equation:

$$R_{t_i} = \frac{|\bigcup_{v_k \in S'} h_{kt_i}|}{q} \quad (6)$$

Hence, the average coverage percentage ($\mathrm{E_{cp}}$) of all selected segments among all time slices can be computed as:

$$\mathrm{E_{cp}} = \frac{\sum_{i=1}^{n} \mathrm{R_{t_i}}}{\mathrm{n}} \quad (7)$$

Definition 6: (Average Uniform Degree) The $\mathrm{E_{cp}}$ metric illustrates the number of nodes inside an interested area, yet ignores the relative spatial distribution of those nodes. To describe the spatial distribution of selected segments, another coverage metric is introduced, i.e., the uniform degree. Jaimes et al. formulate the uniform degree as variance of samples [13]. Ji et al. use the entropy-based method to illustrate the uniform degree [15]. However, neither of the two methods considers the relative spatial location among participants.

2D entropy [16] is widely used in the field of image segmentation, because it illustrates the relationship of one square and its neighborhood. In this paper, we use 2D entropy to describe the uniform degree of participant distribution at one time slice. The number of participants in a square is considered as the gray value. Hence, the uniform degree a segments in a time slice t_i can be computed as follows:

$$\begin{aligned} U_{t_i} &= -\sum_a \sum_b p_{ab}(t_i)\log_2(p_{ab}(t_i)) \\ p_{ab}(t_i) &= \frac{f(a,b,t_i)}{q} \end{aligned} \quad (8)$$

where a is the gray value of a square, b is the gray value of its neighborhood, and $f(a, b, t_i)$ is the repetition number of 2-tuples (a, b) at t_i time slice. Hence, $p_{ab}(t_i)$ is the ratio between the occurrence times of 2-tuples (a, b) and the overall subarea number q at t_i time slice.

Therefore, the average uniform degree ($\mathrm{E_{ud}}$) can be formulated as:

$$\mathrm{E_{ud}} = \frac{\sum_{i=1}^{n} \mathrm{U_{t_i}}}{\mathrm{n}} \quad (9)$$

Definition 7: (Coverage Assessment) The final coverage assessment should consider both the coverage percentage and the uniform degree. In this paper, the objective is to maximize the uniform degree while make sure the coverage percentage. Hence, the coverage assessment (Γ) is calculated as the following equation:

$$\Gamma = \mathrm{E_{cp}} + \delta \times \mathrm{E_{ud}}, \quad (10)$$

where δ is a weight coefficient to indicate the importance of uniform degree. In practice, some applications may only focus on the coverage percentage, while others may emphasize both requirements.

The complexity of the coverage assessment computation is analyzed as follows. The computation complexity of computing the coverage matrix from the location matrix is $O(m)$, since there are m participants. Computing the average coverage percentage costs $O(n \times m)$. The computation of uniform degree in each time slice can be divided into the following three steps. The first step is to compute the participant number in each square, which costs $O(m \times q)$. The second step is to count $f(a, b)$, which costs $O(max(a) \times max(b))$. The third step is to calculate the 2D entropy, which costs $O(max(a) \times max(b))$ too. Therefore, the computation complexity of computing the average uniform degree is $O(min(n \times m \times q, n \times max(a) \times max(b)))$, where $max(a) \times max(b) \leq m \times m$.

2.3. The TSS problem formulation

Based on the previous definitions and assumptions, the trajectory segment selection (TSS) problem is formalized as follows: Given an MCS task with q squares and n time slices, and a trajectory set S, how to select a subset of participants S' to maximize the coverage assessment, while satisfying the constraint of limited budget. In addition, in order to obtain as much sensing data as possible, the budget should be made full use of, which means the budget surplus should be less than a required threshold.

$$\begin{aligned} &\max \quad \Gamma = \frac{\sum_{i=1}^{n} R_{t_i}}{q \times n} + \delta \times \frac{\sum_{i=1}^{n} U_{t_i}}{n} \\ &s.t. \\ &\quad 0 \leq B - \sum_k \sum_i c(s_{i(i+1)}(k)) \leq min\ surplus \\ &\quad s_{i(i+1)}(k) \in S' \\ &\quad i \in [1, n-1] \\ &\quad k \in [1, m] \end{aligned} \quad (11)$$

We can prove that the TSS problem is an NP-complete problem, and the proof is given in the appendix.

Since it is difficult to find an exact solution in a polynomial time for an NP-complete problem, approximate algorithms are designed. Both greedy algorithm and genetic algorithm are often used to solve NP-complete problem. As the location relationship among participants affects the computation results of Γ, hence the greedy algorithm is unsuitable for TSS problem. Furthermore, the genetic algorithm is a kind of Monte Carlo method, thus it usually can not make full use of the budget. The TSS problem considers both the number of coverage squares and the square distribution. Therefore, it is a challenge to propose a proper selection algorithm to solve the TSS problem. In the next section, a reverse greedy algorithm will be introduced to solve the TSS problem.

3. Backward Segment Selection Algorithm

In this section, a reverse greedy algorithm called Backward Segment Selection Algorithm (B-SSA) is proposed. The objective of the algorithm is to recruit a subset S' of trajectory segments based on the given parameters S, C, B, and maximize the coverage quality.

Algorithm 1 Trimming redundancy segments (S, C, B)

```
1:  $S' = S$; $d = 0$;
2:  $C = \sum_{k}^{m} \sum_{i}^{n} c(s_{i(i+1)}(k))$;
3:  repeat
4:      if $d < d_{max}$ then
5:          $d = d + \Delta d$;
6:      end if
7:      for $t, k$ do
8:          if $D(s_{t(t+1)}(k), s_{t(t+1)}(k+1)) \leq d$ then
9:              $S' = S' - s_{t(t+1)}(k)$;
10:             $C = C - c(s_{t(t+1)}(k))$;
11:         end if
12:     end for
13: until $C \leq B$
```

3.1. Basic idea

The basic idea of the backward segment selection algorithm is to prune away the redundant segments. The redundancy means that two trajectory segments in one time slice is too close to each other. Hence, one of them should be cut off. We suppose that one of them will be trimmed if the distance between two segments is less than a threshold d. The threshold d is determined according to specific requirements, which reflects the sensing granularity. A small value of d means that a large set of segments are preserved and a great deal of sensing data can be collected; on the contrary, a larger d indicates less sensing information. After a round of trimming, the total cost of the remained trajectory segments will be computed and compared with the budget. If the cost exceeds the budget, the sensing distance will be enlarged, and a new round of trimming will be conducted. The algorithm will not stop until the budget B is exhausted. It is revealed that, if the budget is enough, the recruiter can set a small d and recruit more segments. In this way, a tradeoff between d and B is achieved. The participants will be informed which of their trajectory segments are recruited, and the sensing samples within selected segments will be adopted.

Algorithm 1 reports the details about the selection method. The output S' is the set of selected segments. The time complexity of Algorithm 1 is $O(m^2n)$, which is polynomial.

The distance between two trajectory segments can be measured by multiple methods as mentioned in [17]. Since the participant trajectories are well mapped through the location matrix in Definition 1. If the trajectories are winding, they can be mapped in a more fine-grained manner by narrowing the temporal gap between time slices. The partition granularity can be determined according to different application requirements. Therefore, the segment distance is formulated as the Euclidean distance of their endpoints $D(l_{(k1)i}, l_{(k2)i})$. If the distance of the endpoints of two segments are both less than d, the distance between the two segments is less than d.

$$D(s_{ij}(k1), s_{ij}(k2)) = \begin{cases} \leq d & D(l_{(k1)i}, l_{(k2)i}) \leq d, D(l_{(k1)j}, l_{(k2)j}) \leq d \\ > d & else \end{cases} \tag{12}$$

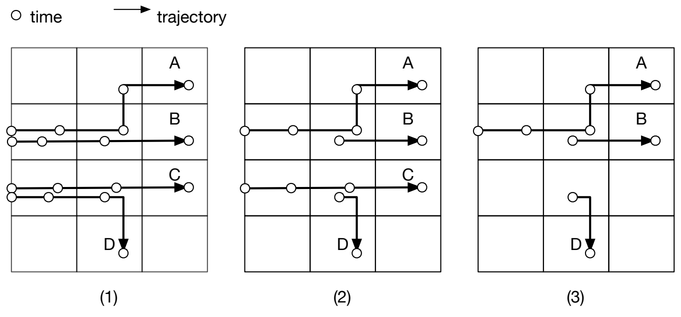


Figure 5: An example of trajectory segment selection algorithm

To ease the understand of Algorithm 1, an example is demonstrated in Fig. 5. In Fig. 5(1), the trajectories of four participants A, B, C, D are shown. Each trajectory is divided into multiple segments according to time slices. The area of interest are divided into multiple squares, and the side length of one square is e. Suppose $\Delta d = e/2$. Fig. 5(2) is the result of the first-round trimming. Because the trajectory distance between A and B in the first two time slices is less than $d = \Delta d$, a part of trajectory of B is expunged. In the same way, D is expunged. Actually, the result of first trimming is reasonable enough, because selected trajectory segments distribute dispersedly, and no data redundancy exist. However, if the total cost still exceeds the budget, a second round of trimming is needed, and $d = 2 \times \Delta d$. The trimming result is shown in Fig. 5(3), and more trajectory segments are expunged. Those segments remained in Fig. 5(3) are the segments finally being selected.

3.2. Offline backward segment selection algorithm

Suppose we can get all the participant trajectories in advance by many different ways, such as prediction based on navigation information, or uploaded by participants themselves. According to the global trajectory information, an offline segment selection algorithm based on the aforementioned basic idea is designed.

The design value of Δd should be large at first for efficient trimming, and be reduced as the increasing number of iterations to find the optimal solution. Hence, we set $\Delta d = \lambda/r$, where r is the number of iteration rounds and λ is a constant number. As r becomes larger and larger, which means more and more iterations have been executed, Δd becomes smaller and smaller. In this way, the offline algorithm can get a threshold d in a progressive method.

While the distance between two segments is less than the threshold, one of them should be trimmed. A random selection of the trimmed one is adopted in the basic algorithm. Actually, one selection can influence the next selection in the algorithm. For example, if there are three segments A, B, C. The distance between A and B is the same with that between B and C. If A is trimmed in a random selection process, B or C will be trimmed, too. Hence, only one segment will be left. However, if B is trimmed at first, A and C will be both left, since the distance between them is bigger than the threshold. Therefore, a more efficient selection method should be designed in each iteration.

Algorithm 2 Offline B-SSA (S, C, B)

```
1: $S' = S$; $d = 0$;
2: $C = \sum_{s_{i(i+1)}(k) \in S'} c(s_{i(i+1)}(k))$;
3: repeat
4:     if $d < d_{max}$ then
5:         $d = d + \frac{\lambda}{r}$;
6:     end if
7:     for $t, k$ do
8:         if $D(s_{t(t+1)}(k), s_{t(t+1)}(k+1)) \leq d$ then
9:             $S1' = S' - s_{t(t+1)}(k)$;
10:            $S2' = S' - s_{t(t+1)}(k+1)$;
11:            $E1 = (U_t(S1') + U_{t+1}(S1'))/2$
12:            $E2 = (U_t(S2') + U_{t+1}(S2'))/2$;
13:            if $E1 > E2$ then
14:                $S' = S1'$;
15:                $C = C - c(s_{t(t+1)}(k))$;
16:            else
17:                $S' = S2'$;
18:                $C = C - c(s_{t(t+1)}(k+1))$;
19:            end if
20:        end if
21:    end for
22: until $C \leq B$
```

To maximize the uniform degree of selected segments, 2D entropy should be computed in each iteration, the distribution with smaller entropy should be given up, and the related segment should be trimmed. However, computing the average 2D entropy among all time slices in each iteration is a time-consuming task. To speed up the computation process, as well as to select a better segment, only average 2D entropy of the related two time slices is computed. The offline segment selection algorithm (offline B-SSA) can be seen in Algorithm 2.

3.3. Online backward segment selection algorithm

In some cases, it is not able to predict all the trajectories accurately, because participants are dynamic and they may change their plans. To make the selection process more efficient, an online segment selection algorithm is proposed. When participants move dynamically, we adjust the prediction according to historical selection information. In this process, there are two kinds of trajectories, historical trajectories and predicted trajectories. Based on the historical trajectories of one participant, we can predict the trajectory at next time slice based on different ways. We do not discuss much about the prediction method which is beyond the research purpose. We just suppose the next segment can be predicted with a high probability.

As participant trajectories appear dynamically, the MCS server should determine selecting or not a piece of coming trajectory segment in real time. In our backward segment selection algorithm, the basic idea is to select trajectory segment according to the distance between segments. If the distance between two segments are less than a threshold, one of them should be trimmed. Hence, when new segments come, the distance between them and historical segments should be computed and compared with the threshold. In the offline algorithm, the threshold can be determined by multiple iterations. However, in the online situation, there is no chance to try multiple times and determine a threshold. Therefore, a dynamic threshold is used in the online selection algorithm.

Algorithm 3 Online B-SSA (S,C,B)

```
1: Separate samples into $\varphi$ parts;
2: $x = -2 \sim 2$;
3: for $j < \varphi$ do
4:     $max = 0$;
5:     for $x$ do
6:         $\delta_{temp} = 10^x$;
7:         execute offline B-SSA on $\varphi_{i \neq j}$ parts;
8:         if $\frac{\Gamma_{BSSA} - \Gamma_{benchmarks}}{\Gamma_{benchmarks}} >$ max then
9:             max=$\frac{\Gamma_{BSSA} - \Gamma_{benchmarks}}{\Gamma_{benchmarks}}$;
10:            $\delta = \delta_{temp}$;
11:        end if
12:    end for
13:    execute offline B-SSA on $\varphi_j$ part;
14:    if $\delta$ makes $\Gamma$ gain of $\varphi_j$ the biggest then
15:        efficiency($\delta$)= +1;
16:    end if
17: end for
18: select the $\delta$ with most efficiency;
19: while new segment do
20:    $d_{avr} = \frac{1}{t_{i-1}} \sum_{1}^{t_{i-1}} d$;
21:    $B_r = \frac{1}{budget} \sum_{1}^{t_{i-1}} b$;
22:    if $\frac{E_{cp}}{E_{opt}} > B_r$ then
23:        $d = d_{avr} \times (1 - \frac{\epsilon}{t_i})$;
24:    else
25:        $d = d_{avr} \times (1 + \frac{\epsilon}{t_i})$;
26:    end if
27:    $D$=min distance between new and selected segments;
28:    if $D \leq d$ then
29:        select the segment;
30:    else
31:        pass the segment and wait new coming;
32:    end if
33: end while
```

In the process of task, some participants may leave and new participants may join. Hence, the number of coming segments in each time slices is different. The budget is divided into multiple parts according to the number of time slices. The number of budget in each part is determined by the coverage percentage of selected segments and the average threshold of historical selection. When a new segment comes, the average threshold d_{avr}, ratio of the average coverage percentage and optimal coverage percentage computed by offline algorithm $\frac{E_{cp}}{E_{opt}}$ and budget ratio B_r are computed. If $\frac{E_{cp}}{E_{opt}} > B_r$, which means the budget part in this time slice is less than the average number. Hence, the threshold d should be decreased and more segments should be selected in the future. In contrast, if $\frac{E_{cp}}{E_{opt}}<B_r$, the threshold should be enlarged. The coverage percentage instead of the uniform degree is used in this selection process. Because the uniform degree is a global parameter and not suitable with online situation.

There is another problem that, although the B-SSA can solve the TSS problem with a given parameter, it can not determine the value of the parameter δ. To get an appropriate parameter δ in the coverage assessment formulation, we use the Leave-one-out, a popular method, to adjust parameters in Machine Learning. Suppose there are n time slices, we can separate these slices into multiple parts and divide the budget into multiple parts too. In each iteration, execute the offline segment selection algorithm on all the parts except one of them, and select the value from given parameter range that can maximize the objective function gain, where $\Gamma_{gain} = (\Gamma_{BSSA} - \Gamma_{benchmarks})/\Gamma_{benchmarks}$. We test if the selected parameter is efficient in the left part; if not, we say that the selected parameter makes a mistake. Execute the iteration on every part, and select the value which makes the least mistakes as the value of parameter δ.

Algorithm 3 reports the details about our online selection algorithm.

4. Benchmarks

To measure the efficiency of the proposed methods, two benchmarks are set up. One is the greedy participant selection algorithm, and the other is the genetic segment selection algorithm. These two kinds of algorithms are common to efficiently search optimal solutions of NP-hard problems.

4.1. Greedy participant selection algorithm

The greedy participant selection algorithm (greedy-PSA) is proposed by He et al. [12]. The objective of this algorithm is to maximize the average coverage percentage through selecting participants. Because the greedy selection method can only search a local optimal solution, the uniform degree is a global parameter. Hence, we use the greedy participant selection algorithm (greedy-PSA) to search the optimal solution for the coverage percentage maximizing problem, and test the uniform degree of the solution.

The main idea of the greedy-PSA is that, there are two sets of participants, one is selected set and the other is unselected set. In each selection iteration, select a new participant randomly from the unselected set, and add it into the selected set. Compute the objective function after each selection iteration. Always select the participant who can get the largest objective function, which is the ratio of the E_{cp} to the total cost of selected participants in this algorithm. Repeat the selection process until the budget is exhausted.

According to the selection process, in each iteration, we have to traverse every participants in the unselected set. In the worst case, there are m selection iterations in total. Hence, the complexity is $O(m^2 \times n)$ which is faster than our segment selection algorithm, the reason is that we redefine the objective granularity and the time complexity of 2D entropy computation is high.

Algorithm 4 Greedy-PSA (H,C,B)

```
1:  $S' = \emptyset$; $C = 0$; $CP = 0$;
2:  repeat
3:      for $m$ do
4:          $S_{temp} = S'$; $C_{temp} = C$;
5:          $max = 0$; $maxID = 0$;
6:          $S_{temp} = S_{temp} + v_i$; $C_{temp} = C_{temp} + c(s_{1n}(i))$;
7:          $CP = \mathrm{E_{cp}}(\mathrm{Equation}[7])$;
8:          if $\frac{CP}{C_{temp}} > max$ then
9:              $max = \frac{CP}{C_{temp}}$; $maxID = i$;
10:         end if
11:     end for
12:     $S' = S' + v_{maxID}$;
13: until $C \geq B$
14: if $C > B$ then
15:     $S' = S' - v_{maxID}$;
16: end if
```

The detailed greedy selection process is illustrated in Algorithm 4. The input is the coverage matrix and the budget defined in Section 2. The output is a subset of participants. By comparing the greedy-PSA and our backward-SSA, the redundancy is evaluated by adopting participant selection instead of the segment selection method, which is discussed in Section 5.

4.2. Genetic segment selection algorithm

Genetic algorithm is a kind of heuristic method. The main idea of genetic algorithm is to imitate the heredity and evolution. In the TSS problem, a segment selection solution can be represented by a chromosome easily. A segment is associated with 1 if it is selected, and 0 if not. Hence, a $m \times n$-dimensional vector consisting of multiple 1s and 0s, which is the concept of a chromosome in the genetic algorithm, can represent a segment selection solution. The objective of this algorithm is to find the best chromosome in all generations. The best chromosome should maximize the objective function, which is Γ defined in Section 2.

The genetic algorithm can be processed by the following steps. The initial generation of chromosomes are generated randomly. The next generation can be made through crossover or mutation. Crossover means that a pair of “parent” chromosomes exchange some of their genes randomly to produce “child” chromosomes. Mutation means that some of genes are altered to create a new individual. In each generation, the one whose cost over than budget will be discarded to make sure the constraint is active. To simulate the evolution, the top $\omega\%$ chromosomes of each generation will survive and generate the next generation. After predefined evolution time, the one with the biggest *coverage assessment* in the last generation will be selected as the final selection decision.

5. Performance Evaluation

In this section, we evaluate our B-SSA based on a real dataset, and compare the results with that of the benchmarks.

Algorithm 5 Genetic-SSA (S,C,B)

```
1: $Num = 0$;
2: generate the initial population randomly;
3: **repeat**
4:     **for** each chromosome **do**
5:         **if** cost>B **then**
6:             discard the chromosome;
7:         **end if**
8:         compute $\Gamma$;
9:     **end for**
10:    sort based on $\Gamma$ and select top $\omega\%$ chromosomes;
11:    $Num = Num + 1$;
12:    generate next generation through crossover or mutation;
13: **until** Num=predefined repeat number
14: return the chromosome with biggest $\Gamma$;
```

5.1. Experiment setup

In this paper, the participant trajectories are supposed to be predicted and obtained. The experiments are carried out based on a real dataset. The dataset is downloaded from the Crowdad website [18], which is a popular website with large numbers of datasets. The dataset is the mobility traces of taxi cabs in Rome, Italy. It contains GPS coordinates of approximately 320 taxis collected over 30 days. The test of offline B-SSA and benchmarks which are also offline methods is based on data of one day, since the integrated dataset is too large and it is too costing to compute all the data. The number of taxis in that day is 158, and the size of the sensing region is $63.0364km * 66.2210km$. As the location information of different participants is not time-aligned, we first formulate the time grid and regard the time within k seconds as the same time. Based on the time and location data, the location matrix L is obtained, in which each element is a pair of longitude and latitude of a participant at a specific time.

To obtain fair comparison results, the greedy participant selection algorithm (greedy-PSA) and genetic segment selection algorithm (genetic-SSA) are also implemented for comparison with the proposed backward segment selection algorithms.

5.2. Coverage assessment comparison

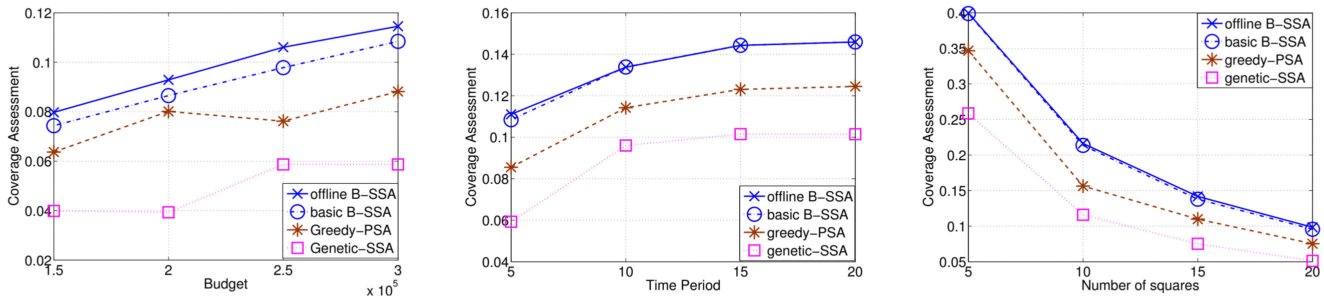
We compare the coverage assessment of four algorithms with different settings. The first algorithm is basic B-SSA introduced in Section 3.1, the second algorithm is offline B-SSA introduced in Section 3.2, the difference of these two algorithms is whether or not compute the uniform degree in each selection iteration. The latter is much costing as well as accurate. The third and fourth algorithm is the greedy-PSA and genetic-SSA introduced in Section 4, respectively.

There are several adjustable parameters in our model, such as the spatial and temporal partition granularity q, n, the budget B, and the selection parameter λ. These parameters can be determined according to different application requirements. To compare the results of four algorithms comprehensively, we varied these parameters in the evaluation, and the results can be seen in Figure 6.

Firstly, Fig. 6(a) illustrates the coverage assessment comparison of the four algorithms with varied budgets. The X axis is the budget B and the Y axis is the computing results of Γ presented in section 2. In this evaluation, the $q = 18 * 18$, the $n = 24h/5s$ and the $\lambda = 1\ km$. It can be seen from Fig. 6(a) that, the backward SSA and offline B-SSA increases with the increment of the given budget, while that of the greedy-PSA and genetic-SSA fluctuate. He et al. implement a greedy participant selection algorithm and a genetic participant selection algorithm in [12], the results are also fluctuated in their evaluation, which are in accord with our simulation. In this situation with the settings of other parameters, the performance of the offline B-SSA is better than basic B-SSA, although the advantage of the former is small. However, the results of the offline B-SSA is much better than greedy-PSA and genetic-SSA. The most increment can be as high as 37% than greedy-PSA when the budget is 250000, and 137% than genetic-SSA when the budget is 200000.The least increment is 18.7% than greedy-PSA when the budget is 200000, and 100% than genetic-PSA when the budget is 250000. By the way, the total cost of all the segments is nearly 500000.

Secondly, Fig. 6(b) illustrates the coverage assessment comparison of the four algorithms with varied temporal partition granularities. The X axis is granularities $k = (t_{i+1} - t_i)$. For example, $k = 10$ means that the GPS locations of participants are sampled every ten seconds. The Y axis is the coverage assessment. In this simulation, other parameters were assigned values as $B = 250000$, $\lambda = 0.25km$, $q = 18 * 18$. It can be concluded from Fig. 6(b) that, the results of the offline B-SSA and basic B-SSA are almost the same. The reason is that the λ is small enough, hence the selection granularity is refined. It can be inspired from the results that, to get refined results, we can take the offline B-SSA or take a small λ for basic B-SSA. The increments from B-SSA than the two benchmarks are stable with varied time period, the values are about 17% and 43% respectively.

Thirdly, Fig. 6(c) illustrates the coverage assessment comparison of the four algorithms with varied spatial partition granularities. The X axis is the square numbers. For instance, $x = 10$ means there are totally $10 * 10$ squares in the interested area. In this simulation, $B = 250000$, $k = 5s$, $\lambda = 0.25km$. All the coverage assessment of the four algorithms decrease while the square number becomes larger. The reason is that the budget is fixed, hence the total segments selected are fixed. Therefore, when the square number increases, the ratio of selected squares and total squares dropped. It is shown in Fig. 6(c) that B-SSA performs at most 67% better than greedy-PSA and 100% better than genetic-SSA when there are $20 * 20$ squares. However, when there are only 25 squares, the increment is only 14% and 60% respectively. That is because, our objective function is based on the square number. If there is only one square, it is very easy for all the four algorithms to get the largest results. Hence, the less the square number is, the closer the results of different algorithms are. However, it is common sense that the result is more accurate when there are more squares.



(a) Varied budget (b) Varied time period (c) Varied square number

Figure 6: Coverage assessment comparison between backward-SSA and benchmarks

5.3. *Coverage percentage comparison*

The coverage percentage comparison between backward-SSA and greedy-PSA indicates that the segment selection method outperforms the participant selection method. The comparison results are illustrated in Fig. 7, in which the basic B-SSA shows an obvious advantage.

In addition, the execution time of two algorithms are also compared. Because the genetic-SSA is a random method, its execution time is much less than other algorithms. Besides, the offline B-SSA is more complex due to the 2D entropy computation in each iteration. The process of backward-SSA and greedy-PSA is similar, therefore we compare their execution time. From the results, which is illustrated in Fig. 8, we can see that the execute time of greedy-PSA is less when there are little budget, and that of backward-SSA is less when the budget is large.

5.4. *Cost comparison*

With a given budget, four algorithms choose different segments or participants. Therefore, the sums of cost with the four algorithms are not exactly the same and fluctuating centered the budget. The total cost sum of all participants in dataset is $4.9177 * 10^5$. Part of the cost sum of selected segments or participants with given budget are listed in Table 1. It can be seen that the cost of offline B-SSA is a little larger than basic B-SSA, that is because the former selects the segment with better uniform degree instead of less costing. The genetic-SSA wastes much of the budget.

5.5. *Evaluation of online B-SSA*

To evaluate the online B-SSA, some inaccurate data is introduced to the original dataset to simulate the inaccuracy of trajectory prediction. The prediction accuracy is getting higher with more real data is used.

Figure 9 illustrates the results of one leave-one-out process on one part of history data. The X axis indicates different value of δ, $\delta = 10^x$. The Y axis is the coverage assessment (Γ). Backward-SSA always performs better than benchmarks no matter what δ is. According to the online selection method introduced in Section 2, the one with most Γ gain is selected as the value of δ. After computing, δ is determined as 0.1.

The online selection process is based on the distance threshold d, and d is based on different application requirements. In our online selection algorithm, the threshold in the first time slice is determined by experience, which means the results of offline selection algorithm in this paper. Actually, the results of the online selection process is affected deeply by the choose of threshold at first. Hence, we evaluate the impact of different threshold in this part, and compare the results with that of the offline backward segment selection algorithm.

Figure 10 illustrates the results of the online selection process with different thresholds and offline selection process on the same dataset. In this experiment, we set $threshold\ in\ the\ first\ time\ slice = d \times 2 \times minBudget/maxBudget$, and d is illustrated in the figure. From the compare results, we can see that offline algorithm performs better than online algorithm. With proper distance threshold, the online algorithm can reach the coverage percentage results of offline algorithm. However, the uniform degree results between online algorithm and offline algorithm is bigger, that is because the online algorithm can not select segments in a global view.

6. Related Work

6.1. *Participant recruitment*

Participant recruitment denotes that selecting proper participant subset based on request of a recruiter among all the participants. Some recruitment mechanisms are based on continuous connectional network [10] , and some are based on opportunistic network [19] [20]. The process of identifying suitable participants is usually performed considering parameters such as availability, expertise, performance, context, etc. [21]. And the communication cost between participants and the MCS server should be considered in opportunistic network [22]. Some participant recruitment methods are related with incentive mechanisms [23] [24].

The availability of participants usually in terms of spatial and temporal coverage, which will be discussed in the next subsection. The performance of participants usually includes reputation, time delay, and data accuracy. Performance based recruitment has been used in many crowdsourcing platforms, such as

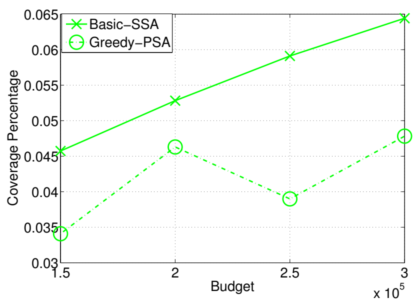


Figure 7: R of basic B-SSA and greedy-PSA

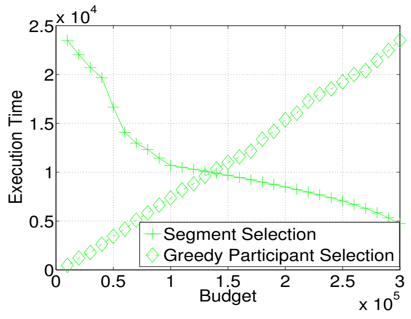


Figure 8: Execution time of basic B-SSA and greedy-PSA

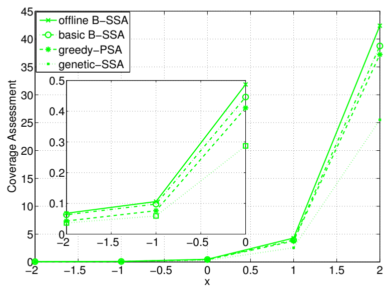


Figure 9: Γ of algorithms with varied δ

Table 1: Cost sum of four algorithms with given budget

Budget(10^4)	5	10	15	20	25	30
offline B-SSA (10^4)	4.6402	9.8775	13.488	16.346	21.464	29.817
basic B-SSA (10^4)	4.8611	9.5540	13.470	15.783	18.689	23.939
greedy-PSA (10^4)	4.9969	9.9814	14.995	19.978	24.984	29.982
genetic-SSA(10^4)	2.9696	5.8071	11.510	11.426	21.465	21.291

Amazon Mechanical Turk and so on. How to establish reputation assessment mechanism is a research hotspot. The expertise of participants usually combines with social tags, and can be get in social networks. The context of participants usually means the circumstance around users. A participant selection scheme was proposed in [25] considering the social attributes, task delay and reputation.

6.2. Coverage problem

The coverage problem has been researched in many other areas, such as Unmanned Aerial Vehicle (UAV) networks [26], robotics [27], and so on. However, the mobility of participants in the MCS framework is different with UAV or robotics. Smart devices are under the control of users instead of control center, which means recruiters can not totally control or predict the trajectory of participants, although some researches reveal that humans experience a combination of periodic movement that is geographically limited and random jumps correlated with their social networks [14]. We classify existing coverage problem researches in the MCS framework into two kinds: only current location concerned algorithm and mobility concerned algorithm.

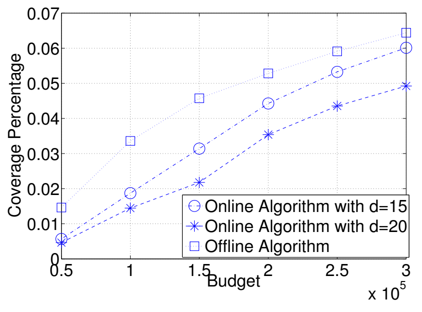
The only current location concerned algorithms. Jaimes et al. [11] formulate the sensing scope of a user as a disk centered at the user, and weighted as the number of users covered by this disk. Two greedy algorithms are used to find users union which can achieve the maximum coverage with limited budget. The maximum coverage in this paper means the maximum weight instead of geographical coverage. A reverse auction incentive mechanism is used to negotiate between users and recruiters. Jaimes et al. further concern getting better representative samples in [13] by selecting the farthest samples away from selected samples. Density map is proposed in [28], where the current estimations of variable density is concerned to determine the sample location and the number of users.

Obviously, ignoring the mobility of participants is very unrealistic, because mobility is an important feature of the MCS system. However, the methodologies and algorithms in those papers are instructive.

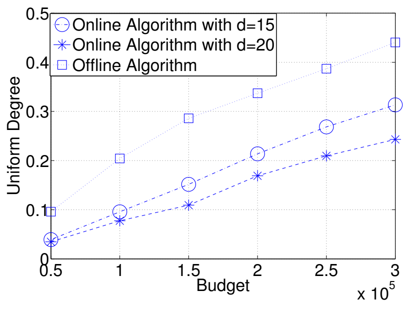
The mobility concerned algorithms. Reddy et al. [10] propose a framework for participants recruitment and pointed out that the coverage based recruitment problem can be formulated into an NP-hard problem, and the greedy algorithm can be used to find an adequate solution when costs are identical. However, the problem is not fully discussed in [10]. He et al. [12] separate the area of interested into numbered regions, and formulate the trajectory of vehicles into a matrix, whose elements are regions covered by one user in a period of time. A greedy algorithm is used for maximum spatial coverage problem, and a genetic algorithm is used for minimum temporal coverage. Zhao et al. [29] separate the task time into multiple period of time, and further separate a period time into multiple sample time. The number of coverage repetition of a region by different users at different time is used as the criterion. A greedy algorithm is used to find the participants subset which can gain the best coverage under the criterion.

However, none of these papers concern about getting better representative data among the area of interest, and none of them focus on segment selection.

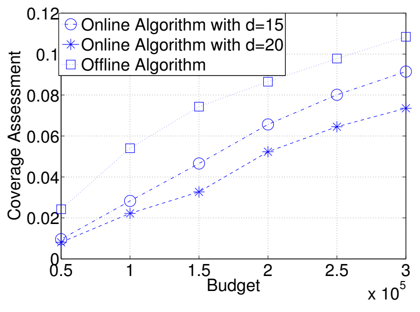
Hamid et al. [30] present a trajectory recruitment scheme to choose the minimum number of vehicles that achieve a required level of a road. The basic greedy algorithm is adapted, and two cases are discussed. The author only focused on one-dimensional trajectory recruitment problem. In this paper, we consider the two-dimensional trajectory segment recruitment problem with limited budget.



(a) coverage percentage



(b) uniform degree



(c) coverage assessment

Figure 10: Comparison between online algorithm and offline algorithm

7. Conclusions

In this paper, a trajectory segment selection algorithm in the MCS framework was presented, which selected different trajectory segments of participants instead of selecting participants to execute a mobile crowd sensing task. Two parameters, i.e., the percentage and the uniform degree were considered to construct the objective function. Two benchmarks, including the greedy participant selection algorithm and the genetic segment selection algorithm were implemented to compare with our backward segment selection algorithm. Plentiful and contrastive experiments were implemented in this paper, and the simulation results proved that our proposed algorithm get a better Γ under varied parameters. The least increment was 17%, and the largest increment was as high as 137%.

We also proposed a basic online selection algorithm based on the spatial distance, which needed more discussion in our future research. As it is hard to predict participant trajectories accurately, online selection with participants moving dynamically was more practical. Hence, more efficient and practical online selection algorithms will be designed in our future research.

Appendix

NP-comleteness proof of TSS problem

We use Θ to represent the trajectory segment selection (TSS) problem.

Θ: Given an interested area A, time period T and a participant set V, where A is divided into q squares, T is divided into n slices. Each participant has a trajectory segment set $S(k) \in S$, and each trajectory segment in $S(k)$ is associated with two squares which are in two time slices separately and cost $C(k)$. The optimization objective is to find a subset of S to maximize the *Coverage Assessment* Γ within budget B.

$$\Gamma = \frac{\sum_{i=1}^{n} |\bigcup_{k} h_{kt_i}|}{q * n} + \delta * \frac{1}{n} * \sum_{i=1}^{n} (-\sum_{a} \sum_{b} p_{ab} \log_2(p_{ab}))$$

where the first part is the average coverage percentage and the second part is the average uniform degree.

(1) Given a subset $S' \in S$, it can be determined in polynomial time whether S' is a solution of Θ. Thus, based on the definition of NP, it can be concluded that $\Theta \in$ NP.

(2) The notation Θ^* is used to denote the classical *k Maximum Coverage* (KMC) problem.

Θ^*: Given a universal set $E = \{u_1, u_2, \cdots, u_n\}$, a collection of subset of the universe $S = \{S_1, S_2, \cdots, S_m\}$, where any $S_i \in S$ satisfies $S_i \subseteq E$. The optimization objective is to find $S' \subseteq S$ maximizing the number of covered items, and $|S'| = K$.

We reduce the KMC problem to our TSS problem as follows. $E = \{A_{11}, A_{12}, \cdots, A_{1q}, A_{21}, A_{22}, \cdots, A_{2q}, \cdots, A_{nq}\}$, where A_{ij} is the j_{th} square at i_{th} time slice. S_i is the related squares set associated with the segment set $S(i)$ in the TSS problem. $c_{i(i+1)}(k) = 1$, hence $K = B$. $\delta = 0$.

The problem Θ^* is an NP-hard problem [31], and Θ^* is a special case of Θ where the cost of each segment is set to 1, and the parameter δ is set to 0. Hence, $\Theta \in$ NP-hard.

Since both $\Theta \in$ NP and $\Theta \in$ NP-hard hold, it can be proven that $\Theta \in$ NP-complete.

Acknowledgment

The work is supported in part by the National Natural Science Foundation of China (NSFC) under Grant Nos. 61402513, 61402494, 61379144, 61422214, and 61501482, Guangxi Natural Science Foundation under Grant Nos. 2015GXNSFBA139243 and 2016GXNSFBA380182, National Natural Science Foundation for Outstanding Excellent Young Scholars of China under Grant 122661422214, in part by the National Basic Research Program (973 program) under Grant 2014CB347800, and the Scientific Research Foundation of Guangxi University under Grant Nos. XGZ150322 and XGZ141182. The authors would like to thank these sponsors for their generous support.

References

[1] X. Teng, D. Guo, Y. Guo, X. Zhou, Z. Ding, Z. Liu, Ionavi: An indoor-outdoor navigation service via mobile crowdsensing, ACM Transactions on Sensor Networks (TOSN) 13 (2) (2017) 12.

[2] X. Zhou, D. Guo, X. Teng, Magspider: a localization-free approach for constructing global indoor map for navigation purpose, Proceedings of the ACM Turing 50th Celebration Conference-China (2017) 44.

[3] L. Wang, D. Zhang, A. Pathak, C. Chen, H. Xiong, D. Yang, Y. Wang, Ccs-ta: quality-guaranteed online task allocation in compressive crowdsensing, Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (2015) 683–694.

[4] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, A. T. Campbell, A survey of mobile phone sensing, Communications Magazine, IEEE 48 (9) (2010) 140–150.

[5] H. Ma, D. Zhao, P. Yuan, Opportunities in mobile crowd sensing, IEEE Communications Magazine 52 (8) (2014) 29–35.

[6] B. Guo, Z. Wang, Z. Yu, Y. Wang, N. Y. Yen, R. Huang, X. Zhou, Mobile crowd sensing and computing: The review of an emerging human-powered sensing paradigm, ACM Computing Surveys (CSUR) 48 (1) (2015) 7.

[7] R. K. Ganti, F. Ye, H. Lei, Mobile crowdsensing: current state and future challenges., Communications Magazine, IEEE 49 (11) (2011) 32–39.

[8] H. Gao, C. H. Liu, W. Wang, J. Zhao, Z. Song, X. Su, J. Crowcroft, K. K. Leung, A survey of incentive mechanisms for participatory sensing, IEEE Communications Surveys & Tutorials 17 (2) (2015) 918–943.

[9] D. Zhao, X. Y. Li, H. Ma, Budget-feasible online incentive mechanisms for crowdsourcing tasks truthfully, IEEE/ACM Transactions on Networking 24 (2) (2016) 647–661. arXiv:1404.2399.

[10] S. Reddy, D. Estrin, M. Srivastava, Recruitment framework for participatory sensing data collections, Pervasive Computing (2010) 138–155.

[11] L. G. Jaimes, I. Vergara-Laurens, M. A. Labrador, A location-based incentive mechanism for participatory sensing systems with budget constraints, Pervasive Computing and Communications (PerCom), 2012 IEEE International Conference on (2012) 103–108.

[12] Z. He, J. Cao, X. Liu, High quality participant recruitment in vehicle-based crowdsourcing using predictable mobility, Computer Communications (INFOCOM), 2015 IEEE Conference on (2015) 2542–2550.

[13] L. G. Jaimes, I. Vergara-Laurens, A. Chakeri, Spread, a crowd sensing incentive mechanism to acquire better representative samples, Pervasive Computing and Communications Workshops (PERCOM Workshops), 2014 IEEE International Conference on (2014) 92–97.

[14] E. Cho, S. A. Myers, J. Leskovec, Friendship and mobility: user movement in location-based social networks, Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining (2011) 1082–1090.

[15] S. Ji, Y. Zheng, T. Li, Urban sensing based on human mobility, Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (2016) 1040–1051.

[16] J. Fan, R. Wang, L. Zhang, D. Xing, F. Gan, Image sequence segmentation based on 2d temporal entropic thresholding, Pattern Recognition Letters 17 (10) (1996) 1101–1107.

[17] Y. Zheng, Trajectory Data Mining : An Overview, ACM Trans. Intell. Syst. Technol. 6 (3) (2015) 1–41.

[18] L. Bracciale, M. Bonola, P. Loreti, G. Bianchi, R. Amici, A. Rabuffi, CRAWDAD dataset roma/taxi (v. 2014-07-17), Downloaded from http://crawdad.org/roma/taxi/20140717.

[19] M. Karaliopoulos, O. Telelis, I. Koutsopoulos, User recruitment for mobile crowdsensing over opportunistic networks, Computer Communications (INFOCOM), 2015 IEEE Conference on (2015) 2254–2262.

[20] H. Zhou, J. Chen, H. Zheng, J. Wu, Energy efficiency and contact opportunities tradeoff in opportunistic mobile networks, IEEE Transactions on Vehicular Technology 65 (5) (2016) 3723–3734.

[21] H. Amintoosi, S. S. Kanhere, A trust-based recruitment framework for multi-hop social participatory sensing, Distributed Computing in Sensor Systems (DCOSS), 2013 IEEE International Conference on (2013) 266–273.

[22] H. Zhou, S. Xu, D. Ren, C. Huang, H. Zhang, Analysis of event-driven warning message propagation in vehicular ad hoc networks, Ad Hoc Networks 55 (2017) 87–96.

[23] Z. Song, B. Zhang, C. H. Liu, A. V. Vasilakos, J. Ma, W. Wang, Qoi-aware energy-efficient participant selection, in: Sensing, Communication, and Networking (SECON), 2014 Eleventh Annual IEEE International Conference on, IEEE, 2014, pp. 248–256.

[24] C. H. Liu, J. Fan, P. Hui, J. Wu, K. K. Leung, Toward qoi and energy efficiency in participatory crowdsourcing, IEEE Transactions on Vehicular Technology 64 (10) (2015) 4684–4700.

[25] J. Ren, Y. Zhang, K. Zhang, X. S. Shen, Sacrm: Social aware crowdsourcing with reputation management in mobile sensing, Computer Communications 65 (2015) 55–65.

[26] Y. Chen, H. Zhang, M. Xu, The coverage problem in UAV network: A survey, Computing, Communication and Networking Technologies (ICCCNT), 2014 International Conference on (2014) 1–5.

[27] E. Galceran, M. Carreras, A survey on coverage path planning for robotics, Robotics and Autonomous Systems 61 (12) (2013) 1258–1276.

[28] D. Mendez, M. A. Labrador, Density maps: Determining where to sample in participatory sensing systems, Mobile, Ubiquitous, and Intelligent Computing (MUSIC), 2012 Third FTRA International Conference on (2012) 35–40.

[29] D. Zhao, H. Ma, L. Liu, Energy-efficient opportunistic coverage for people-centric urban sensing, Wireless networks 20 (6) (2014) 1461–1476.

[30] S. A. Hamid, G. Takahara, H. S. Hassanein, On the recruitment of smart vehicles for urban sensing, Global Communications Conference (GLOBECOM), 2013 IEEE (2013) 36–41.

[31] J. Fan, M. Lu, B. C. Ooi, W. C. Tan, M. Zhang, A hybrid machine-crowdsourcing system for matching web tables, Proceedings - International Conference on Data Engineering (2014) 976–987.