Deadline-Sensitive User Recruitment for Mobile Crowdsensing with Probabilistic Collaboration

Mingjun Xiao*,†, Jie Wu†, He Huang‡, Liusheng Huang*, and Chang Hu*

*School of Computer Science and Technology / Suzhou Institute for Advanced Study, University of Science and Technology of China
†Department of Computer and Information Sciences, Temple University
‡School of Computer Science and Technology, Soochow University, China

Abstract—Mobile crowdsensing is a new paradigm in which a group of mobile users exploit their smart devices to cooperatively perform a large-scale sensing job over urban environments. In this paper, we focus on the Deadline-sensitive User Recruitment (DUR) problem for probabilistically collaborative mobile crowdsensing. Unlike previous works, mobile users in this problem perform sensing tasks with probabilities, and multiple users might be recruited to cooperatively perform a common task, ensuring that the expected completion time is no larger than a deadline. Owing to such a probabilistic collaboration, DUR can be formalized as a non-trivial set cover problem with non-linear programming constraints and an optimization objective of real function. We first prove that the DUR problem is NP-hard. Then, we propose a greedy DUR algorithm, called gDUR, to solve this problem. Next, we prove that the gDUR algorithm can achieve a logarithmic approximation ratio. Furthermore, we extend the problem to a more complex case where sensing duration is taken into consideration, and we propose a sensing-duration-aware user recruitment algorithm, called dDUR. Finally, we validate the performance of the proposed algorithms through extensive simulations, based on a real mobile social network trace and a synthetic trace.

Index Terms—Crowdsensing, mobile social network, probabilistic collaboration, user recruitment

I. INTRODUCTION

In recent years, there has been an explosive proliferation of smartphones. These smartphones have generally been equipped with multi-core processors, gigabytes of memory, and diverse sensors, so that they can be seen as powerful mobile sensors. Thanks to this, a new sensing paradigm, called mobile crowdsensing, is proposed [4]. Roughly speaking, mobile crowdsensing refers to a group of mobile users being coordinated to perform a large-scale sensing job over urban environments through their smartphones. Since mobile crowdsensing can perform sensing jobs that individual users cannot cope with, it has stimulated many applications, such as urban WiFi characterization, traffic information mapping, wireless indoor localization, and so on, attracting much attention [3], [5], [12], [18], [25].

By far, there has been much research on mobile crowdsensing, including platform design, user recruitment or task allocation algorithms, incentive mechanisms, and so on [1], [2], [6]–[8], [10], [14], [15], [19], [21]–[24]. Among them, user recruitment or task allocation is one of the most important topics [2], [6], [7], [10], [15], [21], [24]. In this paper, we focus on the Deadline-sensitive User Recruitment (DUR) problem for probabilistically collaborative mobile crowdsensing. More specifically, a requester wants to collect some sensing data from many points of interest (PoIs) in an urban area, just like [2], [6], [7], [10]. Then, it publishes a crowdsensing request to some mobile users via mobile social networks. These mobile users move around in the urban area every day. Each user might pass by (i.e., cover) some PoIs frequently, so that it can collect the related sensing data with some probabilities, as shown in Fig. 1. If a mobile user participates in crowdsensing, it will charge a cost from the requester as the reward. The crowdsensing job is expected to be accomplished before a given deadline. Our main concern is determining which users should be recruited, so that the requester can minimize the total cost, while ensuring that the expected completion time of the crowdsensing is no larger than the given deadline.

Our DUR problem for probabilistically collaborative mobile crowdsensing in this paper differs from existing user recruitment problems. Existing works mainly focus on deterministic mobile crowdsensing, in which the trajectory of each user is known and deterministic (e.g., [7], [10], [15]), or each user will determine a route (e.g., [2], [6]) for performing sensing tasks; when the trajectory or route covers a PoI, the user can successfully perform its sensing tasks alone, without any cooperation among users. In contrast, our problem is based on the observation that mobile users in real traces do not always move along a fixed path, showing the characteristic of non-deterministic mobility [10], [13], [21]. That is to say, it is probabilistic that mobile users perform sensing tasks. Hence, multiple users need be recruited to cooperatively perform a common task, in order to improve the probability of success. Due to this probabilistic collaboration, our problem can be
formalized as a non-trivial set cover problem with non-linear programming constraints and an optimization objective of real function. The methodology in existing works [2], [6], [7], [10], [15] cannot deal with such a problem [11].

In this paper, we carefully design a utility function, based on which we turn our DUR problem into a Minimum Submodular Cover with Submodular Cost (MSC/SC) problem, and adopt the greedy strategy to solve this problem. Furthermore, we also extend the problem and the solution to a more complex case. More specifically, the major contributions include:

1) We introduce a DUR problem for mobile crowdsensing with probabilistic collaboration. Unlike existing works, mobile users in this problem perform sensing tasks with probabilities, and multiple users need to cooperatively perform a common task, resulting in a non-trivial set cover problem with non-linear programming constraints.

2) We prove the NP-hardness of the DUR problem. Moreover, we propose a utility function, and prove it to be submodular. Owing to this utility function, we turn the DUR problem into an MSC/SC problem to be solved by using a greedy approximation algorithm, called gDUR. In addition, we derive the logarithmic approximation ratio of the gDUR algorithm.

3) We extend the DUR problem to the case, where sensing duration is taken into consideration. To solve this problem, we also propose a submodular utility function, and based on which, we design a sensing-duration-aware user recruitment algorithm, called dDUR. Moreover, we analyze the corresponding approximation ratio.

4) We conduct extensive simulations on a real trace and a synthetic trace to prove the significant performances of the proposed algorithms.

The remainder of the paper is organized as follows. We first review related works in Section II. Then, we introduce the model, and the problem in Section III. The gDUR and dDUR algorithms are proposed in Sections IV and V, respectively. In Section VI, we evaluate the performances of our algorithms through extensive simulations. We conclude the paper in Section VII. Some complex proofs are moved to the Appendix.

II. RELATED WORKS

There has been much research on the user recruitment or task allocation problem of mobile crowdsensing [2], [6], [7], [10], [21], [24]. Most of these works focus on the deterministic mobile crowdsensing. For example, M. Cheung et al. in [2] formulate a movement-related task allocation problem as a task selection game, and propose a distributed algorithm for each user to select its task and determine its movement. Z. He et al. in [7] propose a greedy approximation algorithm and a genetic algorithm for the user recruitment problem of crowdsensing in vehicular networks, where future trajectories of users are taken into account. S. He et al. in [6] considered the task allocation problem with the constraint of time budgets. In these works, the trajectory of each user is known and deterministic (e.g., [7]), or each user will determine a route (e.g., [2], [6]) for performing sensing tasks, and when the trajectory or route covers a PoI, the user can successfully perform its sensing tasks alone. There is no cooperation among users, different from our probabilistically collaborative DUR problem.

Among the existing works, only [10] partially discussed a special non-deterministic mobile crowdsensing. In this work, M. Karaliopoulos et al. studied the problem of recruiting the users whose paths can cover some PoIs with a minimum cost. The authors first consider a deterministic mobility scenario, where each user has a fixed path, and they formulate the user recruitment problem as a trivial set cover problem with a submodular objective function to be solved. Then, the authors also discuss the non-deterministic scenario, in which users’ paths might cover PoIs with some probabilities. In this scenario, the authors let the additive sum of probabilities of each PoI being covered by multiple paths be no less than 1. For example, if a PoI is covered by two paths with the probabilities $p_1$ and $p_2$, then they must be subject to $p_1 + p_2 \geq 1$. Since the constraint based on the additive probability is a linear programming constraint, the problem in [10] is still a trivial set cover problem. In contrast, our problem is a probabilistic set cover problem with non-linear programming constraints and an optimization objective of real function. The methodology in [10], which is only suitable for the set cover problems with linear programming constraints, cannot deal with our problem, according to the corresponding theories in [10], [11]. In the following, we will design a novel utility function, based on which we can turn it into an MSC/SC problem to be solved.

Additionally, Q. Zhao et al. in [24] and M. Xiao et al. in [21] studied the task allocation issues by formulating them as online scheduling problems, also different from our problem.

III. MODEL & PROBLEM

A. Model

We consider a mobile crowdsensing, in which a requester wants to collect some sensing data from many PoIs in an urban area. The crowdsensing job is assumed to have been divided into many sensing tasks according to the PoIs, denoted by $S = \{s_1, s_2, \cdots, s_m\}$, where each task $s_j$ ($1 \leq j \leq m$) is related to a specific PoI. On the other hand, there are many mobile social network users, moving around in the urban area every day. Many real traces demonstrate that mobile social network users will periodically visit some locations that they prefer with probabilities $[9]$, [16], [20]. For example, mobile users might go to their offices, homes, shopping malls, or other places every day. As a result, each user might periodically pass (i.e., cover) one or more specific PoIs with probabilities. Therefore, these users can be recruited by the requester to perform the corresponding sensing tasks. In this paper, we only discuss the users who are willing to participate in the crowdsensing, denoted by $U = \{u_1, \ldots, u_n\}$. The users who can perform task $s_j$ are denoted as $U_j$, and the tasks that user $u_i$ ($1 \leq i \leq n$) can deal with are denoted as $S_i$. Moreover, according to the users’ periodic mobile behaviors, time is divided into many equal-length sensing cycles, denoted by $T$. For instance, a sensing cycle might be a day, or several hours, etc. Then, the detailed crowdsensing is conducted as follows.
First, the requester publishes all sensing tasks in $\mathcal{S}$ to the users in $\mathcal{U}$ via mobile social networks. Then, each user $u_i$ can determine the tasks that it can perform, i.e., $\mathcal{S}_i$. Meanwhile, user $u_i$ can also determine the probability of performing every task $s_j \in \mathcal{S}_i$ in each sensing cycle, called the processing probability and denoted by $p_{ij}$. Actually, many mobile phones provide the functionality to record the trajectories of users. Through this functionality, $u_i$ can derive the frequency of visiting the PoI related to task $s_j$, which can be used to estimate $p_{ij}$. For example, if $u_i$ has recorded its trajectories of 5 sensing cycles, among which it has visited the PoI in 3 sensing cycles, it will set $p_{ij} = 0.6$. Here, if user $u_i$ visited multiple times in a same sensing cycle, it will still be counted as one time, since we only concern whether the user can perform the task in this sensing cycle.

Second, after having derived each sensing probability $p_{ij}$, user $u_i$ will tell the requester which tasks it can deal with and the related probabilities of performing these tasks via the mobile social networks. At the same time, user $u_i$ will also tell the requester that it will charge a cost from the requester as the reward for participating in crowdsensing. We assume that there is a negotiation between the requester and the user, during which they can determine the cost, denoted by $c_i$. In fact, many mechanism can be used to produce the cost and ensure the truthfulness, such as [19], [23]. In this paper, we will not discuss the detailed negotiation mechanism.

Finally, according to the responses from the users in $\mathcal{U}$, the requester makes the decision to recruit some users for performing the sensing tasks in $\mathcal{S}$. If a user $u_i$ is recruited by the requester, it will perform each task in $\mathcal{S}_i$, when passing by the corresponding PoI, until it is told that the task has been completed by other users. In order to improve the probability of success, multiple users might be recruited to perform a common task. Once having completed a task, each user will send the results back to the requester via the mobile social networks.

Let $\mathcal{P} = \{p_{ij}|u_i \in \mathcal{U}, s_j \in \mathcal{S}_i\}$ and $\mathcal{C} = \{c_i|u_i \in \mathcal{U}\}$. Then, we can simply use a four tuple $(\mathcal{U}, \mathcal{S}, \mathcal{P}, \mathcal{C})$ to describe the above mobile crowdsensing with probabilistic collaboration, as shown in Fig. 2.

### B. Problem

We focus on the DUR problem in the above mobile crowdsensing, i.e., which users in $\mathcal{U}$ should be recruited by the requester to perform the tasks in $\mathcal{S}$, so that it can minimize the total cost, while ensuring that the expected completion time of the crowdsensing is no larger than a given deadline $T$. As multiple users might be recruited to perform a task, the probability of a task being processed is actually a joint probability, called joint processing probability. We use the set $\Phi$ to denote a user recruitment solution, i.e., the set of users that the requester recruits.

Based on the joint processing probability of task $s_j$ in each sensing cycle, we can get the corresponding expected completion time, i.e., $\frac{T}{\rho^j_\Phi}$. Further, the DUR problem can be formalized as follows:

\begin{align}
\text{Minimize} : & \quad C(\Phi) = \sum_{u_i \in \Phi} c_i \\
\text{Subject to} : & \quad \Phi \subseteq \mathcal{U} \\
& \quad \frac{1}{\rho^j_\Phi} \leq T, \quad 1 \leq j \leq m
\end{align}

Here, Eq.4 indicates that the expected completion time of each task is no larger than the given deadline $T$. In other words, the joint processing probability of each task $s_j$ is no less than $\frac{T}{\rho^j_\Phi}$, i.e., $\rho^j_\Phi = 1 - \prod_{u_i \in \Phi} (1 - p_{ij}) \geq \frac{T}{T}$. It is this constraint that makes our user recruitment become a set cover problem with non-linear programming constraints, different from the trivial set cover problems in existing works [2], [6], [7], [10], [15].

In this paper, we assume that the DUR problem has at least a feasible solution. That is to say, each task $s_j$ can be performed before the deadline, i.e., $\rho^j_\Phi = 1 - \prod_{u_i \in \Phi} (1 - p_{ij}) \geq \frac{T}{T}$. If there is no feasible solution for the problem, the requester can have the problem be solvable by expanding the user set $\mathcal{U}$ or prolonging the deadline $T$. Moreover, for simplicity, $T$ is assumed to be an integral multiple of $\tau$; otherwise, we can use $\rho^j_\Phi \geq 1/\lfloor \frac{T}{\tau} \rfloor$ to replace $\frac{T}{\rho^j_\Phi} \leq T$ in Eq.4, where $\lfloor \frac{T}{\tau} \rfloor$ is the floor of $\frac{T}{\tau}$. The following sections will show that our algorithms can still work well in this case.
Algorithm 1 The gDUR Algorithm

Require: $\mathcal{U}, S, \mathcal{P}, C, \tau, T$

Ensure: $\Phi$

1: $\Phi = \emptyset$;
2: while $f(\Phi) < \frac{m\tau\theta}{T}$ do
3: Select a user $u_i \in \mathcal{U} \setminus \Phi$ to maximize $\frac{f(\Phi \cup \{u_i\}) - f(\Phi)}{c_i}$;
4: $\Phi = \Phi \cup \{u_i\}$;
5: return $\Phi$;

In addition, we extend the DUR problem to a more practical case. We leave the extended DUR problem to be discussed in Section V, for the integrity of description. Here, we also list the main notations in Table I.

IV. DEADLINE-SENSITIVE USER RECRUITMENT

In this section, we first analyze the complexity of the DUR problem. Then, we propose the gDUR algorithm based on a utility function, followed by an example. Finally, we prove the correctness and approximation ratio of the algorithm.

A. Problem Hardness Analysis

Before the solution, we first prove that the DUR problem is NP-hard, as shown in the following theorem.

**Theorem 1:** The DUR problem is NP-hard.

*Proof:* We consider a special case of the DUR problem: given a probabilistically collaborative mobile crowdsensing $(\mathcal{U}, \mathcal{P}, C)$ and a deadline $\tau$, where $\mathcal{P} = \{p_j = p \mid 0 < p \leq 1, u_i \in \mathcal{U}, s_j \in \mathcal{S}_1\}$, $C = \{c_i = 1 \mid u_i \in \mathcal{U}\}$, and $\tau = \tau$, determine a set $\Phi \subseteq \mathcal{U}$, such that the requester can minimize $C(\Phi) = \sum_{u_i \in \Phi} c_i = |\Phi|$, while the expected processing time of each task is no larger than $\tau$. Actually, this special DUR problem is to select the minimum number of users from $\mathcal{U}$ who can process all tasks in $\mathcal{S}$. When we replace each $u_i$ in $\mathcal{U}$ by using $\mathcal{S}_i$ $(\subseteq \mathcal{S})$, i.e., the set of tasks that $u_i$ can process, this problem can be seen as a set cover problem, a well-known NP-hard problem: given a task set $\mathcal{S}$, a collection of subset $\{\mathcal{S}_i|1 \leq i \leq n\}$, find a minimum size of subcollection of $\{\mathcal{S}_i|1 \leq i \leq n\}$ that covers all tasks in $\mathcal{S}$. That is to say, the special DUR problem is NP-hard. Consequently, the general DUR problem is also at least NP-hard. The theorem holds.

B. The gDUR Algorithm

Since DUR is NP-hard, we propose a greedy algorithm to solve it. The greedy criterion is that the user who has the largest probability to process the most tasks with the least cost will be recruited and added into the set $\Phi$ first, which is based on the following utility function:

**Definition 1:** Utility function $f(\Phi)$ indicates the utility about the total probability of the users in set $\Phi$ processing all tasks in $\mathcal{S}$ before the deadline, defined as follows:

$$f(\Phi) = \theta \sum_{j=1}^{m} \min \left\{ \rho_j \cdot \frac{\tau}{\tau} \right\},$$

where $\theta = \max \{\theta_1, \theta_2\}$ is a constant related to the approximation factor of the gDUR algorithm, in which $\theta_1 = \frac{\tau}{\tau} \sum_{m/2}^{m} \rho_j$, and $\theta_2 = \max \left\{ \frac{\tau}{\tau} \sum_{i=1}^{n} \rho_j | 1 \leq i \leq n, \rho_j < \frac{\tau}{\tau} \right\}$.

By using this defined utility function, we can turn our problem into an MSC/SC problem. We will demonstrate this in the following subsections. Here, we only present the gDUR algorithm based on this utility function, as shown in Algorithm 1. The gDUR algorithm starts from an empty user set $\Phi$. In each round, it adds the user having the maximum $(f(\Phi \cup \{u_i\}) - f(\Phi))/c_i$ value into $\Phi$. The algorithm terminates when $f(\Phi) < \frac{m\tau\theta}{T}$. The computation overhead is dominated by Step 3, which is $O(n^2m)$.

In addition, there is a small trick in the gDUR algorithm. Note that $\theta$ is a constant. It is only related to the approximation factor, and will only be used in the theoretical analysis. Hence, we can simply let $\theta = 1$ in the real implementation of gDUR, since it will not change the comparison results in Steps 2 and 3, and also will not change the final result.

C. Example

To better understand Algorithm 1, we use an example shown in Fig. 3 to illustrate the user recruitment procedure. In the example, $\tau = 1$ day, $T = 2$ days, $\mathcal{U}, \mathcal{S}, \mathcal{P}$ and $C$ are marked in Fig. 3(a). According to Definition 1, $\theta = 15$, and $\frac{m\tau\theta}{T} = 22.5$. Then, the algorithm is conducted as follows:

1. First round: $\Phi = \emptyset$.
2. Second round: Since $\Phi = \emptyset$ and $f(\emptyset) = 0 < 22.5$, we first compute $f(\emptyset \cup \{u_1\}) = 6.75$. Likewise, we have $f(\emptyset \cup \{u_2\}) = 4.5$, and $f(\emptyset \cup \{u_3\}) = 6.75$. Both $u_1$ and $u_3$ can maximize $f(\Phi \cup \{u_i\}) - f(\Phi)$.
3. Third round: Since $\Phi = \{u_1\}$ this time and $f(\{u_1\}) = 13.5 < 22.5$, we continue the user recruitment procedure by computing $f(\{u_1, u_2\}) - f(\{u_1\}) = 2$. Moreover, we have $f(\{u_1, u_3\}) - f(\{u_1\}) = 4.5$. Since $u_3$ maximizes $f(\Phi \cup \{u_i\}) - f(\Phi)$ in this round, we add $u_3$ into $\Phi$. Now, $\Phi = \{u_1, u_3\}$ and $f(\Phi) = 22.5$. The algorithm terminates and returns the recruited user set $\Phi = \{u_1, u_3\}$.

Fig. 3(b) shows the result $\Phi = \{u_1, u_3\}$ for this user recruitment. It is easy to check that this result is a correct solution. For this example, this solution is even optimal. Moreover, when we let $\theta = 1$, we can get the same result.

D. Correctness

Before the correctness analysis, we first prove an important property of the defined utility function.

**Theorem 2:** 1) $f(\emptyset) = 0$; 2) $f(\Phi)$ is an increasing function.

*Proof:* 1) If $\Phi = \emptyset$, then $\rho_\Phi = 0$ for each $s_j \in \mathcal{S}$, according to Eq. 1. Thus, $f(\emptyset) = 0$, according to Definition 1.
2) Without loss of generality, we consider two user sets $\Phi_1$ and $\Phi_2$, where $\Phi_1 \subseteq \Phi_2$. According to Eq. 1, we have $\rho_j^{\Phi_1} \leq \rho_j^{\Phi_2}$ for each $s_j \in S$. Then, $\min\{\rho_j^{\Phi_1}, \frac{T}{\tau}\} \leq \min\{\rho_j^{\Phi_2}, \frac{T}{\tau}\}$. Consequently, we have $f(\Phi_1) = \theta \sum_{j=1}^{m} \min\{\rho_j^{\Phi_1}, \frac{T}{\tau}\} \leq \theta \sum_{j=1}^{m} \min\{\rho_j^{\Phi_2}, \frac{T}{\tau}\} = f(\Phi_2)$. Therefore, $f(\Phi)$ is an increasing function.

Based on the monotone increasing property of the utility function, we can derive the correctness of the gDUR algorithm in the following theorem.

**Theorem 3:** Algorithm 1 is correct. That is, it will produce a feasible solution of the DUR problem, as long as the problem is solvable. More specifically, 1) Algorithm 1 will terminate for sure; 2) $f(\Phi) = \frac{m_{\Theta}}{\tau}$ if and only if $\Phi$ is a user set who can process each task in $S$ with an expected completion time no larger than the deadline $T$.

**Proof:** 1) For Algorithm 1, in each round of iteration, a user will be added into the recruited user set $\Phi$. In the worst case, all users are recruited after $n$ rounds of iteration. Then, we have $f(\Phi) = f(\cup) = \frac{m_{\Theta}}{\tau}$, and the algorithm will terminate. 2) $\Rightarrow$: According to Definition 1, $f(\Phi) = \frac{m_{\Theta}}{\tau}$ only when $\min\{\rho_j^{\Phi}, \frac{T}{\tau}\} = \frac{T}{\tau}$ for all $j \in [1, m]$. This means that $\rho_j^{\Phi} \geq \frac{T}{\tau}$ for all $j \in [1, m]$. Thus, for the arbitrary task $s_j$ in $S$, the expected completion time satisfies $\frac{T}{\tau} \leq T$.

$\Leftarrow$: If $\Phi$ is a set of users who can process each task in $S$ with an expected completion time no larger than $T$, we have $\rho_j^{\Phi} \geq \frac{T}{\tau}$ for all $j \in [1, m]$. Consequently, $\min\{\rho_j^{\Phi}, \frac{T}{\tau}\} = \frac{T}{\tau}$ for all $j \in [1, m]$. Thus, $f(\Phi) = \frac{m_{\Theta}}{\tau}$ according to Definition 1.

**Note:** Note that $f(\cup) = \frac{m_{\Theta}}{\tau}$. Then, based on Theorem 3, we can equivalently replace the constraint Eq. 4 by using $f(\Phi) = f(\cup)$. That is, we have:

**Corollary 1:** The DUR problem can be equivalently reformulated as:

$$\min C(\Phi) | f(\Phi) = f(\cup), \Phi \subseteq \cup \}.$$  

(6)

**E. Performance Analysis**

To analyze the approximation ratio of the proposed gDUR algorithm, we first show that our DUR problem can be categorized as the MSC/SC problem. Actually, according to [17], a problem can be seen as an MSC/SC problem, if: 1) the problem can be formalized as Eq. 6, i.e., $\min C(\Phi) | f(\Phi) = f(\cup), \Phi \subseteq \cup \} ;$ 2) $f(\Phi)$ is a polymatroid function on $2^d$; 3) $f(\Phi)$ is also a polymatroid function on $2^d$. In the following, we first prove the polymatroid property of $f(\Phi)$.

**Theorem 4:** $f(\Phi)$ is a submodular function. More specifically, for two arbitrary user sets $\Phi_1$ and $\Phi_2$, $\Phi_1 \subseteq \Phi_2$, and $\forall u_k \in \cup \setminus \Phi_2$, the submodular property holds, i.e., $f(\Phi_1 \cup \{u_k\}) - f(\Phi_1) \geq f(\Phi_2 \cup \{u_k\}) - f(\Phi_2)$.

**Proof:** See Appendix A.

**Theorem 5:** $f(\Phi)$ is a polymatroid function on $2^d$.

**Proof:** According to Theorems 2 and 4, $f(\Phi)$ is an increasing, submodular function with $f(\emptyset) = 0$. Hence, we conclude that $f(\Phi)$ is a polymatroid function on $2^d$.

Furthermore, we have that $C(\Phi)$ is also polymatroid:

**Theorem 6:** $C(\Phi)$ is a modular function as well as a polymatroid function on $2^d$.

**Proof:** $C(\Phi)$ is said to be modular if and only if, for two arbitrary user sets $\Phi_1$ and $\Phi_2$, $C(\Phi)$ satisfies the equation: $C(\Phi_1) + C(\Phi_2) = C(\Phi_1 \cap \Phi_2) + C(\Phi_1 \cup \Phi_2)$. Since $C(\Phi) = \sum_{u_j \in \Phi} \rho_j$, it is straightforward to verify that the equation holds. Hence, $C(\Phi)$ is a modular function, which implies the submodular property. Moreover, according to $C(\Phi) = \sum_{u_j \in \Phi} \rho_j$, we have that $C(\Phi)$ is an increasing function with $f(\emptyset) = 0$. Thus, $C(\Phi)$ is also a polymatroid function.

According to Corollary 1 and Theorems 5, 6, we can conclude that our DUR problem is an MSC/SC problem. For MSC/SC problems, [17] has proposed a lemma, which can be used to analyze the approximation performance of the proposed gDUR algorithm.

**Lemma 1:**[17] Consider an MSC/SC problem: $\min \{C(\Phi) | f(\Phi) = f(\cup), \Phi \subseteq \cup \}$, in which $f$ is a polymatroid real function on $2^d$, and $f(\cup) \geq \omega$ where $\omega$ is the cost of a minimum submodular cover. For a greedy algorithm of this problem, if the selected $u_i$ in each iteration always satisfies $f(\cup \setminus \{u_i\}) - f(\Phi) \geq 1$, then the greedy solution is a $(1 + \frac{\ln f(\cup)}{\omega})$-approximation, where if $C(\Phi)$ is modular, then $\omega = 1$.

Before using Lemma 1 to analyze the approximation ratio of the proposed gDUR algorithm, we first show that our utility function $f(\Phi)$ meets the constraints in Lemma 1, as follows:

**Theorem 7:** Our utility function $f(\Phi)$ satisfies: 1) $f(\cup) \geq \omega$, where $\omega$ is the cost of the optimal submodular cover of the DUR problem; 2) in each iteration of Algorithm 1, the selected user $u_i$ satisfies $f(\cup \setminus \{u_i\}) - f(\Phi) \geq 1$.

**Proof:** See Appendix B.

Based on the above analysis, we can give the approximation ratio of the gDUR algorithm by the following theorem.

**Theorem 8:** The proposed gDUR algorithm can achieve a $(1 + \frac{\ln m_{\Theta}}{\omega})$-approximation solution, where $\omega$ is the cost of the optimal solution for the DUR problem.

**Proof:** According to Lemma 1 and Theorem 7, gDUR can achieve a $(1 + \frac{\ln f(\cup)}{\omega})$-approximation solution, where $\omega$ is the cost of the optimal solution. According to Theorem 6 and Lemma 1, we have $\omega = 1$. Thus, the approximation ratio is $1 + \frac{\ln f(\cup)}{\omega} = 1 + \frac{\ln m_{\Theta}}{\omega}$. The theorem holds.

**V. EXTENSION**

In this section, we extend our problem and algorithm to a more practical scenario, in which sensing duration is taken into consideration. We first introduce the extended problem, and then, propose the eDUR algorithm to solve this problem, followed by the performance analysis.

**A. The Extended Problem**

Assume that when a mobile user $u_i$ in $\cup$ passes by a PoI to perform the related sensing task $s_j$ ($s_j \in S$), there is an average sensing duration $d_{ij}$ (minutes or hours) in each sensing cycle, only during which the user $u_i$ can collect the sensing data. Then, the expected sensing duration of user $u_i$ performing task $s_j$ is $d_{ij} p_{ij} T / \tau$, where $p_{ij} T / \tau$ is actually the expected number of sensing cycles for user $u_i$ performing task $s_j$ before
Algorithm 2 The dDUR Algorithm

Require: \( \mathcal{U}, \mathcal{S}, \mathcal{P} = \{ p_{ij} \mid u_i \in \mathcal{U}, s_j \in \mathcal{S} \}, \mathcal{C}, \tau, T, D \)

Ensure: \( \Phi \)

1. \( \Phi = \emptyset \);
2. while \( h(\Phi) < \frac{m \tau \theta}{D} + \vartheta \) do
3. Select a user \( u_i \in \mathcal{U} \setminus \Phi \) to maximize \( \frac{h(\Phi \cup \{u_i\}) - h(\Phi)}{c_i} \);
4. \( \Phi = \Phi \cup \{u_i\} \);
5. return \( \Phi \);

B. The dDUR algorithm

We adopt the same strategy in the last section to solve the problem. First, we define another utility function for the constraint of sensing duration (i.e., Eq. 10):

\[
g(\Phi) = \frac{\vartheta}{mD} \sum_{j=1}^{m} \min\{\sigma_j^\Phi, D\},
\]

where \( \vartheta = \theta \) if \( D > 0 \); \( \vartheta = 0 \) and \( g(\Phi) = 0 \), if \( D = 0 \).

Taking both constraints (i.e., Eqs. 10 and 11) into consideration, we combine the two utility functions as follows:

Definition 3: Combinational utility function \( h(\Phi) \) is the combination of utility functions \( f(\Phi) \) and \( g(\Phi) \). That is,

\[
h(\Phi) = f(\Phi) + g(\Phi)
\]

Based on the combinational utility function \( h(\Phi) \), we present the dDUR algorithm, as shown in Algorithm 2. The dDUR algorithm also starts from an empty user set \( \Phi \). In each round, it adds the user having the maximum \( c_i \) value into \( \Phi \). The algorithm terminates when \( h(\Phi) = \frac{m \tau \theta}{D} + \vartheta \). The computation overhead is still \( O(n^2m) \).

C. Example

We use the example in Fig. 4 to illustrate the dDUR algorithm. In the example, all \( d_{ij} \)’s are marked in Fig. 4(a), and \( D = 3 \), with units displayed in minutes. Other parameters are the same as those in Fig. 3(a). In this example, \( \theta = 15 \), and \( \frac{m \tau \theta}{D} + \vartheta = 37.5 \). Then, the algorithm is conducted as follows:

1. First round: \( \Phi = \emptyset \) and \( h(\emptyset) = 0 \).
2. Second round: We have \( h(\{u_1\}) = 19.5 < 37.5 \), and \( h(\{u_2\}) = 7.5 \). Both \( u_1 \) and \( u_3 \) can maximize \( h(\Phi \cup \{u_i\}) - h(\Phi) \). Then, we add \( u_1 \) into \( \Phi \).
3. Third round: Due to \( \Phi = \{u_1\} \) and \( h(\{u_1\}) = 19.5 < 37.5 \), we continue the user recruitment procedure by computing \( h(\{u_1, u_2\}) - h(\{u_1\}) = 4.33 \) and \( h(\{u_1, u_3\}) - h(\{u_1\}) = 7.33 \). Since \( u_3 \) maximizes \( h(\Phi \cup \{u_i\}) - h(\Phi) \) in this round, we add \( u_3 \) into \( \Phi \).
4. Fourth round: Now, \( \Phi = \{u_1, u_3\} \) and \( h(\{u_1, u_3\}) = 34.17 < 37.5 \). Then, we add the last user \( u_2 \) into \( \Phi \), to get that \( \Phi = \{u_1, u_2, u_3\} \) and \( h(\{u_1, u_2, u_3\}) = 37.5 \). The algorithm terminates and returns the recruited user set \( \Phi = \{u_1, u_2, u_3\} \).

Fig. 4(b) shows the result \( \Phi = \{u_1, u_2, u_3\} \) for this user recruitment procedure. It is easy to check that this result is a correct solution. Moreover, for this example, this solution is the only feasible solution.

D. Performance Analysis

First, we prove the correctness of the dDUR algorithm.

Theorem 9: 1) \( h(\Phi) \) is an increasing function with \( h(\emptyset) = 0 \); 2) \( h(\Phi) = \frac{m \tau \theta}{D} + \vartheta \) if and only if \( \Phi \) is a feasible solution of the extended DUR problem.

Proof: 1) According to Eqs. 7 and 12, \( g(\emptyset) = 0 \), and \( g(\Phi) \) is an increasing function. Hence, \( h(\Phi) \) is an increasing function with \( h(\emptyset) = 0 \), according to Theorem 2 and Definition 3.

2) Since \( f(\Phi) \leq \frac{m \tau \theta}{D} \) and \( g(\Phi) \leq \vartheta \) according Eqs. 5 and 12, we have \( h(\Phi) = \frac{m \tau \theta}{D} + \vartheta \) if and only if \( f(\Phi) = \frac{m \tau \theta}{D} \) and \( g(\Phi) = \vartheta \), which means that \( \Phi \) satisfies the constraints of Eqs. 11 and 10, respectively. Thus, \( h(\Phi) = \frac{m \tau \theta}{D} + \vartheta \) if and only if \( \Phi \) is a feasible solution.

Theorem 9 shows that when the dDUR algorithm terminates, i.e., \( h(\Phi) = \frac{m \tau \theta}{D} + \vartheta \), it will produce a feasible solution. This implies the correctness of dDUR. Moreover, since \( h(\Phi) = \frac{m \tau \theta}{D} + \vartheta \), we can also re-formalize the extended problem, like Corollary 1.

Corollary 2: The extended DUR problem can be equivalently re-formalized as:

\[
\text{Minimize}\{C(\Phi) \mid h(\Phi) = h(\Phi), \Phi \subseteq \mathcal{U}\}.
\]

Now, we prove the polymatroid property of \( h(\Phi) \). First, we have the following theorem:

Theorem 10: \( g(\Phi) \) is a submodular function. More specifically, for two arbitrary user sets \( \Phi_1 \) and \( \Phi_2 \), \( \Phi_1 \subseteq \Phi_2 \), and \( \forall u_k \in \mathcal{U} \setminus \Phi_2 \), the submodular property holds, i.e., \( g(\Phi_1 \cup \{u_k\}) - g(\Phi_1) \geq g(\Phi_2 \cup \{u_k\}) - g(\Phi_2) \).
Successful Processing Ratio

Total Cost

VI. Evaluation

We conduct extensive simulations to evaluate the performances of the proposed algorithms. The compared algorithms, the traces that we used, the simulation settings, and the results are presented as follows.

A. Algorithms in Comparison

In order to evaluate our algorithms, we implement two other user recruitment algorithms for comparison: MCUR (Minimum Cost User Recruitment) and MCURP (MCUR with Probabilistic mobility). As we discussed in Section II, our problem is different from the existing works, previous user recruitment algorithms cannot be directly applied in our problem. Hence, MCUR and MCURP are designed, mainly based on the idea of the most related algorithms in [7], [10]. Both of them start from an empty user set $\Phi$. In each round, MCUR adds the user $u_k$, who can maximize the incremental tasks with minimum cost, i.e., $\sum_{u_j \in \Phi \cup \{u_k\}} |\Delta| - |\Delta_{\Phi}|$, into the user set $\Phi$, until all tasks are covered, i.e., $U_{\Phi} = S$. In contrast, MCURP adds the user $u_k$, who can maximize the effective increments of the probabilities of tasks being processed with minimum cost, i.e., $\sum_{i=1}^{|S|} \min\{\sum_{u_j \in \Phi \cup \{u_k\}} P_{ij}, 1\} - \sum_{i=1}^{|S|} \min\{\sum_{u_j \in \Phi} P_{ij}, 1\}$, into the user set $\Phi$, until the additive sum probabilities of all tasks being processed reach 1, i.e., $\min\{\sum_{u_j \in \Phi} P_{ij}, 1\} = 1$ for $\forall s_j \in S$.

B. The Traces Used and Settings

We adopt the widely-used Cambridge Haggle Trace Set [13], which has also been used in [10], [21]. This trace set includes a total of five traces of Bluetooth device connections by people carrying mobile devices (iMotes) over a certain number of days. Among the five traces, we use the trace, generally called Infocom2006, in our simulations, since this trace contains some fixed nodes. These can be seen exactly as the Pols in our model, and it can also provide adequate covers to these fixed nodes. More specifically, the Infocom2006 trace contains 78 iMotes carried by Infocom 2006 conference participants and 20 fixed nodes situated at various places in the conference hotel, such as conference rooms, the bar, the concierge, and so on.

In our simulations, the set of mobile nodes in the Infocom2006 trace is mapped to the user set $U$, and the fixed nodes are mapped to the Pols as well as the task set $S$. Moreover, we set the sensing cycle $\tau$ as an hour. Then, we extract parts of the trace, and estimate each probability $p_{ij}$ as the average probability of the tasks belonging to the Pols at these two points. In addition, we generate the cost $c_i$ for each mobile user, which is randomly selected from a cost range $[10, 30]$.

Since the scale of the real trace is very limited, we also randomly generate some synthetic traces in order to make our evaluation more convincing. More specifically, $|U|$ is selected from $\{100, 200, 300, 400\}$ and $|S|$ is selected from $\{20, 40, 60, 80\}$. Each sensing duration $d_{ij}$ is randomly selected from a time range $[0.1, 4]$ minutes, and the sensing cycle $\tau$ is set as one day.
Successful Processing Ratio

**C. Evaluation Metrics, Methods and Results**

The major metrics in our simulations include the total cost and the successful processing ratio. The total cost is the total cost that the requester needs to pay to all recruited users. The successful processing ratio is the ratio of the number of successfully processed crowdsensing jobs and all crowdsensing jobs. Here, a successfully processed crowdsensing job means that the completion time is no larger than the deadline, and the total sensing duration is no less than the given threshold.

To evaluate the performances of gDUR, dDUR, MCUR, and MCURP, we first conduct two groups of simulations by using the Infocom2006 trace. In the first group of simulations, we set the deadlines as 10, 15, 20, 25 hours, respectively, and ignore the constraint of sensing duration by setting \( D = 0 \). In the second group of simulations, we let \( D = 4 \) minutes. The total costs and successful processing ratios are depicted in Figs. 5(a)-5(d). When \( D = 0 \), the dDUR algorithm will be degraded to gDUR, and thus, they achieve the same results. When \( D = 4 \) minutes, dDUR has larger successful processing ratios than gDUR, while gDUR has smaller total costs than dDUR, since dDUR takes the sensing duration into consideration, so that it recruits more users than gDUR, resulting in larger total costs as well as higher successful processing ratios. Moreover, the results show that gDUR and dDUR have about 96.7% and 92.4% smaller total costs than MCURP, and also have about 166% and 227% larger successful processing ratios than MCUR, respectively. This is because our algorithms take into consideration both the cost performance and the successful processing ratio. The results show that dDUR and gDUR demonstrate much better integrative performances than the two compared algorithms.

In addition, we also find that the MCUR algorithm has achieved even fewer costs than our algorithms. This is because MCUR does not consider the deadline constraint, so that it recruits many fewer users than our algorithms. Although it produces fewer total costs, it also results in very low successful processing ratios, as shown in Figs. 5(a) and 5(c). Most crowdsensing jobs cannot be completed before the deadlines. On the other hand, we also find that the MCURP algorithm has achieved even larger successful processing ratios than our algorithms in some simulations. This is due to the reason that, in order to ensure the additive sum of probabilities of each task being covered to be larger than 1, MCURP recruits many more users than our algorithms. Although recruiting more users can achieve larger successful processing ratios, it also leads to very large total costs, as shown in Figs. 5(b) and 5(d). Actually, in some simulations, MCURP even recruits all users, resulting in low efficiency. In summary, our algorithms demonstrate much better integrative performances than the compared algorithms.

Second, we also evaluate the performances of the four algorithms by using synthetic traces. In the first group of simulations, we set the deadlines as 2, 4, 6, 8 days, respectively, and let \( D = 0 \). In the second group of simulations, we set \( D = 4 \) minutes. In these simulations, each \( c_i \in C \) is randomly selected from \([10, 40]\), and each \( p_{ij} \in P \) is randomly selected from \([0, 0.2]\). The total costs and successful processing ratios are depicted in Figs. 6(a)-6(d). Like the evaluation over the Infocom2006 trace, the results of both groups’ simulations show that gDUR and dDUR have about 67.6% and 59.0% smaller total costs than MCURP, and also have about 12.2 and 17.0 times larger successful processing ratios than MCUR, respectively, demonstrating much better integrative performances.

Third, we also conduct two additional groups of simulations on the synthetic traces by changing \( P \) and \( C \). In the first group of simulations, we let each \( c_i \in C \) be randomly selected from \([10, 20], [10, 40], [10, 60], [10, 80]\), and \( p_{ij} \in P \) be randomly selected from \([0, 0.1]\). In the second group of simulations, we let each \( p_{ij} \in P \) be randomly selected from \([0, 0.1], [0, 0.2], [0, 0.3], [0, 0.4]\). Each \( c_i \in C \) is randomly selected from \([10, 80]\). Moreover, in these simulations, we set \( T = 6 \) days and \( D = 10 \) minutes. The results are shown in Figs. 7(a)-7(d), from which we can derive that gDUR and dDUR have about 75.0% and 46.2% smaller total costs than MCURP, and also have about 4.33 and 10.9 times larger successful processing ratios than MCUR, respectively. Both gDUR and dDUR demonstrate much better integrative performances than the two compared algorithms.

**VII. CONCLUSION**

We study the DUR problem in the probabilistically collaborative mobile crowdsensing. First, we formalize this problem as a set cover problem with non-linear programming constraints, and prove its NP-hardness. Then, we design a submodular utility function, based on which we propose the greedy approximation algorithm gDUR, and derive the corresponding approximation ratio. Moreover, we extend the DUR problem to the case, where sensing duration is taken into consideration, and we propose another approximation algorithm dDUR, followed by the analysis of approximation ratio. Extensive simulations based on a real trace and a synthetic trace also verify the performances of the two algorithms.
ACKNOWLEDGMENT

This research was supported in part by the National Natural Science Foundation of China (NSFC) (Grant No. 61572457, 61379132, 61502261, 61303206, 61572342), NSF grants CNS 1449860, CNS 1461932, CNS 1460971, CNS 1439672, CNS 13011774, ECCS 1231461, and the NSF of Jiangsu Province in China (Grant No. BK20131174, BK2009150).

REFERENCES


Appendix

A. Proof of Theorem 4

We first prove that when $|\Phi_2| - |\Phi_1| = 1$, $f(\Phi_1 \cup \{u_k\}) - f(\Phi_1) \geq f(\Phi_2 \cup \{u_k\}) - f(\Phi_2)$. Then, we extend it to the general case where $|\Phi_2| - |\Phi_1| = \omega > 1$.

First, without loss of generality, we let $\Phi_2 \setminus \Phi_1 = \{u_h\}$ according to the assumption $\Phi_1 \subseteq \Phi_2$ and $|\Phi_2| - |\Phi_1| = 1$. To prove the submodular property of $f(\Phi)$, we consider the joint successful processing probability of $\forall s_j \in S$, which can be divided into the following three cases:

Case 1: $u_k$ cannot process task $s_j$, i.e., $u_k \notin U_k$. For this case, $p_{kj} = 0$. Then, we have $\Phi_{1\cup\{u_k\}} = \Phi_1$, $\Phi_{2\cup\{u_k\}} = \Phi_2$, according to Eq. 1. As a result, $\min(\rho_{2}, \frac{T}{\tau}) - \min(\rho_{1}, \frac{T}{\tau}) = \min(\rho_{2}, \frac{T}{\tau}) - \min(\rho_{1}, \frac{T}{\tau}) = 0$.

Case 2: $u_k$ can process task $s_j$, but $u_h$ cannot process this task, i.e., $u_h \in U_h$, and $u_k \notin U_k$. For this case, $p_{bh} = 0$. According to Eq. 1, $\Phi_{2} = \Phi_{1 \cup \{u_k\}} = \Phi_1$, and $\Phi_{2 \cup \{u_k\}} = \Phi_{1 \cup \{u_k, u_h\}} = \Phi_{1 \cup \{u_k\}}$. Consequently, we can get $\min(\rho_{2}, \frac{T}{\tau}) - \min(\rho_{1}, \frac{T}{\tau}) = \min(\rho_{1}, \frac{T}{\tau}) - \min(\rho_{1}, \frac{T}{\tau}) = 0$.

Case 3: Both $u_k$ and $u_h$ can process task $s_j$, i.e., $u_k, u_h \in U_j$. We divide this case into two sub-cases: $p_{kj} \geq p_{hj}$ and $p_{kj} < p_{hj}$. According to Eq. 1, $p_{kj} \geq p_{hj}$ means $\Phi_{2 \cup \{u_k\}} \geq \Phi_{1 \cup \{u_k, u_h\}}$, otherwise, $p_{hj} < p_{kj}$ means $\Phi_{2 \cup \{u_k, u_h\}} < \Phi_{1 \cup \{u_k, u_h\}}$.

For the first sub-case, we have $\Phi_{2 \cup \{u_k, u_h\}} \geq \Phi_{1 \cup \{u_k, u_h\}} \geq \Phi_{1 \cup \{u_k\}} \geq \Phi_1$. Then, we can get:

$$\min(\rho_{2}, \frac{T}{\tau}) - \min(\rho_{1}, \frac{T}{\tau}) \geq 0$$

$$\min(\rho_{2}, \frac{T}{\tau}) - \min(\rho_{1}, \frac{T}{\tau}) = \min(\rho_{2}, \frac{T}{\tau}) - \min(\rho_{1}, \frac{T}{\tau}) = 0$$

Therefore, when $\omega > 1$, we have $p_{kj} \geq p_{hj}$, we have $p_{kj}(1 - \rho_{2}) = p_{kj}(1 - \rho_{1}) - p_{hj}(\frac{T}{\tau}) < 0$; when $p_{hj} \leq \rho_{2}$, we can get $p_{kj}(1 - \rho_{2}) - (\frac{T}{\tau} - \rho_{2} \Phi_{1 \cup \{u_k\}}) = p_{kj}(1 - \rho_{2}) - \rho_{2} \Phi_{1 \cup \{u_k\}} - \frac{T}{\tau} < 0$ and when $p_{hj} > \rho_{2}$.

Note that, when $\frac{T}{\tau} \geq \frac{T}{\tau} - \rho_{2} \Phi_{1 \cup \{u_k\}}$, we have $p_{kj}(1 - \rho_{2}) = p_{kj}(1 - \rho_{1}) - p_{hj}(\frac{T}{\tau}) < 0$; when $p_{hj} \leq \rho_{2}$, we can get $p_{kj}(1 - \rho_{2}) - (\frac{T}{\tau} - \rho_{2} \Phi_{1 \cup \{u_k\}}) = p_{kj}(1 - \rho_{2}) - \rho_{2} \Phi_{1 \cup \{u_k\}} - \frac{T}{\tau} < 0$ and when $p_{hj} > \rho_{2}$.
\( \bar{r} \geq \rho_j \cup \{u_k, u_k\} \), we can obtain \( \bar{r} - \rho_j - (\bar{r} - \rho_j \cup \{u_k, u_k\}) = \rho_j (1 - \rho_j) > 0 \), according to Definition 1. Then, comparing Eqs. 15 and 16, we have: \( \min \{\rho_j \cup \{u_k, u_k\}, \bar{r}\} - \min \{\rho_j, \bar{r}\} \geq \min \{\rho_j \cup \{u_k, u_k\}, \bar{r}\} - \min \{\rho_j, \bar{r}\} \).

For the second sub-case, we have \( \rho_j \cup \{u_k, u_k\} \geq \rho_j \cup \{u_k, u_k\} \geq \rho_j \). Then, according to Definition 1, we can get:

\[
\min \{\rho_j \cup \{u_k, u_k\}, \bar{r}\} - \min \{\rho_j, \bar{r}\} = \min \{\rho_j \cup \{u_k, u_k\}, \bar{r}\} - \min \{\rho_j, \bar{r}\}.
\]

Let \( s_j \) be such a task. Then, \( \rho_j' < \bar{r} \), while \( \rho_j \cup \{u_k\} \geq \bar{r} \).

Moreover, we have:

\[
\frac{f(\Phi \cup \{u_k\}) - f(\Phi)}{c_h} \geq \frac{f(\Phi' \cup \{u_k\}) - f(\Phi')}{c_h} \quad (\text{the submodular property of } f(\Phi))
\]

\[
\frac{\theta(\rho_j' - \rho_j)}{\bar{r} - \rho_j'} \geq \frac{\theta(\bar{r} - \rho_j)}{\bar{r} - \rho_j'} \geq 1.
\]

Therefore, the theorem is correct.

**C. Proof of Theorem 10**

To prove the submodular property of \( g(\Phi) \), we compare \( \min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\} \) and \( \min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\} \) for \( \forall s_j \in S \).

We first consider the case \( u_k \notin U_j \). According to Eq. 7, we have \( \sigma_{j1} \cup \{u_k\} = \sigma_{j1} \cup \{u_k\} \) and \( \sigma_{j1} \cup \{u_k\} = \sigma_{j2} \cup \{u_k\} \). Thus, \( \min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\} = \min \{\sigma_{j2} \cup \{u_k\} - \min \{\sigma_{j1}, \Phi\} \).

Then, we consider the case \( u_k \in U_j \). It is simple to verify that \( \sigma_{j1} \) is an increasing function about \( \Phi \). Thus, this case can be divided into two sub-cases: \( \sigma_{j1} \leq \sigma_{j1} \cup \{u_k\} \leq \sigma_{j2} \cup \{u_k\} \) and \( \sigma_{j1} \leq \sigma_{j2} \cup \{u_k\} \leq \sigma_{j2} \cup \{u_k\} \).

For the first sub-case \( \sigma_{j1} \leq \sigma_{j1} \cup \{u_k\} \leq \sigma_{j2} \cup \{u_k\} \), we have:

\[
\min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\} = \min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\}.
\]

\[
\min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\} = \min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\}.
\]

\[
\min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\} = \min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\}.
\]

\[
\min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\} = \min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\}.
\]

\[
\min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\} = \min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\}.
\]

\[
\min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\} = \min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\}.
\]

\[
\min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\} = \min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\}.
\]

\[
\min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\} = \min \{\sigma_{j1} \cup \{u_k\}, \Phi\} - \min \{\sigma_{j1}, \Phi\}.
\]

Therefore, \( f(\Phi) \) is a submodular function.

**B. Proof of Theorem 7**

1) Note that \( \ell \) is a feasible solution. Hence, we have \( f(\ell) = \frac{\mathbf{w}^T \mathbf{d}}{\mathbf{w}^T \mathbf{d}} \). Consequently, \( f(\ell) \geq \frac{\mathbf{w}^T \mathbf{d}}{\mathbf{w}^T \mathbf{d}} = \sum_{i=1}^n c_i \geq \text{opt} \), where \( \text{opt} \) is the cost of the optimal solution of the DUR problem.

2) Without loss of generality, let \( u_k \) be the user recruited in the last round of iteration, and the recruited user set (except \( u_k \)) of this round be \( \Phi \). Then, we have \( \Phi \subseteq \Phi \). Moreover, there must be at least a task, whose joint successful processing probability becomes no less than \( \bar{r} \) in the last round of iteration; otherwise, the algorithm would have terminated before.