sensors

MDPI

Article

Coverage-Balancing User Selection in Mobile Crowd Sensing with Budget Constraint

Yanan Wang [1], Guodong Sun [1,*] and Xingjian Ding [2]

[1] School of Information Science and Technology, Beijing Forestry University, Beijing 100083, China; wangyanan@bjfu.edu.cn

[2] Department of Computer Science, Renmin University, Beijing 100072, China; dxj@ruc.edu.cn

* Correspondence: sungd@bjfu.edu.cn; Tel.: +86-106-233-8374

Received: 2 April 2019; Accepted: 16 May 2019; Published: 23 May 2019

Abstract: Mobile crowd sensing (MCS) is a new computing paradigm for the internet of things, and it is widely accepted as a powerful means to achieve urban-scale sensing and data collection. In the MCS campaign, the smart mobilephone users can detect their surrounding environments with their on-phone sensors and return the sensing data to the MCS organizer. In this paper, we focus on the coverage-balancing user selection (CBUS) problem with a budget constraint. Solving the CBUS problem aims to select a proper subset of users such that their sensing coverage is as large and balancing as possible, yet without violating the budget specified by the MCS campaign. We first propose a novel coverage balance-based sensing utility model, which effectively captures the joint requirement of the MCS requester for coverage area and coverage balance. We then formally define the CBUS problem under the proposed sensing utility model. Because of the NP-hardness of the CBUS problem, we design a heuristic-based algorithm, called MIA, which tactfully employs the maximum independent set model to determine a preliminary subset of users from all the available users and then adjusts this user subset to improve the budget implementation. MIA also includes a fast approach to calculating the area of the union coverage with any complicated boundaries, which is also applicable to any MCS scenarios that are set up with the coverage area-based sensing utility. The extensive numeric experiments show the efficacy of our designs both in coverage balance and in the total coverage area.

Keywords: mobile crowd sensing; coverage area; coverage balance; user selection

1. Introduction

With the proliferation of the smartphones with multiple built-in sensors, recent years have seen more and more mobile crowd sensing (MCS) applications [1]. MCS is a novel technology to achieve urban-scale monitoring and has been applied in traffic monitoring, public safety, and information sharing [2–5]. A typical MCS campaign, as shown in Figure 1, involves three major components: a requester, an MCS platform, and multiple mobilephone users. The requester publishes some sensing tasks to the MCS platform, in order to obtain some valuable observations on her region of interest (RoI). Since each user can only cover a small fraction of the RoI (i.e., user's mobilephone is limited in sensing range or coverage), the platform has to recruit multiple users to collaboratively cover the given RoI. Finally, the users who are selected to participate in the MCS campaign return their sensing data to the platform. In general, the requester or the platform needs to pay the selected users for their sensing data.

From the point of view of the requester, the sensing utility offered by the platform depends on how well or in which extent her RoI can be covered [6–11]. If her RoI is fully covered, then she can obtain the observation on any spot of her RoI. In real-life MCS scenarios, however, the requester or the platform cannot afford any high charge from the users; in other words, an MCS campaign often comes up with a budget constraint. Thus, one critical issue facing by the MCS user recruitment is to achieve a desirable tradeoff between the sensing utility it can offer to the requester and the payment for users' sensing data.

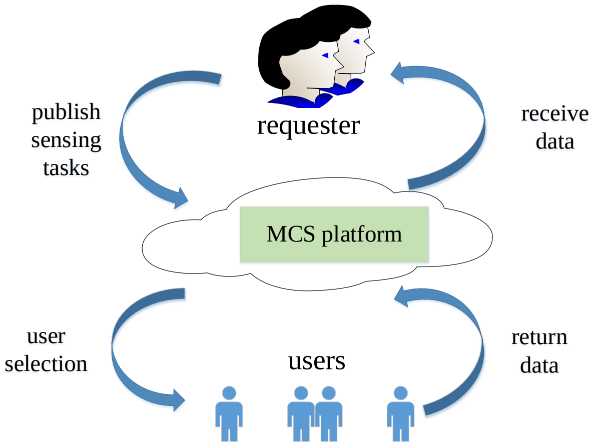


Figure 1. Architecture of typical mobile crowd sensing (MCS) campaigns.

A few recent works on MCS user selection have modeled the sensing utility with the sensing coverage and attempted to maximize the sensing utility or minimize the overall cost paid to the selected users. To measure the MCS sensing utility, these previous works usually take into account the total sensing area that the selected users can cover; that is, the larger area those users can cover, the higher sensing utility the requester can obtain. However, such a coverage area-based sensing utility model cannot comprehensively reflect the requester's requirement—one missing component in evaluating the MCS sensing utility is the geographical balance of sensing coverage that the selected users can offer. In addition to the overall coverage area, we often have to take into account the coverage balance in many typical MCS and wireless sensor network scenarios [12–14], especially when the budget is very limited so that the platform only can recruit a handful of users to perform sensing task. For instance, some MCS applications employ compressive sensing or entropy-based approaches to achieve an accurate overall estimation on each sub-area of the given RoI, even with less users who sparsely distribute in the RoI. For such applications, the more balancing the overall sensing coverage, the lower the entropy that the sensing data can lead to, which can also benefit the sparse signal recovery that is highly needed by the compressive sensing. In general, for two MCS user selections that can achieve the same coverage area, the one with the better coverage balance is always more preferable.

In this paper we concentrate on the coverage-balancing user selection (CBUS) problem in the MCS with a budget constraint, and then design effective algorithm to solve this problem. The major contributions of this paper are as follows. To the best of our knowledge, this study is the first work introducing the coverage balance into the MCS sensing utility measurement. We formally define the CBUS problem, which is NP-hard; and we design a heuristic-based algorithm, called MIA, to solve the CBUS problem. MIA also involves a general approach to fast calculating the union sensing coverage with any complicated boundary; this approach could be complementary to solving any user recruitment problems that are defined under the coverage area-based sensing utility model. Finally, we conduct numeric experiments with a wide range of settings, and the results show the efficacy and efficiency of our designs.

The remainder of this paper is structured as follows. Section 2 presents the MCS system model and defines the CBUS problem. Section 3 details the design of MIA. Section 4 compares our designs with two baseline algorithms. Section 5 briefly reviews major research works related to ours. Finally, Section 6 concludes this paper.

2. Assumptions and Models

2.1. System Model of MCS

We assume that the region of interest (RoI) of requester is a square region of a by a, and that during the MCS campaign, each user keeps position-fixed or moves only in a very restricted space relative to the RoI, such as in the office room or around the bus stop. In the real-time MCS application, the user movement can be neglected because the sensing tasks published by the requester must be completed in a short term [15]. In the ambient noise and traffic accidents monitoring, for instance, users can complete a sensing task and return sound and photo data in a few seconds or even shorter. Hence it is reasonable to assume stationary users in these real-time MCS scenarios. We denote by $\mathcal{U}$ the set of all available users, who are ready to participate in the MCS campaign, and by A, the area of requester's RoI. If user $u_i \in \mathcal{U}$ is recruited to participate in an MCS campaign, she will charge the platform c_i, which compensates her efforts in contributing sensing data. We assume that the platform knows the charge of each available user even before the MCS campaign launches.

2.2. Models of Sensing Coverage and Sensing Utility

In this paper, we consider the disk sensing coverage model, as shown in Figure 2; specifically, the on-phone sensor of user u can cover a disk-shape region that centers the position of u with some sensing radius. We assume that all users have an identical sensing radius r, and r is much less than a. For simplicity, we assume the sensing coverage (disk) of each user is completely within the given RoI. Next we define the sensing utility based on the above sensing coverage model.

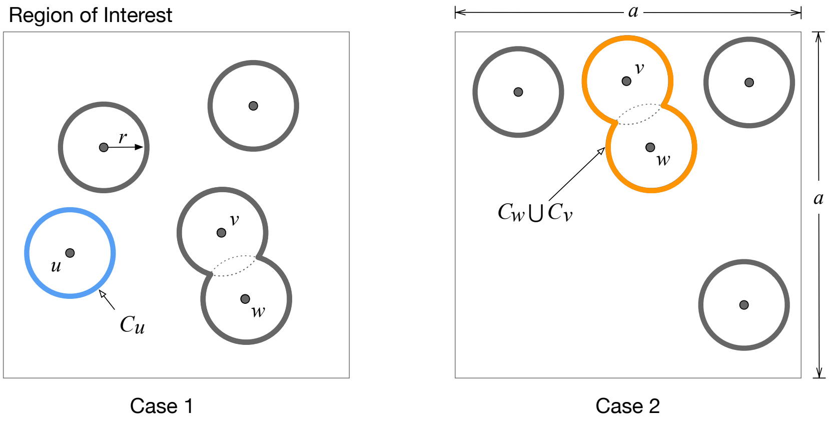


Figure 2. Comparison of two user selection cases, where the left one is more geographically-balancing than the right one, though both can achieve an identical area of total sensing coverage in the square RoI of size $a \times a$.

Definition 1. *Sensing utility based on coverage area. In an MCS campaign with the user set $\mathcal{U}$ and a RoI of area A, the sensing utility that $\mathcal{U}$ can achieve is defined by*

$$\phi(\mathcal{U}) = \frac{S(\bigcup_{u_i \in \mathcal{U}} C_i)}{A}, \tag{1}$$

where C_i is the sensing coverage of u_i and function $S(\cdot)$ is to calculate the area of the union coverage of all the input users.

The union coverage of a set of users can be obtained geometrically, but in Section 3.4, we will present a fast approach to effectively estimating the union coverage with any complicated boundary. Clearly, the area-based sensing utility measures how much a RoI can be covered. In Figure 2, for instance, each of the two cases has five users, and both of them achieve the same area-based sensing utility. It is easy to observe in Figure 2 that the total sensing coverage of Case 1 is more balancing than that of Case 2. To offer the requester a better sensing coverage, in this paper, we propose a new metric for profiling the coverage-based sensing utility: this metric jointly considers the area of the total sensing coverage and the geographical balance of the selected users (i.e., their sensing disks). We next present how to quantitatively measure the geographical balance of the users distributing in a RoI.

Let $\boldsymbol{x}_i$ represent the 2D coordinate of user u_i's position. We first calculate the median or the centroid position $\bar{\boldsymbol{x}}$ by Equation (2), and then, the average distance from any user to position $\bar{\boldsymbol{x}}$ by Equation (3), where $d(\cdot)$ is the function to obtain the Euclidean distance between two positions on plane.

$$\bar{\boldsymbol{x}} = \frac{1}{n} \sum_{u_i \in \mathcal{U}} \boldsymbol{x}_i \tag{2}$$

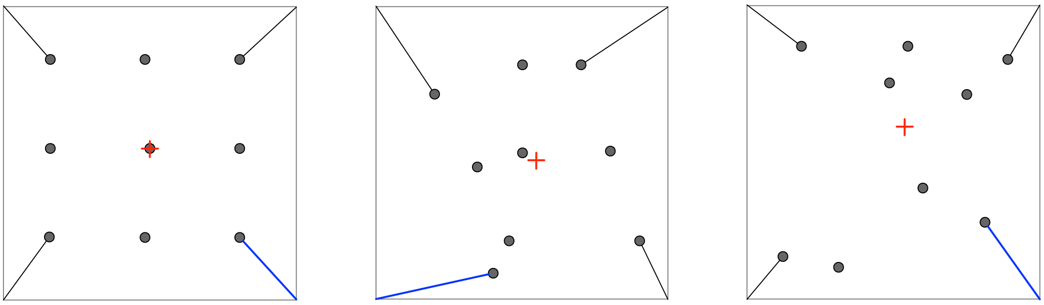
$$\bar{d} = \sqrt{\frac{1}{n} \sum_{u_i \in \mathcal{U}} d(\boldsymbol{x}_i, \bar{\boldsymbol{x}})} \tag{3}$$

Definition 2. *Loss of coverage balance. Given the positions of all users in $\mathcal{U}$ and a RoI, the coverage balance loss of $\mathcal{U}$, denoted by $\ell(\mathcal{U})$, is defined as*

$$\ell(\mathcal{U}) = \frac{\max\limits_{\forall \boldsymbol{y} \in [\text{RoI}]} \min\limits_{u_i \in \mathcal{U}} \{d(\boldsymbol{y}, \boldsymbol{x}_i)\}}{\bar{d}}, \tag{4}$$

where [RoI] represents the position set of four RoI corners.

The loss of coverage balance defined above measures the geographical distribution of a set of points that are deployed in a 2D plane. Figure 3 shows three deployments of points in an identical square, each of which includes nine points. Intuitively, deployment D_1 is more balancing than both D_2 and D_3. By Definition 2, we can obtain $\ell(D_1) = 2.04$, less than the coverage balance losses of D_2 and D_3. Basically, the presented coverage balance loss is established on two deviation measures: (1) the deviation of users from their centroid, i.e., the denominator term of Equation (4), and (2) their deviation from the four RoI corners, i.e., the numerator term of Equation (4). Obviously, the greater the value of $\ell(\mathcal{U})$, the less balancing the geographical distribution of the users. In the three deployments shown in Figure 3, we calculate the entropy, $H(D_i)$, for a random event happing within the region of deployment D_i. We here consider a random event that uniformly occurs in region of D_i. The entropy for this event is based on the sensing coverage and then it is one effective way of evaluating the sensing utility [16,17]. In detail, let p_k be the probability that the random event can be simultaneously detected or covered by k users deployed in D_i, and then we can express the entropy of this event with $H(D_i) = -\sum_{k=1}^{m} p_k \log p_k$, where m is the number of users in D_i (here in the example of Figure 3, $m = 9$ in each deployment). Figure 4 compares the variations of entropy under three deployments shown in Figure 3. We can see that when the sensing range is small, almost all the users in each deployment cannot cover the random event, i.e., the value of p_k is very close to zero, and consequently, the corresponding entropy is low in each deployment. Similarly, larger sensing ranges lead to lower entropy, because every random event can be covered with higher probability. Given a sensing range of user, however, the more balancing the coverage of a deployment, the lower the entropy of a random event, especially when users are densely deployed in RoI. Such an observation indicates that balancing coverage benefits the decrease of sensing uncertainty or the increase of sensing utility. It is necessary to take into account the coverage balance in the MCS user selection.



(**a**) deployment D_1 (**b**) deployment D_2 (**c**) deployment D_3

Figure 3. Comparison of three deployments of points in the coverage balance loss. In each deployment, the side length of the square region of interest (RoI) is 10, and the red cross represents the median position of all the nine points. The losses of coverage balance in the three sub-figures are $\ell(D_1) = 2.04$, $\ell(D_2) = 2.23$, and $\ell(D_3) = 2.37$.

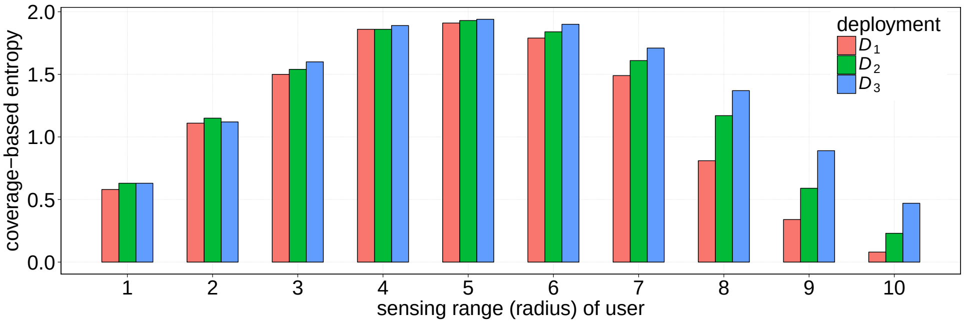


Figure 4. Coverage-based entropies of a random event under the three deployments shown in Figure 3.

Definition 3. *Sensing utility based on coverage area and balance (a.k.a coverage balance-based sensing utility). For a given user set $\mathcal{U}$ and a RoI, we denote by $v(\mathcal{U})$ the coverage balance-based sensing utility that $\mathcal{U}$ can achieve in this RoI, and it is given by*

$$v(\mathcal{U}) = \frac{\phi(\mathcal{U})}{\ell(\mathcal{U})}. \tag{5}$$

Different from the coverage area-based sensing utility model widely used in literature, Definition 3 considers not only the total sensing coverage area of the users but their geographical balance as to the given RoI. Targeting the maximization of such a joint metric, the budget-feasible user recruitment made by the MCS platform can offer the requester a more desirable and beneficial observation about her RoI, especially when the budget is very limited.

2.3. Problem Description

In this section, we first formally describe the problem to be addressed in this paper, and then, we analyze its intractability in computation.

Definition 4. *The coverage-balancing user selection (CBUS) problem: to choose a subset U of $\mathcal{U}$ such that the coverage balance-based sensing utility of U can be maximized, while the total charge of all the users of U does not violate requester's budget. The CBUS problem can be formally written as follows.*

$$\max \quad v(U) \tag{6}$$

$$st. \quad U \subseteq \mathcal{U} \tag{7}$$

$$\sum_{u_i \in U} c_i \leq B. \tag{8}$$

Since $\mathcal{U}$ is finite in size, it must have a finite number of subsets, indicating that there always exists an optimal and nonempty solution for the CBUS problem. Yet the CBUS problem is NP-hard—for a general CBUS instance, we cannot solve it in polynomial time unless P = NP. Theorem 1 proves the computational intractability of the CBUS problem.

Theorem 1. *The general CBUS problem is NP-hard.*

Proof. We first consider a new problem (denoted by r-CBUS) that can be reduced from the CBUS problem and then prove the NP-hardness of r-CBUS. Simply, the r-CBUS problem is selecting a subset of $\mathcal{U}$ to maximize the coverage area-based sensing utility, $\phi(\mathcal{U})$, while the total charge of $\mathcal{U}$ is not beyond requester's budget B. By Definition 1, we know $\phi(\mathcal{U}) \geq 0$. For any $A \subseteq B \subseteq \mathcal{U}$, we have $\phi(A) \leq \phi(B)$ and we can easily prove the submodularity of function $\phi(\cdot)$, that is, $\phi(A) + \phi(B) \geq \phi(A \cup B) + \phi(A \cap B)$. This indicates that the r-CBUS problem is basically a submodular maximization problem with a knapsack constraint, which is a well-known NP-hard problem [18]. In general, the objective function of the CBUS problem, $\nu(\cdot)$, is non-monotone over the domain of $2^{\mathcal{U}}$, in which the enumerator term $\phi(\cdot)$ is monotone but the denominator term $\ell(\cdot)$ is non-monotone. Without considering the non-monotonicity of the CBUS problem, we can reduce CBUS to the r-CBUS problem. Thus we can conclude that the general CBUS problem is at least NP-hard. □

In fact, the non-monotonicity of the objective function $\nu(\cdot)$ of CBUS makes it ineffective to solve CBUS directly by employing greedy policy or local search, which are often applied to discrete optimization problems.

3. Designs

In this section, we first briefly introduce the basic idea of our algorithm for the CBUS problem and then describe the detailed designs; finally, we show a fast approach to calculating the union coverage, which greatly helps enhance the efficiency of our algorithm in real-life scenarios.

3.1. Overview of Our Designs

Obviously, uncontrolled coverage overlaps of multiple users impact the area-based sensing utility. In addition, if most of the selected users are huddled together within a few sub-regions of RoI, we cannot achieve a desirable coverage balance. These observations inspire us to obtain an approximate CBUS solution in such a greedy way: selecting as many users as possible until the budget is violated, while keeping their sensing coverages as separate as possible. With the above heuristic, we design an algorithm, called MIA, which involves two sequential stages.

In the first stage, MIA determines a subset U of $\mathcal{U}$ such that $|U|$ is maximized while making sure each user of U does not overlap with any others. There are three mutually-exclusive situations facing the resulted subset U: (1) the total charge of the users of U is exactly equal to the budget, (2) U overruns the budget, and (3) the budget is not fully fulfilled by U, which leaves room for adding extra users into U. If one of the last two cases happens, MIA enters the second stage to do the user adjustment: deleting some users from U and adding extra users into U for the second and the third cases, respectively. The two stages of MIA are described in Sections 3.2 and 3.3.

3.2. Determining Maximum Users without Overlaps

Since we attempt to pick up as many non-overlapping users as possible in the first stage of MIA, we naturally employ the maximum independent set (MIS) problem to model what this stage needs to achieve. For a given undirected, connected graph G with n vertices and m edges, the MIS problem is to insert non-adjacent vertices of G into V_I such that the size of V_I can be maximized [19–21]. If an MCS campaign is ready to launch, we can form the users into a disk graph, denoted by $\tilde{G}$, in which the vertices are just the users and an edge is drawn between two vertices if both of them overlap in sensing coverage. With doing so, the MIS solution for $\tilde{G}$ indicates a subset U of $\mathcal{U}$ such that any two users of U are independent or their sensing disks are non-overlapping.

The MIS problem is a well-known NP-hard problem in graph theory. One of the state-of-the-art works on MIS is due to Y. Liu, et al. [22], who present the one-k-swap algorithm for general MIS problems. We use this algorithm to satisfy the requirement of the first stage of MIA. To make this paper easy to follow, we next resort to the graph terminology and Figure 5 to briefly explain how the one-k-swap algorithm works with $\mathcal{O}(n^2)$ time complexity.

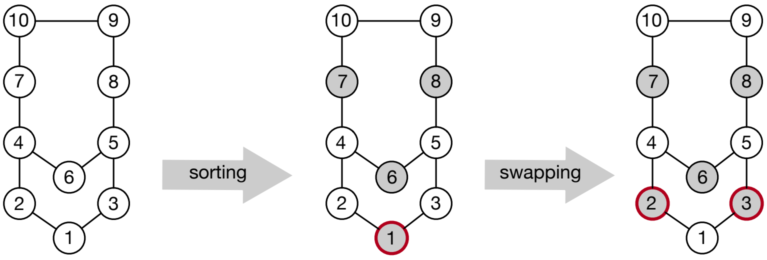


Figure 5. Basic idea of the one-k-swap algorithm in the independent set augmentation. Before and after swapping, the gray vertices form two independent sets with the sizes of four and five, respectively.

In essence, one-k-swap is a greedy iterative algorithm, which continues to select qualified vertices and then struggles to improve the results by the swapping operation. Figure 5 shows a walk-through example including 10 vertices. one-k-swap first sorts these vertices in an ascending order of vertex degree, yielding the vertex set $\tilde{V} = \{v_1, v_2, v_3, v_6, v_7, v_8, v_9, v_{10}, v_4, v_5\}$. Then, one-k-swap sets $\tilde{V}_I = \emptyset$ and continues to examine the vertices of $\tilde{V}$ one by one, during which it inserts the examined vertex v_i into $\tilde{V}_I$ if and only if v_i is non-adjacent with any vertex of $\tilde{V}_I$; such an iteration terminates with $\tilde{V}_I = \{v_1, v_6, v_7, v_8\}$, as shown in the middle part of Figure 5. Iteratively, one-k-swap augments $\tilde{V}_I$ into a larger independent set by continual swaps. More specifically, it exchanges v_1 with v_2 and v_3, i.e., deleting v_1 from $\tilde{V}_I$ but adding v_2 and v_3 into $\tilde{V}_I$ in the right part of Figure 5, which increases the size of $\tilde{V}_I$ by one without producing any conflicts. Such swapping operations stop the moment the size of $\tilde{V}_I$ does not increase.

As aforementioned, we can form the users of $\mathcal{U}$ into a disk graph $\tilde{G}$, according to the overlaps among these users. If we run the one-k-swap algorithm on $\tilde{G}$, we will surely obtain a subset U_I of $\mathcal{U}$ such that further adding a user into U_I will lead to at least one sensing coverage overlap.

3.3. Budget-Based User Adjustments

If the determined subset U exactly fulfills requester's budget, i.e., $\sum_{u \in U} c_u = B$, we can reasonably terminate MIA; otherwise, we have to do user adjustment on U for better budget implementation. The second stage of MIA employs two greedy algorithms, Algorithms 1,2, to deal with the budget overrun and the budget surplus, respectively.

When the budget overrun occurs, i.e., $\sum_{u \in U_I} c_u > B$, Algorithm 1 greedily deletes users until the budget overrun is removed. The greedy criterion is based on the marginal profit in terms of sensing utility we can obtain if deleting a user from U_I. From line 3 of Algorithm 1, the marginal profit $\Delta\nu$ indicates that how much the total sensing utility will be decreased at a unit user cost. In other words, this greedy criterion guides MIA to preferentially delete the user who charges more but leads to less impact on the total sensing utility. If user u^* is determined in line 3 of Algorithm 1, she will be deleted from U_I.

Algorithm 1: Deleting Users in the Case of Budget Overrun.

```
Input: set $U_I$
Output: $U_I^- \subset U_I$
$U_I^- \leftarrow U_I$
while $\sum_{u \in U_I^-} c_u > B$ do
    Select user $u^* \in U_I^-$ such that $\Delta\nu = \frac{\nu(U_I^-) - \nu(U_I^- - \{u^*\})}{c_{u^*}}$ can be minimized
    $U_I^- \leftarrow U_I^- - \{u^*\}$
return
```

When a budget surplus happens to U_I, MIA uses Algorithm 2 to add extra users into U_I, aimed at spending the whole budget. Similar with Algorithm 1, Algorithm 2 also includes a greedy iteration, which always picks out the user from $\mathcal{U} \backslash U_I$, such that the increase of sensing utility comes with less cost, i.e., the marginal profit $\Delta\nu$ in line 3 of Algorithm 2 can be maximized. In the iterations of Algorithm 2, the value of $\Delta\nu$ is possibly negative because of the non-decreasing monotonicity of function $\nu(\cdot)$. Thus Algorithm 2 can terminate when each available user of $\mathcal{U} \backslash U_I^+$ only contributes non-positive marginal profit.

Algorithm 2: Adding Users in the Case of Budget Surplus.

```
  Input: $\mathcal{U}$ and $U_I$
  Output: $U_I^+$
1 $U_I^+ \leftarrow U_I$
2 while $\sum_{u \in U_I^+} c_u < B$ and $\mathcal{U} \backslash U_I^+ \neq \emptyset$ do
3     Select user $u^* \in \mathcal{U} \backslash U_I^+$ such that $\Delta\nu = \frac{\nu(U_I^+ + \{u^*\}) - \nu(U_I^+)}{c_{u^*}}$ can be maximized
4     if $\Delta\nu > 0$ and $c_{u^*} + \sum_{u \in U_I^+} c_u \leq B$ then
5         $U_I^+ \leftarrow U_I^+ + \{u^*\}$
6         $\mathcal{U} \leftarrow \mathcal{U} - \{u^*\}$
7     else
8         return
9 return
```

3.4. Calculation of Coverage Union

Calculating the sensing coverage area is a major component in the proposed algorithm MIA. For a single user u, its coverage area equals πr^2, where r is the sensing radius of u. For a union sensing coverage of multiple users, furthermore, we can calculate its exact area by the means of geometry computation. For a general union coverage, however, the overlaps formed by the communication disks might be very complicated, resulting in irregular boundary and even non-convex holes; and geometrically calculating the area of such a union coverage will surely lead to intensive geometry computation, which considerably impacts the efficiency of MIA.

In this paper we leverage the grayscale representation and the color aberration to "cut out" the region of the given union coverage, and then, MIA can calculate the corresponding area by pixel counting. Strictly, the accuracy of the above approach depends on the pixel size in use. In practice, however, the pixel sizes are at most in millimeters such that the pixel-based area calculation is adequate in accuracy.

4. Evaluation

In this section we conducted numeric experiments to evaluate the performance of our designs. Recall that our main work was to select a subset of users such that the coverage balance-based sensing utility can be maximized under a budget constraint. In our numeric experiments, therefore, we did not need to simulate users' behaviors in data level as many previous works involving data delivery did in their experiments. Our experiments were conducted with a Java-based homemade simulator, which ran on a PC with a CPU of 2.4 GHz and the memory of 8 GB. We first give the two baseline algorithms, and then describe the experimental setups; finally, we show and analyze the experimental results.

4.1. Baseline Algorithms

We compare MIA with two baseline algorithms. The first baseline, called Rand, was selecting users in a completely random way, and it stopped right before the budget overrun occurs. We do not give the details of Rand because it is rather easy to understand. We designed the second baseline algorithm, called the partition-based random selection (ParRand), which was essentially random but employed the RoI partitioning to select users as evenly as possible. ParRand accords with the intuition of making geographically balancing user selection and then serves as a baseline in our evaluation. Algorithm 3 describes the proposed ParRand.

Algorithm 3: (ParRand) Partition-Based Random User Selection.

```
   Input: $\mathcal{U}$, RoI (a square)
   Output: $U$
1  $U \leftarrow \emptyset$
2  $\mathcal{R} \leftarrow \{\text{RoI}\}$
3  while true do
4      $\mathcal{R}^d \leftarrow \emptyset$
5      for $R_i \in \mathcal{R}$ do
6          Divide $R_i$ into four equal sub-squares (cells) and represent them with $\{R_i^1, R_i^2, R_i^3, R_i^4\}$
7          Add cell $R_i^k$ (for all $k$) into $\mathcal{R}^d$
8      $\mathcal{R} \leftarrow \mathcal{R}^d$
9      Shuffle $\mathcal{R}$ randomly for unbiased cell checking
10     for $R_i \in \mathcal{R}$ do
11         Randomly select $u_r$ from the users who locate in cell $R_i$ but have not yet selected in
             previous iterations
12         if $c_r + \sum_{u_i \in U} c_i \leq B$ then
13             $U \leftarrow U + \{u_r\}$
14         else
15             return
16 return
```

The main body of ParRand is a while loop. In each iteration ParRand involves two stages: dividing the RoI into multiple equal cells and randomly selecting a user from each cell as long as the budget is not violated. Figure 6 shows an example of ParRand. In the first iteration, ParRand divided the whole RoI into four equal cells (small squares). In line 9 of ParRand, the set $\mathcal{R}$ including four cells are shuffled to create a random order of checking every cell of $\mathcal{R}$; such a shuffle leads to unbiased user selection over the current cells. The second stage of the first iteration goes to randomly select four users, each from a cell. If there still is a budget surplus in the first iteration, ParRand will enter the next iteration. In the second iteration, ParRand further divides each cell made by the first iteration into four equal cells (smaller squares) and consequently, there are totally 16 cells in the second iteration.

From each of the 16 cells, ParRand attempts to randomly pick out a user who has not yet been selected in previous iterations; in the example of Figure 6, ParRand selects 13 users in its second iteration. Similarly, the third iteration produces 64 cells in the example of Figure 6, but it stops with selecting only two users because the budget overrun will occur otherwise.

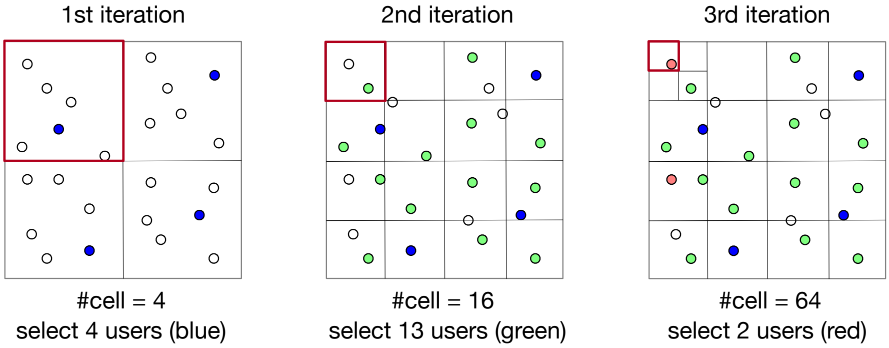


Figure 6. Example of ParRand, which terminates in its third iteration because of the budget overrun.

In experiments, we found that ParRand performed much better in terms of coverage balance than Rand did. The reason relies on that ParRand always selects users over the evenly-partitioned RoI; on the contrast, Rand selects users directly from the whole RoI. Noticeably, one major disadvantage of ParRand is its computational cost, which is clearly shown by Figure 6. If ParRand terminates with k iterations, we then know that ParRand needs to check 4^i cells in the i-th iteration for $1 \leq i \leq k$. Thus, k iterations of ParRand asymptotically includes $\sum_{1 \leq i \leq k} 4^i = \frac{4}{3}(4^k - 1)$ times of cell checking, indicating an exponential growth of time complexity.

4.2. Experimental Setup

We set various cases to evaluate our algorithm and compare it with the two baselines in each common case. In all experiments, we set the RoI to be a square with the side length of $a = 12$, and we set the sensing radius of all users to be one. Before each experiment, we uniformly distributed $|\mathcal{U}|$ users within the RoI at random, and at the same time, we confined their sensing disks to the RoI. In addition, the charge of each user was determined in each experiment to be a random value with $[\check{c}, \hat{c}]$, where $\check{c}$ is always set to one and $\hat{c}$ is a variable to be examined. We repeated 40 experiments with random seeds for each case. Table 1 shows the three experimental cases and the setups of other parameters.

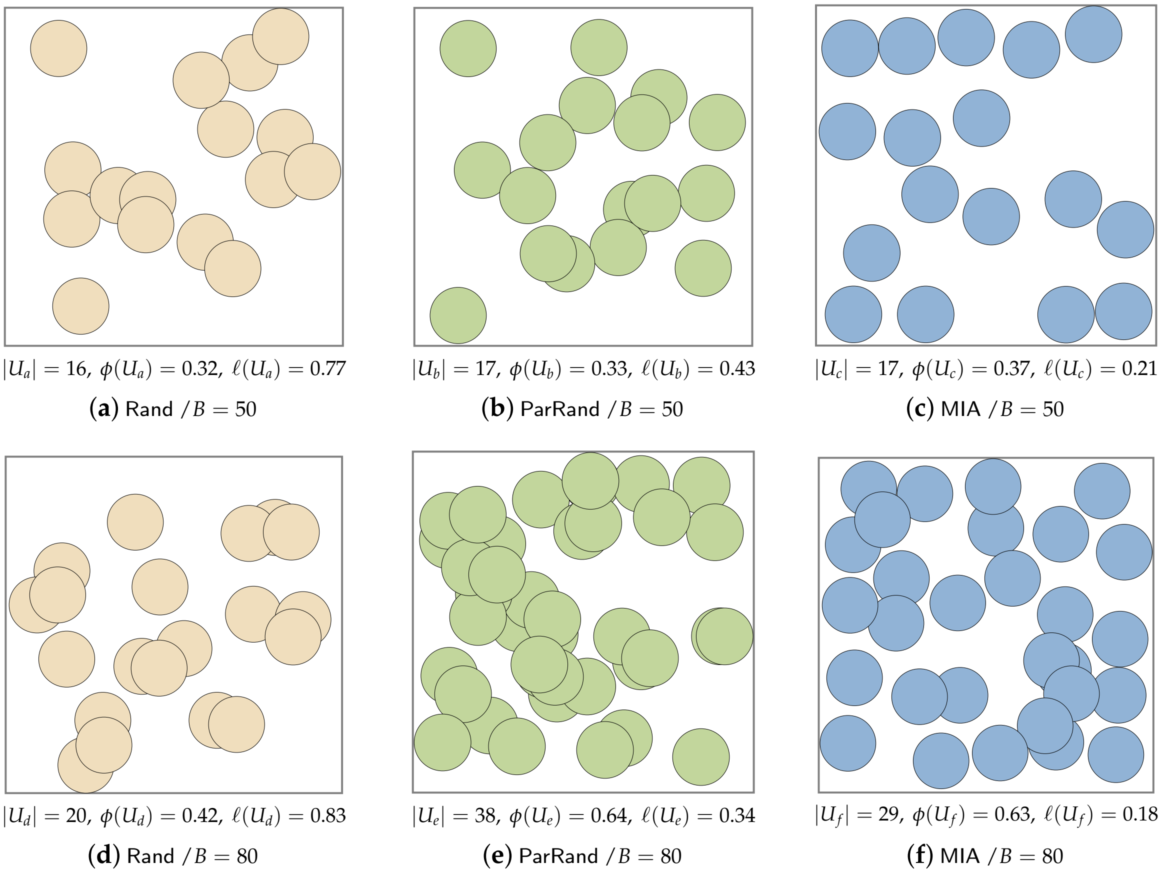
Table 1. Setup of parameters and variable ranges used in each case.

Parameter	Description	Value or Range		
		Case 1	Case 2	Case 3
$\lvert\mathcal{U}\rvert$	the number of initial users	40∼200 (step = 40)	80	80
B	the budget	100	40∼200 (step = 40)	100
$\hat{c}$	the upper bound for user cost	6	6	2∼10 (step = 2)

4.3. Results and Analyses

Figure 7 shows the overall sensing coverage achieved by the proposed MIA and by the two baselines. These six sub-figures snapshot the experimental results under the setups of 200 users and two different budgets (50 and 80). When B is set to 50, only a small fraction of the users can be selected under each algorithm because of the strict budget. By comparing Figure 7a–c, we see that MIA covered more RoI without any coverage overlaps than the other two baselines do, and that MIA achieved better coverage balance. When B was set to 80 in Figure 7d–f, all the three algorithms recruit more users than they do with $B = 50$. Rand picked out only 20 users before the budget overrun was hit, significantly less than the users selected by ParRand and MIA. Since Rand selected users without considering their charges, it was easy to saturate the budget. We also find that with a relatively large budget, MIA has no obvious superiority in total sensing coverage than ParRand, and sometimes, ParRand can cover a little more than MIA, as shown in Figure 7e,f. Nevertheless, MIA outperforms ParRand in terms of coverage balance when $B = 80$. MIA achieves a better tradeoff between the coverage balance and the sensing coverage area.

We evaluate the performance of the three algorithms in coverage area-based sensing utility, and the results are shown in Figure 8. In case 1, Rand always performed worst. With the increase of available users, the total sensing area of MIA was less than, but very close to that of ParRand—the difference was at most 3%. In case 2, which includes 80 available users, the increase of budget incentivizes all the three algorithms to recruit more users. Consequently, when the budget was set to be larger, all of them can cover more and their performance gaps are narrower. In case 3 with a fixed budget of 100, after the upper bound for the user charge goes beyond four, MIA and ParRand obviously perform better than Rand.



$|U_a| = 16,\ \phi(U_a) = 0.32,\ \ell(U_a) = 0.77$

(**a**) Rand $/B = 50$

$|U_b| = 17,\ \phi(U_b) = 0.33,\ \ell(U_b) = 0.43$

(**b**) ParRand $/B = 50$

$|U_c| = 17,\ \phi(U_c) = 0.37,\ \ell(U_c) = 0.21$

(**c**) MIA $/B = 50$

$|U_d| = 20,\ \phi(U_d) = 0.42,\ \ell(U_d) = 0.83$

(**d**) Rand $/B = 80$

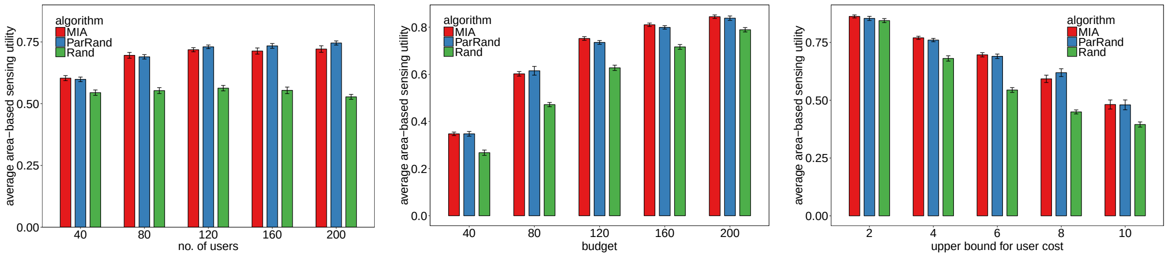
$|U_e| = 38,\ \phi(U_e) = 0.64,\ \ell(U_e) = 0.34$

(**e**) ParRand $/B = 80$

$|U_f| = 29,\ \phi(U_f) = 0.63,\ \ell(U_f) = 0.18$

(**f**) MIA $/B = 80$

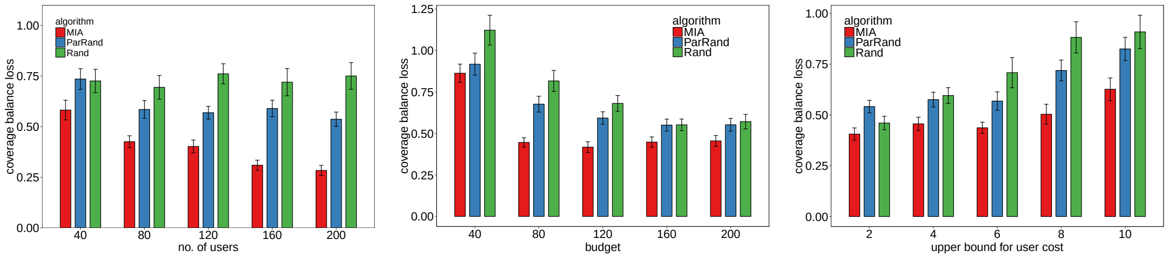
Figure 7. Snapshots of the final sensing coverage after the three algorithms terminate in two common experiments. Both experiments have 200 available users, but one experiment is set with $B = 50$ and the other, with $B = 80$.



(**a**) case 1 (**b**) case 2 (**c**) case 3

Figure 8. Comparison of the three algorithms in coverage area-based sensing utility.

We in Figure 9 compare the three algorithms in terms of coverage balance loss. In Figure 9a, the coverage balance loss increasingly reduces: the loss value drops by half when the number of available users increases from 40 up to 200. Furthermore, Figure 9b,c can also demonstrate the MIA's distinct advantage in coverage balance over the two baselines. In case 2, the higher the budget, the more users were selected by each of the three algorithms, and thus, their performance in coverage balance is continually improved as the budget increases. Although the increase of the upper bound for the user charge reduces the number of users finally selected by each algorithm, MIA rises more slowly in coverage balance loss than the other two baselines do.



(**a**) case 1 (**b**) case 2 (**c**) case 3

Figure 9. Comparison of the three algorithms in coverage balance loss.

5. Related Work

Coverage problem is a critical topic in wireless sensor networks [23], in which the sensing utility is reckoned to be closely related to the sensing coverage. In the community of wireless sensor networks, the coverage problems often target minimizing the number of sensors while guaranteeing a full and connected coverage. In wireless sensor networks, the coverage determination needs to create a connected network among the selected sensors. In MCS, however, the user's mobilephone can directly connect with the platform via the internet. In addition, the works on wireless sensor network coverage problems put less attention on coverage balance.

In the MCS community, researchers have done some preliminary works in recent years. Wang et al. [7] consider the coverage area-based sensing utility model, and propose an algorithm to minimize the overall system cost with some QoS constraint. Xiong et al. [24] design CrowdTasker for MCS, which aims at maximizing the coverage quality of sensing tasks while agreeing with a given incentive budget constraint. Zhang et al. [8] target the maximization of sensing quality with budget constraint under the MCS scenario different from ours. They measure the sensing quality by the coverage of mobilephone users that can be achieved in user movement; in detail, their coverage-based sensing quality is determined by the number of points of interest that a user can cover. Xiong et al. [9] present iCrowd, which recruits users to cover more subareas of the RoI while minimizing the payment to the selected users. Similar with our coverage model, Obinikpo et al. [25] avoid more than two users who cover the same target, and they propose a scheduling policy to save the energy of sensing devices. But they do not consider the budget constraint in their models. Yu et al. [10] consider the *k*-coverage model, which needs multiple users to cover any given point of interest. Girolami et al. [26] propose the binary coverage model and the aging coverage model for MCS, and employ spatial interpolation strategy to improve the sensing utility. In their models, however, the RoI is divided into a finite set of equal square tiles, and they measure the degree of coverage by the number of the tiles that can be covered by users' sensing disks. Basically, their models measure the sensing utility based on the sensing area, and do not take into account the coverage balance.

In spite of the ongoing advances in MCS, there is still a lack of effective solutions for the user selection that can achieve balancing coverage under a constrained budget.

6. Conclusions

In this paper we have proposed the CBUS problem, which is to achieve the coverage-balancing user selection in the MCS with a budget constraint. We have proven the NP-hardness of CBUS and then designed algorithm MIA to approximately solve it. MIA leverages the maximum independent set model to first obtain a relatively balancing coverage over the given RoI, and then includes two algorithms to deal with the budget overrun and the budget surplus. Extensive numeric experiments show the efficacy of our designs both in the coverage balance and in the total coverage area. In the future work, we will extend our model of coverage balance and design effective algorithm such that they can profile the MCS user selection in a spatiotemporal way, and then, can well improve the sensing utility of long-term sensing tasks.

Author Contributions: G.S. and Y.W. designed the algorithms. Y.W. and X.D. established the simulation testbed and conducted the experiments. Y.W. and X.D. wrote this paper.

Funding: The authors were supported, in part, by the Fundamental Research Funds for the Central Universities of China with Grant No. 2017ZY20 and by the National Natural Science Foundation (NSF) Project of China with Grant No. 61300180.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Guo, B.; Wang, Z.; Yu, Z.; Wang, Y.; Yen, N.; Huang, R.; Zhou, X. Mobile Crowd Sensing and Computing: The Review of an Emerging Human-Powered Sensing Paradigm. *ACM Comput. Surv.* **2015**, *48*, 7. [CrossRef]
2. Antonic, A.; Marjanovic, M.; Pripuzic, K.; Zarko, I. A Mobile Crowd Sensing Ecosystem Enabled by CUPUS: Cloud-based Publish/Subscribe Middleware for the Internet of Things. *Future Gener. Comput. Syst.* **2016**, *56*, 607–622. [CrossRef]
3. Guo, B.; Chen, H.; Yu, Z.; Xie, X.; Huangfu, S.; Zhang, D. FlierMeet: A mobile crowdsensing system for cross-space public information sharing. *IEEE Trans. Mob. Comput.* **2015**, *14*, 2020–2033. [CrossRef]
4. Gregoria, E.; Improta, A.; Lenzini, L.; Luconi, V.; Redini, N.; Vecchio, A. Smartphone-based crowdsourcing for estimating the bottleneck capacity in wireless networks. *J. Netw. Comput. Appl.* **2016**, *64*, 62–75. [CrossRef]
5. Elhamshary, M.; Youssef, M.; Uchiyama, A.; Yamaguchi, H.; Higshino, T. TransitLabel: A Crowd-Sensing System for Automatic Labeling of Transit Stations Semantics. In Proceedings of the 14th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys), Singapore, 26–30 June 2016; pp. 193–206.
6. Azzam, R.; Mizouni, R.; Otrok, H.; Ouali, A.; Singh, S. GRS: A Group-Based Recruitment System for Mobile Crowd Sensing. *J. Netw. Comput. Appl.* **2016**, *72*, 38–50. [CrossRef]
7. Wang, Z.; Huang, D.; Wu, H.; Deng, Y.; Aikebaier, A.; Teranishi, Y. QoS-Constrained Sensing Task Assignment for Mobile Crowd Sensing. In Proceedings of the 2014 IEEE Global Communications Conference (Globecom), Austin, TX, USA, 8–12 December 2014; pp. 311–316.
8. Zhang, M.; Yang, P.; Tian, C.; Tang, S.; Gao, X.; Wang, B.; Xiao, F. Quality-aware Sensing Coverage in Budget Constrained Mobile Crowdsensing Networks. *IEEE Trans. Veh. Technol.* **2015**, *65*, 7698–7707. [CrossRef]
9. Xiong, H.; Zhang, D.; Chen, G.; Wang, L.; Gauthier, V.; Barnes, L. iCrowd: Near-Optimal Task Allocation for Piggyback Crowdsensing. *IEEE Trans. Mob. Comput.* **2016**, *15*, 2010–2022. [CrossRef]

10. Yu, Z.Y.; Zhou, J.; Guo, W.; Guo, L.; Yu, Z.W. Participant selection for *t*-sweep *k*-coverage crowd sensing tasks. *World Wide Web* **2018**, *21*, 741–758. [CrossRef]
11. Huang, P.; Zhu, W.; Guo, L. Optimizing Movement for Maximizing Lifetime of Mobile Sensors for Covering Targets on a Line. *Sensors* **2019**, *19*, 273. [CrossRef] [PubMed]
12. Goto, K.; Sasaki, Y.; Hara, T.; Nishio, S. A Mobile Agents Control Method for Sensor Data Gathering Based on Geographical Granularity in Dense Mobile Sensor Networks. *J. Inf. Process.* **2012**, *53*, 741–753.
13. Wang, L.; Zhang, D.; Pathak, A.; Chen, C.; Xiong, H.; Yang, D.; Wang, Y. CCS-TA: Quality-Guaranteed Online Task Allocation in Compressive Crowdsensing. In Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing, (Ubicomp), Osaka, Japan, 7–11 September 2015.
14. Restuccia, F.; Ghosh, N.; Bhattacharjee, S.; Das, S.; Melodia, T. Quality of Information in Mobile Crowdsensing: Survey and Research Challenges. *ACM Trans. Sens. Netw.* **2017**, *13*, 34. [CrossRef]
15. Hu, T.; Xiao, M.; Hu, C.; Gao, G.; Wang, B. A QoS-sensitive task assignment algorithm for mobile crowdsensing. *Pervasive Mob. Comput.* **2017**, *41*, 333–342. [CrossRef]
16. Wang, H.; Yao, K.; Pottie, G.; Estrin, D. Entropy-based Sensor Selection Heuristic for Target Localization. In Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks, Berkeley, CA, USA, 26–27 April 2004.
17. Zilan, R.; Erten, Y. Use of entropy to control coverage and energy dissipation in wireless sensor networks. In Proceedings of the 22nd International Symposium on Computer and Information Sciences, Ankara, Turkey, 7–9 November 2007.
18. Sviridenko, M. A note on maximizing a submodular set function subject to a knapsack constraint. *Oper. Res. Lett.* **2004**, *32*, 41–43. [CrossRef]
19. Andrade, D.; Resende, M.; Werneck, R. Fastlocal search for the maximum independent set problem. *J. Heuristics* **2012**, *18*, 525–547. [CrossRef]
20. Wan, P.; Jia, X.; Dai, G.; Du, H.; Frieder, O. Fast and simple approximation algorithms for maximum weighted independent set of links. In Proceedings of the IEEE INFOCOM 2014—IEEE Conference on Computer Communications Infocom, Toronto, ON, Canada, 27 April–2 May 2014; pp. 1653–1661.
21. Xiao, M.; Nagamochi, H. Exact algorithms for maximum independent set. *Inf. Comput.* **2017**, *225*, 126–146. [CrossRef]
22. Liu, Y.; Lu, J.; Yang, H.; Xiao, X.; Wei, Z. Towards Maximum Independent Sets on Massive Graphs. *Proc. VLDB Endow.* **2015**, *8*, 2122–2133. [CrossRef]
23. Sangwan, A.; Singh, R.P. Survey on Coverage Problems in Wireless Sensor Networks. *Wirel. Pers. Commun.* **2015**, *80*, 1475–1500. [CrossRef]
24. Xiong, H.; Zhang, D.; Chen, G.; Wang, L.; Gauthier, V. CrowdTasker: Maximizing Coverage Quality in Piggyback Crowdsensing under Budget Constraint. In Proceedings of the 2015 IEEE International Conference on Pervasive Computing and Communications (PerCom), Louis, MO, USA, 23–27 March 2015.
25. Obinikpo, A.; Zhang, Y.; Song, H.; Luan, T.; Kantarci, B. Queuing Algorithm for Effective Target Coverage in Mobile Crowd Sensing. *IEEE Internet Things J.* **2017**, *4*, 1046–1055. [CrossRef]
26. Girolami, M.; Chessa, S.; Dragone, M.; Bouroche, M.; Cahill, V. Using spatial interpolation in the design of a coverage metric for Mobile CrowdSensing systems. In Proceedings of the IEEE Symposium on Computers and Communication (ISCC), Messina, Italy, 27–30 June 2016; pp. 147–152.