Mechanism Design for Mobile Crowdsensing with Execution Uncertainty

Zhenzhe Zheng, Zhaoxiong Yang, Fan Wu and Guihai Chen
Shanghai Key Laboratory of Scalable Computing and Systems
Department of Computer Science and Engineering
Shanghai Jiao Tong University, China
Email: zhengzhenzhe@sjtu.edu.cn, yangzhaoxiong@gmail.com, {fwu, gchen}@cs.sjtu.edu.cn

***Abstract*—Mobile crowdsensing has emerged as a promising paradigm for data collection due to increasingly pervasive and powerful mobile devices. There have been extensive research works that propose incentive mechanisms for crowdsensing, but they all make the assumption that mobile users will positively complete the allocated sensing taskes. In this paper, we consider a new and practical scenario of crowdsensing, where a user may fail to complete the task with a certain probability. It is an important and emerging issue for the incentive mechanisms to ensure fault tolerance for each sensing task under such unreliable scenarios.**

We design reverse auctions to model the strategic interaction between the platform and mobile users, in which users' probability of success and cost to perform the tasks are private information. Considering the task execution uncertainty, the goal of the auction mechanism is to minimize the social cost of user recruitment, while guaranteeing the tasks to be completed with high probability. We prove that minimizing the social cost is an NP-hard problem, and design computationally efficient mechanisms that achieve good approximation ratio and economic-robust properties, *e.g.*, strategy-proofness. We conduct extensive simulations to evaluate the performance of our mechanisms based on a real data set. The evaluation results show that our mechanisms outperform the heuristic algorithm and approach to the optimal solution.

I. Introduction

Recent years have witnessed the explosion of mobile devices equipped with fast network technologies and powerful sensors. Such advance gives rise to mobile crowdsensing – a new paradigm for data collection, which leverages the ubiquitous mobile devices and their embedded rich sensors to collect huge amount of data. In contrast to traditional sensor networks, mobile crowdsensing spares the effort of building hardware infrastructure, making it easy to deploy in a large scale. With these advantages, mobile crowdsensing has been widely applied in a number of applications, such as environment monitoring [1], [2], transportation and traffic planning [3], location services [4], and etc.

F. Wu is the corresponding author.

This work was supported in part by the State Key Development Program for Basic Research of China (973 project 2014CB340303). The work of Z. Zheng was also supported by Google PhD Fellowship and Microsoft Asia PhD Fellowship. This work was also supported in part by China NSF grant 61672348, 61672353, 61422208, and 61472252, in part by Shanghai Science and Technology fund 15220721300, and in part by the Scientific Research Foundation for the Returned Overseas Chinese Scholars. The opinions, findings, and conclusions expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies or the government.

When designing a crowdsensing system, incentive is a crucial issue we need to consider. The sensing tasks consume computational resources on mobile devices, require human effort, and sometimes incur privacy concerns. Thus, users are unwilling to participate in the campaign unless they receive enough rewards. Previous works have studied the incentive mechanisms for crowdsensing extensively [5]–[7]. However, most of the existing incentive mechanisms always assume that when a user is assigned a sensing task, she will successfully complete it, which ignores the potential unreliability in mobile crowdsensing. For example, in opportunistic sensing (one major type of mobile crowdsensing) [8], mobile devices continuously monitor the surrounding environmental context, and collect data only when users occasionally arrive at a location of interest. Within such scenario, due to the uncertainty of mobility, users may fail to complete the data collection tasks.

In this paper, we consider a scenario of task execution uncertainty, in which users only have a certain probability to successfully complete the assigned tasks. We refer such probability as the term of *probability of success (PoS)*. The causes to the failure of completing tasks may come from the uncertainty of mobility pattern, poor network connection during data transmission, or sensor hardware failure when collecting data. We assume that the user's device records several indicators, such as GPS traces, real-time network condition, and the running states of hardware, and is able to calculate the PoS using machine learning techniques [9]. Due to the privacy concern, the PoS is the private information to mobile users, and is not available to the platform [10]. Thus, the platform has to induce the users to truthfully reveal their PoS information, and then makes a global task allocation decision based on such information.

Considering the task execution uncertainty, the platform has to recruit enough mobile users to guarantee fault tolerance such that each task will be completed within a high probability. We incorporate such consideration into the incentive mechanism design. Our goal is to design mechanisms that minimize the social cost of users recruitment, while guaranteeing the probability requirement of each task, and satisfying good economical properties, such as strategy-proofness, which incentivizes users to truthfully reveal their private PoS's.

It is not an easy job to design the incentive mechanism with

execution uncertainty. We summarize the following two major design challenges.

The first design challenge comes from the strategic behaviors of mobile users over multiple private information. In mobile crowdsensing, the goal of selfish and rational users is to maximize their own utilities. Thus, the users attempt to misreport their private information, if doing so can increase their utilities. In the general setting, users have much flexibility to manipulate due to their multiple dimensions of private information, such as preferred task set, PoS, and sensing cost. Designing incentive mechanisms to resist such powerful manipulative behaviours belongs to the multi-dimensional mechanism design [11], which still has no universal solution. The few positive results in this filed, such as the combinatorial auction with single-minded users, have some restrictions on the cheating behaviours on the private parameters other than cost or valuation [12]. However, in our context, we do not have such restriction, because users can report arbitrary value for the PoS. Furthermore, even the classical VCG mechanism (named after Vickrey [13], Clark [14] and Groves [15]) cannot guarantee the strategy-proofness in terms of the PoS, because the payment scheme in VCG mechanism is only independent of the cost, but is depended on the PoS.

The second design challenge is the hardness of deriving the optimal social cost. Even without considering the strategic behaviours of users, minimizing social cost subjects to the probability requirements of tasks can be proved to be NP-hard. Thus, we search for the near-optimal and computationally efficient solution. However, the classical approximation algorithms may not guarantee the economic-robust properties. For example, the fully polynomial-time approximation scheme (FPTAS) for maximum knapsack problem is not monotone [16], which is a sufficient condition to achieve strategy-proofness. Therefore, it is nontrivial to simultaneously achieve both good approximation ratio and economic-robust properties in mechanism design with execution uncertainty.

In this paper, by jointly considering the above two design challenges, we propose a framework of strategy-proof mechanisms for the problem of user recruitment in mobile crowdsensing with task execution uncertainty. We consider the user recruitment process in two representative scenarios, *i.e.*, *single task* model and *multi-task* model, and formulate them as a minimum knapsack problem and a submodular set cover problem, respectively. To design strategy-proof incentive mechanisms, we first restrict the strategic behaviours of users only on the dimension of PoS, and transform multi-dimensional mechanism design in the general setting to single dimensional mechanism design in the domain of PoS. Inspired by execution contingent payment scheme [17], we then extend the payment scheme in VCG to the scenario of execution uncertainty, achieving the strategy-proofness with respective to the PoS. To derive the near-optimal social cost, we propose a FPTAS for the minimum knapsack problem in single task setting, and a greedy algorithm for the submodular set cover problem in multi-task setting. We theoretically prove that both these two winner determination algorithms have good approximation ratios and efficient time complexity. We summarize our contributions in this work as follows.

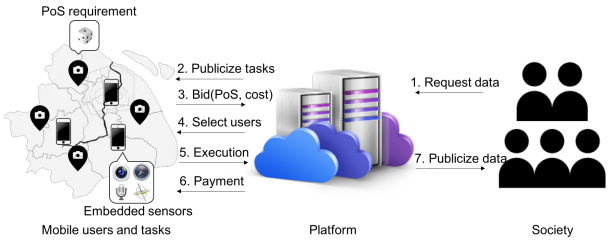


Fig. 1. Mobile Crowdsensing System Architecture.

• First, we formulate the problem of mechanism design for user recruitment in mobile crowdsensing with task execution uncertainty, which aims to minimize the social cost while guaranteeing each task to be successfully completed within a required probability.

• Second, we begin with considering a basic single task model, and propose a FPTAS for winner determination based on dynamic programming approach and scaling technique. Our analysis shows that this winner determination algorithm achieves a $(1+\epsilon)$ approximation ratio with the time complexity of $O(n^3/\epsilon)$, where n is the number of users and ϵ is an approximation parameter.

• Third, we further consider a general multi-task model. We design a greedy algorithm to iteratively select the most cost-efficient user, and prove that its approximation ratio is $O(H(\gamma))$ and the time complexity is $O(n^2t)$, where γ is the maximum PoS of users, $H(\gamma)$ is the γth harmonic numbers, and t is the number of tasks.

• Fourth, we extended the payment scheme in VCG to adopt to the scenario of task execution uncertainty. We theoretically prove that this payment scheme is computationally efficient, and guarantees the strategy-proofness of our mechanisms.

• The last but not least, we conduct extensive experiments to evaluate our incentive mechanisms based on a real data set of Shanghai taxi trace. The evaluation results show that our mechanisms outperform the heuristic algorithms, and approach to the optimal solution. Our mechanisms also guarantee that the task will be completed in a high probability, and resist the strategic behaviours of users.

The rest of the paper is organized as follows. In Section II, we describe our system model, and formulate our problem as a coverage problem. In Section III, we present the design challenges, give our mechanisms for both single task and multi-task cases, and analyze the approximation ratios and the economic-robust properties. In Section IV, we evaluate our mechanisms and report the results. In Section V, we discuss related works. Finally, we conclude our work in Section VI.

II. System Model and Problem Formulation

We give an illustration of mobile crowdsensing system in Figure 1. First, sensing data are requested by the society (Step 1). The platform divides the request into a number of location-aware tasks, and determines the PoS requirement of each task

by query frequency or data quality requirement. Then the platform publicizes the tasks (Step 2) to mobile users, and run a reverse auction (Steps 3 to 6). The auction mechanism selects users to perform tasks based on their bids (including the PoS of completing tasks and sensing cost), and determines the selected users' rewards based on the execution results. Finally, the platform publish the collected data to the society (Step 7). We now present the mathematical models for such mobile crowdsensing system.

We consider a mobile crowdsensing system that consists of a *platform*, and many mobile *users*. The platform has a set of location-aware sensing tasks $\mathcal{T} = \{1, 2, \ldots, t\}$ (*e.g.*, collecting photos of all flower shops in the city), and wants to distribute the tasks to numerous mobile users. For each task $j \in \mathcal{T}$, the platform assigns a *probability of success* (PoS) requirement T_j, *i.e.*, the task j is required to be completed with a probability greater than T_j.

The crowdsensing system also contains a set of users $\mathcal{N} = \{1, 2, \ldots, n\}$. Each user $i \in \mathcal{N}$, depending on her location and other factors, such as specific expertise and personal preference, decides a set of tasks $\mathcal{S}_i \subseteq \mathcal{T}$ that she is willing to perform. The user $i \in \mathcal{N}$ has a *probability of success* (PoS) p_i^j for each task $j \in \mathcal{S}_i$, and a total *cost* c_i to conduct all her tasks $\mathcal{S}_i$. We can interpret the PoS in different ways in various situations. For example, in opportunistic sensing, the PoS for a specific task of a user can be regarded as her probability to pass through the location of the task. We emphasize that the users make their best effort to execute tasks, but still have some probability of failing to complete the tasks. We assume that the users are able to predict such probability through their historical traces and daily journey plan. However, due to privacy preserving policy, this information is private to the user, and is not known to the platform. The cost is incurred whether a user completes the tasks or not, *e.g.*, in opportunistic sensing, smart devices continuously monitor the surrounding environment, and collect data at the background. The tuple of task set, cost, and PoS's for each user $i \in \mathcal{N}$: $\theta_i = (\mathcal{S}_i, c_i, \{p_i^j \mid j \in \mathcal{S}_i\})$, is termed as *type* in mechanism design. As a convention, we use $-i$ in subscript to denote all users except i, *e.g.*, $\boldsymbol{\theta}_{-i}$ denotes the types of all users except i, and $\boldsymbol{\theta} = (\theta_i, \boldsymbol{\theta}_{-i})$ denotes the type profile of all users.

For each user $i \in \mathcal{N}$, her *utility* u_i is defined as the reward r_i she receives, minus the cost of performing the tasks, *i.e.*, $u_i \triangleq r_i - c_i$. In this paper, we consider the strategic setting, in which each user seeks to maximize her utility, and would misreport her type, as long as doing so is profitable.

In contrast to the selfish goals of users, the objective of the platform is to maximize social welfare (minimizing the total cost of the selected users in our context), while ensuring each task satisfies its PoS requirement. We can formulate the optimization problem of the platform as follows:

$$\min_{I \subseteq \mathcal{N}} \sum_{i \in I} c_i$$

$$\text{s.t.} \quad 1 - \prod_{i \in I, \mathcal{S}_i \ni j} (1 - p_i^j) \geq T_j \quad \text{for any } 1 \leq j \leq t.$$

TABLE I
MAIN NOTATIONS

Parameter	Description
$\mathcal{T}, t$	Set of Tasks and its size
T_j	PoS requirement on task j
$\mathcal{N}, n$	Set of users and its size
$\mathcal{S}_i$	Task set of user i
p_i^j	User i's PoS for task j
c_i	User i's cost of performing tasks in her task set
θ_i	User i's type, with $\theta_i = (\mathcal{S}_i, c_i, \{p_i^j \mid j \in \mathcal{S}_i\})$
r_i	User i's reward
u_i	User i's utility, $u_i = r_i - c_i$
$\boldsymbol{\theta}$	The type profile of all users
$\boldsymbol{\theta}_{-i}$	The type profile of all users except i, i.e. $\boldsymbol{\theta}_{-i} = (\theta_1, \ldots, \theta_{i-1}, \theta_{i+1}, \ldots, \theta_n)$
q_i^j	User i's contribution for task j, $q_i^j = -\ln(1 - p_i^j)$
Q_j	Contribution requirement on task j, $Q_j = -\ln(1 - T_j)$

For the convenience of discussion, we make a transformation to the above problem. Let $q_i^j = -\ln(1 - p_i^j)$ denote a user i's *contribution* to the completion of the task j, and let $Q_j = -\ln(1 - T_j)$ correspond to the contribution requirement of the task j. With these notations, we can directly add up the contributions of users, and express the PoS constraint as

$$\sum_{i \in I, \mathcal{S}_i \ni j} q_i^j \geq Q_j \quad \text{for any } 1 \leq j \leq t.$$

A *mechanism* in game theory refers to the protocol of interaction between agents with conflict goals. Specifically, a mechanism $\mathcal{M} = (\mathcal{A}, \mathcal{R})$ consists of an allocation algorithm $\mathcal{A}$ and a payment/reward scheme $\mathcal{R}$. The allocation algorithm receives the (declared) types of users, and outputs the set of winning agents. The reward scheme determines the rewards to incentivize agents to report true types. For a good mechanism in practical crowdsensing systems, it needs to satisfy the following properties:

• **Incentive Compatibility**: A mechanism is incentive compatible if each user maximizes her utility by revealing her true type, regardless of the other users' actions. Formally, for any other users' types θ_{-i} and any declared type $\bar{\theta}_i$, we have: $u_i(\theta_i, \theta_{-i}) \geq u_i(\bar{\theta}_i, \theta_{-i})$.

• **Individual Rationality**: A mechanism is individually rational if each truthful user has a non-negative utility: $u_i(\theta_i, \theta_{-i}) \geq 0$.

• **Computational Efficiency**: A mechanism is computationally efficient if both the allocation scheme and reward scheme can be computed in polynomial time.

We define a mechanism that satisfies incentive compatibility and individual rationality as a *strategy-proof* mechanism [18].

In the following discussion, we first investigate the single task setting, where the platform has only one task to complete. We will omit the task index, and use notations Q, q_i, and c_i in this context. We then consider the multi-task, single-minded setting. The single-minded user has a strict task demand, meaning that the user is satisfied with either being assigned all the tasks from her task set, or none of tasks.

We summarize the frequently used notations in Table I.

III. Fault Tolerant Mechanism Design

In this section, we first discuss design challenges and design rationale. We then propose our mechanisms, including winner determination algorithm and reward calculation scheme, for both single task and multi-task cases. Finally, we prove that the mechanisms achieve both good approximation ratio and strategy-proofness.

A. Design Challenges

1) Failure of Existing Mechanisms: The mechanism design problem considered in this paper is distinct from the previous works [5]–[7] in that, we let PoS to be one of the user's private parameters. Our problem in general setting (both cost and PoS are private parameters) falls into the multi-dimensional mechanism design [11]. The positive results in multi-dimensional mechanism always have certain restrictions on private parameters. For example, in strategy-proof combinatorial auctions with single-minded users [12], the declared bundle would not be a subset of the true bundle, because users would have zero valuations when they are assigned such kind of bundles. However, in our context, there is no similar restriction on the declared PoS, which causes even the classical VCG mechanism to fail. Consider the following example in a single task setting: there are four users with types (cost and PoS pairs) $(3, 0.7), (2, 0.7), (1, 0.5)$, and $(4, 0.8)$, respectively. The platform wants to minimize the total cost while requires the task to be completed with a probability 0.9. If users report their types truthfully, VCG will select user 1 and user 2. However, user 3 can be selected by declaring a PoS of 0.9, and then obtain a utility greater than 0. Hence, VCG is not strategy-proofness in our setting.

The key challenge in designing strategy-proof mechanism in our setting is that the PoS can also affect the outcomes of the allocation. The payment in VCG involves only the effect of costs but not that of the PoS, resulting in the incentive to misreport the PoS information. The basic idea to tackle the problem is to consider the effect of PoS during reward scheme design. In this paper, we adopt the *execution contingent* (EC) reward scheme [17] to achieve this goal. The EC reward scheme distributes different amounts of reward to the user according to whether the user completes the task or not. For example, in single task setting, user i will receive rewards r_i^1 and r_i^2 (possibly negative) for the success and failure of task execution, respectively. Then, her expected utility, with respect to her own private parameters, is

$$u_i = p_i(r_i^1 - r_i^2) - c_i + r_i^2. \tag{1}$$

By properly setting the rewards, we can ensure incentive compatibility in terms of the user's PoS, which will be discussed in detailed design in next section.

We further emphasize that the EC reward scheme is not compatible with most allocation algorithms in mechanism design. Using standard mechanism design technique with monotone allocation algorithms and critical-price based reward schemes, a user will get selected if and only if she declares a bid larger than her critical bid. Due to the monotonicity of the allocation algorithms, the selection boundary for a user is a (piece-wise) line (or hyperplane) in her type space. Suppose that we run optimal allocation algorithm in the former example, which obviously ensures monotonicity in terms of both PoS and cost, we can plot the selection boundary for user 3 in Figure 2. The selection boundary line is not a linear line. However, the EC reward scheme requires that the relation between the PoS and cost should be linear, which can be obtained by setting $u_i = 0$ in Equation (1). Therefore, we cannot align the EC reward scheme with the monotone allocation algorithm to guarantee the incentive compatibility for both PoS and cost.

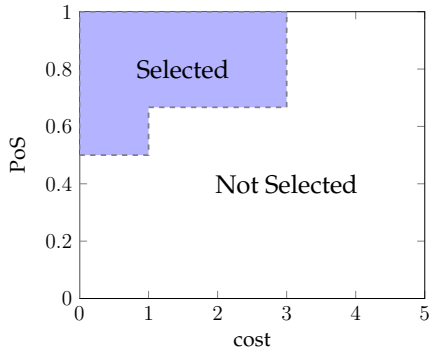


Fig. 2. Selection Boundary of User 3 (User 3 will be selected if $p_3 \geq 2/3$ and $c_3 \leq 3$, or $p_3 \geq 0.5$ and $c_3 \leq 1$).

The above discussion demonstrates the hardness of achieving the strategy-proofness for both the PoS and cost. Thus, we make the problem tractable by assuming that the costs are truthfully declared, which is reasonable in practice, and can be achieved by augmenting the incentive mechanism with a cost verification scheme. The platform can monitor the indicators related to cost, such as energy consumption and data transmission fee, to obtain the actual cost consumption during task execution. By doing that, the platform can verify the declared costs of users, and punish the users who lie about the costs. Therefore, we focus on the incentive mechanism design with economic robustness in the PoS dimension.

2) Hardness of Optimization: We show that even when the types of users are known, the problems we consider are computationally intractable. To be exact, they all fall into the category of **NP-complete** problems. The single task problem is equivalent to the minimum knapsack problem [19], which can be expressed as:

$$\min_{I \subseteq [n]} \sum_{i \in I} c_i \quad \text{s.t.} \sum_{i \in I} q_i \geq Q.$$

As for the multi-task, single-minded problem, we can show that it can be reduced from the problem of weighted set cover [20]. For each instance of weighted set cover with a universe set U and a collection of weighted subsets $\{S_1, S_2, \ldots, S_n\}$, we can construct a corresponding instance for the multi-task, single-minded problem. We generate a task set corresponding to U, and fix the contribution requirement of each task to be 1. For each subset, we can generate a user,

Algorithm 1: Dynamic Programming for Knapsack Problem

Input: A set of users $\mathcal{N}$, a profile of contribution $\boldsymbol{q}$, a profile of cost $\boldsymbol{c}$.
Output: The states of knapsack problem W.

1 Initialize the set of states: $W(0) \leftarrow \{(\varnothing, 0, 0)\}$;
2 **foreach** $j = 1$ ***to*** $|\mathcal{N}|$ **do**
3 $\quad W(j) \leftarrow W(j-1)$;
4 $\quad$ **foreach** $(I, Q, C) \in W(j-1)$ **do**
5 $\quad\quad W(j) \leftarrow W(j) \cup \{(I \cup \{j\}, Q + q_j, C + c_j)\}$;
6 $\quad$ Remove all dominated states from $W(j)$;
7 **return** $W(j)$;

whose task set is exactly this subset, contribution for each task is equal to 1, and cost is assigned as the weight. It can be seen that the optimal solution to our problem is actually the set cover with the minimum cost, which demonstrates the equivalence of these two problems.

Due to the hardness of achieving the optimal solution, we design approximation algorithms to these two problems, and trade optimality off for computational efficiency.

B. Single Task Mechanism

In this subsection, we propose a sealed-bid reverse auction for the single task setting, including a winner determination algorithm and a reward determination scheme.

Winner Determination Algorithm: The task allocation algorithm is a fully polynomial-time approximation scheme for the minimum knapsack problem, which is based on dynamic programming approach and scaling technique. Dynamic programming is a pseudopolynomial and optimal algorithm for knapsack problems. We first present the detailed procedure of the dynamic programming approach in Algorithm 1. We maintain an array entry $W(j)$ for $j = 0, 1, \cdots, n$. Each entry $W(j)$ is a list of tuples (I, Q, C) (also called as states), each of which indicates that there is a user set I from the first j users that have the exact total contribution Q and cost C. In Algorithm 1, we initialize $W(0)$ to be $\{(\varnothing, 0, 0)\}$. For each of $j = 1, 2, \cdots, n$, we first set $W(j) = W(j-1)$, and for each tuple $(I, Q, C) \in W(j-1)$, we also add the tuple $(I \cup \{j\}, Q + q_j, C + c_j)$ to the list $W(j)$ (Lines 3 to 5). We finally remove from $W(j)$ all dominated states (Line 6), where one state (I, Q, C) dominates another state (I', Q', C') if $C \leq C'$ and $Q \geq Q'$. For maximum knapsack problem, we search for the feasible tuple (*i.e.*, the total cost C is no larger than the required cost) with the maximum contribution from $W(n)$, while for minimum knapsack problem, we return the feasible tuple (*i.e.*, the total contribution Q is no less than the required contribution) with the minimum cost from $W(n)$. Algorithm 1 correctly computes the optimal value of the knapsack problem, and takes $O(n \times min(Q_s, C_s))$ time, where $Q_s = \sum q_i$ and $C_s = \sum c_i$.

We now describe the FPTAS for the problem of minimum knapsack. The key idea behind FPTAS for knapsack problem is the scaling technique: rounding down the cost (or contribution) to the nearest integer multiples of a scaling parameter. This scaling technique reduces the exponential time complexity to polynomial time. By carefully selecting the scaling parameter, we can bound the accuracy we lose in rounding. For the maximum knapsack problem, the scaling parameter is set as a fraction of the maximum contribution, which is a lower bound of the optimal value [20]. However, in the minimum knapsack problem, the maximum cost may not be a lower bound of the optimal solution, making it nontrivial to quantify the accuracy loss in this new context. In order to overcome this challenge, we divide the original minimum knapsack problem into multiple subproblems, and apply the scaling technique and dynamic programming for each subproblem. Among multiple feasible and approximation solutions in these subproblems, we output the one with the minimum cost as the final result.

Algorithm 2: Winner Determination Algorithm for Single Task Setting

Input: A set of users $\mathcal{N}$, a profile of contributions $\boldsymbol{q}$, a profile of costs $\boldsymbol{c}$, a contribution requirement Q.
Output: A set of selected user I^*.

1 $(I^*, C^*) \leftarrow (\varnothing, \infty)$;
2 Order users such that $c_1 \leq c_2 \leq \cdots \leq c_n$;
3 **for** $k = 1$ ***to*** n **do**
4 $\quad \mu_k \leftarrow \dfrac{\epsilon c_k}{k}$;
5 $\quad$ **for** $j = 1$ ***to*** k **do**
6 $\quad\quad c'_j \leftarrow \lfloor c_j / \mu_k \rfloor$;
7 $\quad W \leftarrow$ KNAPSACK$(\mathcal{N}^k, \boldsymbol{q}^k, \boldsymbol{c'}^k)$;
8 $\quad$ Find the feasible tuple $(\bar{I}, \bar{Q}, \bar{C})$ in W with the minimum cost, *i.e.*, $\bar{Q} \geq Q^*$ and $\bar{C}$ is minimum;
9 $\quad$ **if** $\bar{C} \times \mu_k \leq C^*$ **then**
10 $\quad\quad C^* \leftarrow \bar{C} \times \mu_k$; $I^* \leftarrow \bar{I}$;
11 **return** I^*;

In Algorithm 2, we first sort users in a non-decreasing order of the costs (Line 2). Then, for each of $k = 1, 2, \cdots, n$, we solve the subproblem involving the first k users. Specifically, we set the scaling parameter μ_k as $\epsilon c_k / k$, and round down each user's cost by setting $c'_j = \lfloor c_j / \mu_k \rfloor$ (Lines 4 to 6). With the scaled costs $\boldsymbol{c'}^k$ and the contribution profile $\boldsymbol{q}^k$, we run the dynamic programming scheme (Algorithm 1) to obtain the set of states W. From the set W, we select the feasible state $(\bar{I}, \bar{Q}, \bar{C})$ with the minimum cost, *i.e.*, $\bar{Q} \geq Q^*$ and $\bar{C}$ is the minimum, as the near-optimal solution for this subproblem (Lines 7 to 8). Finally, we return the feasible solution with the minimum cost over all the subproblems as the result of the winner determination algorithm (Lines 9 to 10).

Reward Calculation Scheme: Our reward calculation scheme is inspired by the ideas of critical bid-based payment scheme in standard mechanism design [12] and the execution contingent reward scheme in uncertain environment [17]. The critical bid is defined as the minimum PoS that a user should

Algorithm 3: Reward Scheme for Single Task Setting

Input: A winning user $i \in I^*$, a set of users $\mathcal{N}$, a profile of contributions $\boldsymbol{q}$, a profile of costs $\boldsymbol{c}$, a contribution requirement Q.

Output: The reward r_i for the user i.

1 Use binary search to find the minimum $\bar{q}_i$ in the range $[0, Q]$ such that $\text{ALLO\_SG}(\mathcal{N}, (\bar{q}_i, \boldsymbol{q}_{-i}), \boldsymbol{c}, Q)$ returns a set of winning users containing user i;

2 $\bar{p}_i \leftarrow 1 - e^{-\bar{q}_i}$;

3 $r_i = \begin{cases} (1-\bar{p}_i) \times \alpha + c_i, & \text{if } i \text{ completed the task,} \\ -\bar{p}_i \times \alpha + c_i, & \text{if } i \text{ did not complete the task.} \end{cases}$;

4 **return** r_i;

declare to win the auction. To guarantee the existence of the critical bid, the winner determination algorithm should satisfy the property of monotonicity, meaning that a winning user will still be a winner if she raises her PoS. We have the following lemma for the winner determination algorithm.

Lemma 1. *The winner determination algorithm (Algorithm 2) is monotone in terms of PoS.*

Proof. Since the PoS has a positive correlation with the contribution, we only have to show the monotonicity of the allocation algorithm in terms of the contribution. Suppose the solution from the subproblem k has been selected by Algorithm 2 as the final result. When a winning user i raises her declared contribution q_i, we distinguish three different changes for the outcomes of the subproblems.

- In the subproblems that previously select i as candidate winning users (including the subproblem k), the user i will still be selected by raising her declared contribution. This is because, in the scaled cost domain, the dynamic programming is an optimal solution for the knapsack problem, which is clearly monotone with respective to the contribution.
- Some subproblems that not previously select i may involve her into the candidate winning user set after user i increasing her declared contribution.
- The remaining subproblems still not select user i, even user i declares a higher contribution.

We note that the corresponding total costs in the first two cases may decrease, while the cost in the last case remains the same. We claim that the solutions from the third type of subproblems will not be selected as the final result. This is because the costs of such subproblems are definitely larger than that of the subproblem k. Therefore, when a winning user i declares a higher contribution, the winner determination algorithm will choose the final result among the subproblems that still select user i, *i.e.*, the first two types of subproblems, implying that the user i will still be a winner. □

After proving the existence of the critical bid, we now design Algorithm 3 to find the critical bid, and determine the reward for a winning user $i \in I^*$. Since the winner determination algorithm is monotone with respective to the contribution, we can use binary search over the range $[0, Q]$ to find the minimum declared contribution $\bar{q}_i$ for the user i such that the winner determination algorithm, denoted by $\text{ALLO\_SG}(\cdot)$, would picks the user i as a winner (Line 1). With the critical contribution $\bar{q}_i$, we can obtain the critical PoS $\bar{p}_i$ by setting $\bar{p}_i = 1 - e^{-\bar{q}_i}$. Depending on whether the user completes the task or not, we further distinguish the reward in the following two cases.

- If the user i completed the task, she would receive the reward $(1-\bar{p}_i) \times \alpha + c_i$;
- If the user i did not complete the task, she would get the reward $-\bar{p}_i \times \alpha + c_i$.

Here α is a reward scaling factor that can be adjusted according to the budget constraint of the platform. We now show that such reward scheme together with the monotone winner determination algorithm guarantee the properties of strategy-proofness.

Theorem 1. *The single task mechanism is strategy-proof.*

Proof. If user i wins the auction, her expected utility is

$$\begin{aligned} u_i &= p_i((1-\bar{p}_i) \times \alpha + c_i) + (1-p_i)(-\bar{p}_i \times \alpha + c_i) - c_i \\ &= (p_i - \bar{p}_i) \times \alpha. \end{aligned}$$

Otherwise, her utility is 0. To prove the theorem, we further distinguish the following two cases.

- User i wins the auction by declaring her true type. Since $\bar{p}_i$ is the user's critical bid, it must be the case $p_i \geq \bar{p}_i$. Thus, the user has a nonnegative utility when bidding truthfully. Deviating from truthful bidding will only cause her to lose the auction, and have a utility of 0, which is not better than bidding truthfully.
- User i loses the auction when declaring her true type, meaning that $p_i < \bar{p}_i$. If she misreports her type and wins the auction, her utility will be negative, which is also not better than bidding truthfully (having zero utility).

To sum up, user i has no incentive to misreport in both of the above two cases, and derives non-negative utility when bidding truthfully. Therefore, the single task mechanism satisfies the properties of incentive compatibility and individual rationality, resulting in the strategy-proofness of the mechanism. □

Theorem 2. *The winner determination algorithm is a* $(1+\epsilon)$*-approximation algorithm for the single task setting.*

Proof. Let $c(I)$ denotes the total cost of users in the set I, i.e. $c(I) = \sum_{i \in I} c_i$, and $c'(I)$ denotes the rounded costs of the users in the set I, i.e. $c'(I) = \sum_{i \in I} c'_i$. Let I^* be the user set return by Algorithm 2, and O be the optimal solution.

Let c_k be the largest cost in O. Consider the kth subproblem, all the users in this subproblem have costs less than c_k. According to the principle of the scaling method, we have

$$u_k c'_j \leq c_j \leq u_k c'_j + u_k, \quad \forall 1 \leq j \leq k. \tag{2}$$

We use $\bar{I}^k$ to denote the user set selected by our algorithm for the kth subproblems. With the above equation, we have

$$c(\bar{I}^k) \leq \sum_{i \in \bar{I}^k} (\mu_k c'_i + \mu_k) = \mu_k c'(\bar{I}^k) + \mu_k |\bar{I}^k| \tag{3}$$

Algorithm 4: Winner Determination Algorithm for Multi-Task, Single-Minded Setting

Input: A set of users $\mathcal{N}$, a profile of types $\boldsymbol{\theta}$, a set of tasks $\mathcal{T}$, a profile of contribution requirements $\boldsymbol{Q}$.

Output: A set of selected user I^*.

1 $I^* \leftarrow \varnothing$; $\bar{\boldsymbol{Q}} \leftarrow \boldsymbol{Q}$;

2 **while** $\exists j : \bar{Q}_j > 0$ **do**

3 $\quad i \leftarrow \arg\max_{i \in \mathcal{N} \setminus I} \left(\sum_{j \in \mathcal{S}_i} \min \left\{ q_i^j, \bar{Q}_j \right\} \right) / c_i$;

4 $\quad I^* \leftarrow I^* \cup \{i\}$;

5 $\quad$ **for** $j \in \mathcal{S}_i$ **do**

6 $\quad\quad \bar{Q}_j \leftarrow \max \left\{ 0, \bar{Q}_j - q_i^j \right\}$;

7 **return** I^*;

For the rounded costs, user set $\bar{I}_k$ has the minimum costs, *i.e.*, $c'(\bar{I}^k) \leq c'(O)$, due to the optimum of the dynamic programming. Together with $|\bar{I}^k| \leq k$, we further have

$$\mu_k c'(\bar{I}^k) + \mu_k |\bar{I}^k| \leq \mu_k c'(O) + \mu_k k. \tag{4}$$

According to Equation (2), $\mu_k c'(O) \leq c(O)$. Due to the setting of the scaling parameter, we get $\mu_k k = \epsilon c_k$. Furthermore, the user with cost c_k is involved in the optimal solution O, so the inequality $\epsilon c_k \leq \epsilon c(O)$ holds. With Equation (4), we get

$$\mu_k c'(O) + \mu_k k \leq c(O) + \epsilon c(O). \tag{5}$$

Combining the above Equations (3)(4)(5), we finally have

$$c(\bar{I}^k) \leq (1 + \epsilon) c(O).$$

Since our algorithm selects the solution with the minimum costs over all the subproblems, *i.e.*, $c(I^*) \leq c(\bar{I}^k)$. Thus, we have $c(I^*) \leq (1 + \epsilon) c(O)$, and complete the proof. □

Theorem 3. *The single task mechanism is computationally efficient.*

Proof. We first analyze the computational complexity of the winner determination algorithm. For the kth subproblem, the maximum cost is c_k, and the corresponding scaled cost is $c'_k = \lfloor c_k / \mu_k \rfloor$, where $\mu_k = \epsilon c_k / k$. The most time consuming step in a subproblem is to call the knapsack procedure, which has time complexity $O(k \times k \times c'_k) = O(k^3/\epsilon)$. We have a total of n subproblems, and thus the running time of the winner determination algorithm is $O(n^4/\epsilon)$.

We next analyze the running time of the reward scheme. The reward scheme calls the task allocation algorithm for at most $\log(Q)$ times. Hence, the computational complexity of the reward scheme is $O(\log(Q) n^4/\epsilon)$.

From the above discussion, we can see that both the winner determination algorithm and the reward scheme have time complexity that is polynomial in the input size and the approximation parameter, *i.e.*, n, $\log(Q)$, and $1/\epsilon$. Thus, the single task mechanism is computationally efficient. □

C. Multi-task, Single-minded Mechanism

Our mechanism for multi-task, single-minded setting is also a sealed-bid auction, including a winner determination algorithm and a reward calculation scheme.

Winner Determination Algorithm: We present the detailed procedure of the greedy algorithm for winner determination in Algorithm 4. It iteratively chooses a user that maximizes the contribution-cost ratio, which is defined as the ratio between her total contribution: $\sum_{j \in \mathcal{S}_i} \min\{q_i^j, \bar{Q}_j\}$ and the cost c_i (Lines 3 to 4). After that, we update the contribution requirements of the tasks (Lines 5 to 6). This iteration process continues until the contribution requirements of all the tasks are satisfied.

Reward Determination Scheme: Similar to the reward scheme for single task setting, our design for multi-task setting is also built upon the critical-bid based payment and the execution contingent reward scheme. The critical bid in multi-task setting is defined as the minimum *total* contribution that the user should decare to win[1]. To ensure the existence of the critical bid, we also need to show the monotonicity of the winner determination algorithm.

Lemma 2. *The task allocation algorithm for the multi-task single minded setting is monotone in terms of contribution.*

Proof. We need to show that if user i gets selected by reporting q_i^j for task j, she will also be selected if she reports $\bar{q}_i^j > q_i^j$. By reporting a higher contribution, user i will increase her contribution-cost ratio, leading to be selected in the same or earlier iteration during the allocation process. Thus, we have proved the monotonicity of the allocation algorithm. □

The new challenges arose in the multi-task single-minded setting is that users now may have multiple critical bids, because they may be selected as winners in different iterations during the winner determination process. The basic idea underlying our design is to take the minimum over all the possible critical bids, and regard it as the ultimate critical bid for the user, which guarantees the property of strategy-proofness.

Algorithm 5 determines the reward to a winning user i. It excludes the user i, and reruns the task allocation algorithm (denoted as ALLO_MP($\cdot$)). In each iteration of the allocation process, we calculate the current critical bid for the user i. Suppose user k is the winner in this iteration, then for user i to be selected, she must have a contribution-cost ratio larger than that of the user k. Thus, the critical bid for the user i in this iteration is $\frac{c_i}{c_k} \sum_{j \in \mathcal{S}_k} \min\{\bar{Q}_j, q_k^j\}$. Taking the minimum among these critical bids from different iterations, we can obtain the minimum contribution, denoted by $\bar{q}_i$, that the user i should declare to be selected in the winner determination process (Lines 2 to 6). With the critical contribution $\bar{q}_i$, we can get the critical PoS by setting $\bar{p}_i = 1 - e^{-\bar{q}_i}$. Finally, we can calculate the reward to user i according to the principles of the execution contingent reward scheme:

[1] In the multi-task setting, we defined the critical bid in terms of contribution rather than PoS for the convenience of discussion.

Algorithm 5: Reward Calculation Scheme for Multi-Task, Single-Minded Setting

Input: A winning user $i \in I^*$, a set of users $\mathcal{N}$, a profile of types $\boldsymbol{\theta}$, a set of tasks $\mathcal{T}$, a profile of contribution requirements $\boldsymbol{Q}$.

Output: The reward r_i for the user i.

$\bar{q}_i \leftarrow \infty$;

Run allocation algorithm without the participation of user i: ALLO_MP($\mathcal{N}, \boldsymbol{\theta}_{-i}, \mathcal{T}, \boldsymbol{Q}$), and maintain the following operations:

for *each iteration of the while loop in Algorithm 4* **do**

  Let $\bar{\boldsymbol{Q}}$ denote the values of the corresponding contribution requirements at the beginning of this iteration;

  Let k be the user selected in this iteration;

  $\bar{q}_i \leftarrow \min\left\{\bar{q}_i, \frac{c_i}{c_k} \sum\limits_{j \in \mathcal{S}_k} \min\left\{\bar{Q}_j, q_k^j\right\}\right\}$;

$\bar{p}_i \leftarrow 1 - e^{-\bar{q}_i}$;

$r_i =$

$$\begin{cases} (1-\bar{p}_i) \times \alpha + c_i, & \text{if i completed any one of tasks,} \\ -\bar{p}_i \times \alpha + c_i, & \text{if i completed none of tasks.} \end{cases}$$

return r_i;

• If the user i completed any one of tasks from her task set, she would receive the reward $(1-\bar{p}_i) \times \alpha + c_i$;

• If the user i completed none of tasks, she would get the reward $-\bar{p}_i \times \alpha + c_i$.

Here, α is a reward scaling parameter.

We have the following properties for the multi-task, single-minded mechanism.

Theorem 4. *The multi-task, single-minded mechanism is strategy-proof.*

Proof. In the multi-task setting, users can cheat on both task sets and contributions. We first argue that cheating on task set is equal to misreport the corresponding contributions. Suppose the true task set of user i is $\mathcal{S}_i$. If user i misreports a task set $\bar{\mathcal{S}}_i$, we can interpret this cheating behaviour as increasing the contributions from zero to some positive number for the tasks in $\bar{\mathcal{S}}_i \backslash \mathcal{S}_i$, and decreasing the contributions from positive to zero for the tasks in $\mathcal{S}_i \backslash \bar{\mathcal{S}}_i$. Thus, we only need to prove the property of strategy-proofness in terms of contributions.

According to the reward scheme, the utility of a losing user is zero, and the expected utility of a winning user i is

$$\begin{aligned} u_i &= \left(1 - \prod_{j \in \mathcal{S}_i} (1-p_i^j)\right) ((1-\bar{p}_i) \times \alpha_i + c_i) \\ &\quad + \prod_{j \in \mathcal{S}_i} \left(1-p_i^j\right) (-\bar{p}_i \times \alpha + c_i) - c_i \\ &= \left(1 - \bar{p}_i - \prod_{j \in \mathcal{S}_i} (1-p_i^j)\right) \times \alpha \\ &= \left(e^{-\bar{q}_i} - e^{-\sum_{j \in \mathcal{S}_i} q_i^j}\right) \times \alpha. \end{aligned} \tag{6}$$

We further consider the following two different scenarios.

• User i wins the auction when declaring true type. Since $\bar{q}_i$ is the critical bid, we must have $\sum_{j \in \mathcal{S}_i} q_i^j \geq \bar{q}_i$ in this case, implying that user i gets a non-negative utility according to Equation (6). When user i misreports her contributions, she may be selected as a winner in some other iteration, and stay the same utility; or she may lose the auction, and get zero utility. The resulting utilities in these two cases cannot be better than that obtained from truthfully bidding.

• User i loses the auction, and gets zero utility when bidding truthfully. In this case, we have $\sum_{j \in \mathcal{S}_i} q_i^j < \bar{q}_i$, because user i loses the auction in the first iteration, meaning that the total contribution, *i.e.*, the left hand side term, is less than the critical bid. Thus, if the user cheats on the contribution to win the auction, she will get a negative utility, which is worse than the utility obtained when bidding truthfully.

From the above discussion, we can claim that users cannot obtain extra utilities by misreporting their contributions. Furthermore, users obtain non-negative utilities when bidding truthfully. Therefore, the multi-task, single minded mechanism is incentive compatible and individual rational, and then satisfies the property of strategy-proofness. □

Before we show the near-optimal property of our mechanism, we first give a useful definition.

Definition 1 (Submodular function)**.** *Let Ω be a finite set, a submodular function is a set function $f : 2^{\Omega} \mapsto \mathbb{R}$, which satisfies*

$$f(X \cup \{x\}) - f(X) \geq f(Y \cup \{x\}) - f(Y)$$

for any $X, Y \subseteq \Omega$ with $X \subseteq Y$ and $x \in \Omega \backslash Y$. A submodular function f is normalized if $f(\varnothing) = 0$, and is monotonically increasing if for any $X \subseteq Y \subseteq \Omega$, $f(X) \leq f(Y)$.

We assume that there is a minimal unit of contribution, denoted as Δq. This can be achieved by specifying a set of PoS values and let users choose their bids from the set. Then, we define $f(I)$ as the number of units of contribution provided by users in I, *i.e.*, $f(I) \triangleq \frac{1}{\Delta q} \sum_{j=1}^{t} \min\{Q_j, \sum_{i \in I: j \in \mathcal{S}_i} q_i^j\}$. It is easy to verify that $f(I)$ is a normalized, monotonically increasing submodular function. We also define $c(I)$ as the total costs of users in I, *i.e.*, $c(I) \triangleq \sum_{i \in I} c_i$. We simplify the notations by setting $c(x) = c(\{x\}), f(x) = f(\{x\})$ for a singleton set. Let O be the optimal set of users that

minimizes the total cost, and I^* be the set of users return by our mechanism. We reorder users such that the user selected in the kth iteration of Algorithm 3 is called user k. We assume there are l iterations in total, and let I_i^* denote the first i selected users, particularly we have $I_0^* = \varnothing$ and $I_l^* = I^*$. We denote the marginal contribution of a user x given a selected user set I as $\Delta f_x(I) = f(I \cup \{x\}) - f(I)$. For the ease of notation, we denote $\Delta_x(i) = \Delta_x f(I_i)$. We also introduce the following three notations.

- $w(x) \triangleq \sum_{i=1}^{l} (\Delta_x(i-1) - \Delta_x(i)) \frac{c(i)}{\Delta_i(i-1)}, \quad \forall x \in I^*$.
- $\gamma \triangleq \max_{i \in [n]} f(i)$.
- $H(x) \triangleq 1 + \frac{1}{2} + \ldots + \frac{1}{x}$.

We first present two useful lemmas.

Lemma 3. $c(I^*) \leq \sum_{x \in O} w(x)$

Proof. For $w(x)$, we have,

$$\begin{aligned} w(x) &= \sum_{i=1}^{l} (\Delta_x(i-1) - \Delta_x(i)) \frac{c(i)}{\Delta_i(i-1)} \\ &= \frac{c(1)}{f(1)} \Delta_x(0) + \sum_{i=1}^{l-1} \left(\frac{c(i+1)}{\Delta_{i+1}(i)} - \frac{c(i)}{\Delta_i(i-1)} \right) \Delta_x(i) \end{aligned}$$

Similarly, we can express $c(I^*)$ as

$$\begin{aligned} c(I^*) =& \sum_{i=1}^{l} \frac{\Delta_i(i-1)}{\Delta_i(i-1)} c(i) \\ =& \sum_{i=1}^{l} \left(\sum_{j=i}^{l} \Delta_j(j-1) - \sum_{j=i+1}^{l} \Delta_j(j-1) \right) \frac{c(i)}{\Delta_i(i-1)} \\ =& \frac{c(1)}{f(1)} \sum_{j=1}^{l} \Delta_j(j-1) + \\ & + \sum_{i=1}^{l-1} \left(\frac{c(i+1)}{\Delta_{i+1}(i)} - \frac{c(i)}{\Delta_i(i-1)} \right) \sum_{j=i+1}^{l} \Delta_j(j-1) \end{aligned}$$

By the property of submodular function and the greedy strategy of the allocation algorithm, we have $\frac{c(i+1)}{\Delta_{i+1}(i)} \geq \frac{c(i)}{\Delta_i(i-1)}$. Therefore, it suffices to prove

$$\sum_{x \in O} \Delta_x(i) \geq \sum_{j=i+1}^{l} \Delta_j(j-1) = f(I^*) - f(I_i^*).$$

For a submodular function $f(\cdot)$, we have $\sum_{x \in Y} (f(X \cup \{x\}) - f(X)) \geq f(X \cup Y) - f(X)$, for any two sets X and Y. Applying this equation, we obtain

$$\sum_{x \in O} \Delta_x(i) \geq f(O \cup I_i^*) - f(I_i^*) \geq f(O) - f(I_i^*) = f(I^*) - f(I_i^*).$$

The last equation holds because users from O and I^* both "cover" the contribution requirements of all the tasks, *i.e.*, $f(O) = f(I^*) = \frac{Q}{\Delta q}$. Therefore, we have proved $c(I^*) \leq \sum_{x \in O} w(x)$. □

Lemma 4. *$w(x) \leq c(x) H(\gamma)$ for each $x \in O$.*

Proof. By the greedy strategy of the allocation algorithm, given the user $x \in O$, we have

$$\frac{c(i)}{\Delta_i(i-1)} \leq \frac{c(x)}{\Delta_x(i-1)} \quad \text{if } \Delta_x(i-1) > 0.$$

Let k be the index of the last iteration that $\Delta_x f(I_k^*) > 0$ and $\Delta_x f(I_{k+1}^*) = 0$. Then, we have

$$\begin{aligned} w(x) &= \sum_{i=1}^{k+1} (\Delta_x(i-1) - \Delta_x(i)) \frac{c(x)}{\Delta_x(i-1)} \\ &= c(x) \sum_{i=1}^{k+1} \frac{\Delta_x(i-1) - \Delta_x(i)}{\Delta_x(i-1)} \\ &\leq c(x) \sum_{i=1}^{k+1} \sum_{j=1}^{\Delta_x(i-1) - \Delta_x(i)} \frac{1}{\Delta_x(i-1) - j + 1} \\ &= c(x) H(f(x)) \\ &\leq c(x) H(\gamma). \end{aligned}$$

Thus, we have completed the proof for $w(x) \leq c(x) H(\gamma)$. □

Using the above two lemmas, we now present the approximation ratio of our mechanism.

Theorem 5. *The winner determination algorithm of multi-task, single-minded mechanism achieves $H(\gamma)$-approximation, where $\gamma = \max_{i \in \mathcal{N}} \frac{1}{\Delta q} \sum_{j \in \mathcal{S}_i} \min\{Q_j, q_i^j\}$.*

Proof. We have

$$c(I^*) \leq \sum_{x \in O} w(x) \leq H(\gamma) \sum_{x \in O} c(y) = H(\gamma) c(O).$$

The first inequation comes from Lemma 3 and the second inequation comes from Lemma 4. □

Theorem 6. *The multi-task, single-minded mechanism is computationally efficient.*

Proof. In the winner determination algorithm, the main loop has at most n iterations since in each iteration a user will be selected. In each iteration, we need to compute the contribution-cost ratios for at most n users, and each user has at most t task. Therefore, the computational complexity of Algorithm 4 is $O(n^2 t)$. Algorithm 5 calculates the reward to at most n users, and for each user it calls Algorithm 4 one time. Thus, the computational complexity of the reward scheme is $O(n^3 t)$. Since both the winner determination algorithm and reward calculation scheme have polynomial time complexity, our mechanism is computationally efficient. □

IV. Evaluation Results

We perform extensive simulations to evaluate our proposed mechanisms based on a real data set of location traces of Shanghai taxis. We report the evaluation results in this section.

TABLE II
DEFAULT SIMULATION PARAMETERS

Description	Values
PoS requirement T	0.8
Reward scaling factor α	10
Tasks of each user	$[10, 20]$
Mean of costs	15
Variance of costs	5

A. Simulation Settings

Each entry of the data set records the *taxi ID*, *time stamp* and *location (longitude and latitude)* of picking up and dropping passengers. We choose the data set in January, 2013 and select 1692 taxis as the population of mobile users from which we will sample. We divide the map of Shanghai into $2km \times 2km$ grids, with each grid representing a location. We assume that a taxi can perform some sensing tasks at the location where it picks up or drops passengers.

To generate the probability that a taxi arrives at a location, we use a Markov Chain to model a taxi's mobility pattern, and learn the transition matrix of the model, *i.e.*, the probability that the taxi transit from one location to another, using the data set. With such mobility model, we can generate the task set for each taxi: we randomly assign each taxi a starting location, and let the locations it will reach with a high probability in the next time slot to be its task set. The size of the task set for each user is sampled from a uniform distribution in the range $[10, 20]$. We sample the cost for each of users according to a normal distribution with mean 15 and variance 5. The PoS requirement of each task is set as a fixed value 0.8. Our default simulation parameters are summarized in Table II.

In the following, we first evaluate the accuracy of our mobility model in predicting users' future locations. Then, we evaluate the performance of our mechanisms with the metrics: *social cost*, *user's utility*, and *achieved PoS's of tasks*. We finally examine the effect of different PoS requirements of tasks on the number of selected users and the corresponding social cost.

B. Evaluation of Mobility Model

Each user has her own mobility pattern, which may affect her probability of finishing a sensing task. In order to predict the probability that a user appears at a certain location, we model her mobility pattern as a Markov process. Specifically, for a given user, suppose there are l locations she often visits. We define $\mathbf{P} \in \mathbb{R}^{l \times l}$ as her transition matrix, where P_{ij} denotes her probability of traveling from location j to location i. Given the taxi data set, we learn the transition matrix by maximum likelihood estimation. Due to the sparsity of data, *i.e.*, not every location pair appears in the data set, we adopt Laplace smoothing technique in the estimation. Thus, the estimation of P_{ij} is given as

$$P_{ij} = \frac{x_{ij}}{x_i + l},$$

where x_{ij} denotes the times the user travels to location j from location i, and $x_i = \sum_{k=1}^{l} x_{ik}$ denotes the total times the user reaches elsewhere from the location i.

To evaluate the prediction accuracy of the mobility model, we take a snapshot of the taxi trace, and predict the 3 to 15 locations that each taxi will most likely arrive at in the next time slot. We calculate the percentage of correct prediction (the taxi's actual destination is in the set of predicted locations), and show the results in Figure 3. We can observe that by setting the number of predicated locations to be 9, the correct prediction percentage is around 0.9. This demonstrates the efficiency of our mobility model in describing the mobility pattern in the taxi data set.

We also plot the empirical probability distribution function (PDF) of the predicted PoS of users in Figure 4. Due to the scarcity of the location transition, most of the PoS's are very low, falling in the range $[0, 0.2]$, which suggests that we need to recruit sufficiently enough number of users for each task to guarantee fault tolerance.

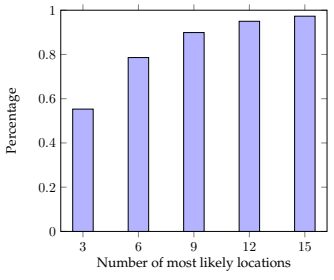


Fig. 3. Location Prediction Accuracy.

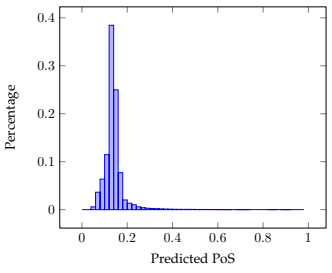


Fig. 4. PDF of Predicted PoS.

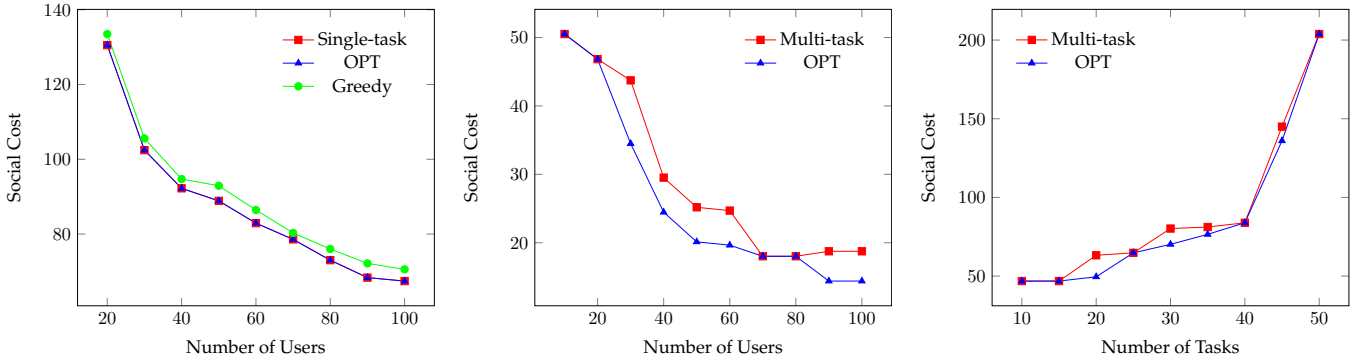
TABLE III
SIMULATION PARAMETERS FOR MULTI-TASK SETTING

Setting	# of Users	# of Tasks	Mean of Cost	PoS requirement
1	$[10, 100]$	15	15	0.8
2	30	$[10, 50]$	15	0.8

C. Evaluation of Social Cost

We first examine the social cost obtained from single task mechanism. We fix a randomly chosen task, and conduct simulations on different number of users in the range $[20, 100]$ with an increase of 10. To give a comparison, we choose two baseline algorithms, namely optimal algorithm *OPT* and greedy algorithm *Greedy*. The *OPT* algorithm computes the optimal social cost through exhaustive search, and the *Greedy* algorithm is a 2-approximation algorithm for the problem of minimization knapsack (referred to as *Min-Greedy* in [21]). We plot our results in Figure 5(a).

From Figure 5(a), we can see that as the number of users increases, the social cost first decreases sharply, and then changes steadily. This is because the costs of users are from the same distribution, and thus the new added users may not bring about a better social cost in an average sense. We also notice that under our setting, even when we choose a relatively large



(a) Social Cost of Single Task Mechanism. (b) Social Cost with Different Numbers of Users. (c) Social Cost with Various Numbers of Tasks.

Fig. 5. Social costs of single task and multi-task mechanisms in different settings with various numbers of users and tasks.

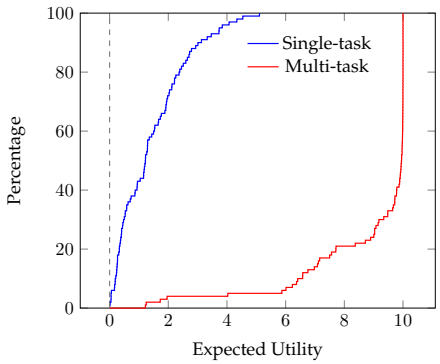


Fig. 6. Empirical CDF of users' utilities

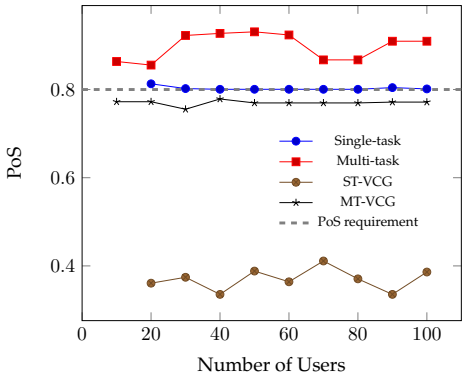


Fig. 7. Average PoS of Tasks.

approximation factor (*e.g.*, $\epsilon = 0.5$), our mechanism works as good as the OPT, and strictly better than the *Greedy* algorithm.

Next, we evaluate the social cost in multi-task setting with various number of users and tasks. We compare our mechanism with the optimal algorithm. We present the simulation parameters in Table III, and show the results in Figure 5(b) and Figure 5(c).

From Figure 5(b), we can see that the social cost decreases as the number of users increases, and when the number of users is sufficiently large, the decrease of social cost becomes stable. This is because in a more competitive market, the platform can recruit the users with higher contribution-cost ratios, and reduce the social cost needed to satisfy the contribution requirements of tasks. In contrast, the social cost increases with more tasks to be completed, since we need to recruit more users. We also note that although the approximation ratio of our mechanism can be large in theoretical analysis, the social costs given by our mechanism in practice are relatively close to that of the optimal algorithm in different scenarios.

D. Evaluation of User's Utility

We plot the empirical CDF of the expected utility of the selected users in Figure 6, to show the individual rationality of our mechanisms. The reward scaling factor α is set to 10. As suggested by the figure, all the selected users have non-negative expected utilities. Besides, the utilities of the users in multi-task setting are mostly higher than those of users in the single task setting. The reason is that the users in the multi-task setting have a higher probability (the probability that users complete any one of the tasks) to receive positive utilities than the users in the single-task setting.

E. Evaluation of Task's PoS

In Figure 7, we compare the achieved PoS's of tasks with the required PoS's. For multi-task setting, we calculate the average PoS of the tasks. In both single task and multi task settings, our mechanisms satisfy the PoS requirements of the tasks. In the single task setting, the achieved PoS's are very close to the required one, while in the multi-task setting, the obtained PoS's are higher than those in the single task setting. This is because the selected users in multi-task setting may still contribute PoS to the tasks that have already reached the required PoS, which is the side benefit of the multi-task setting.

For comparison, we also implement VCG-like mechanisms for single task and multi-task settings, namely *ST-VCG* and *MT-VCG*. As we have pointed out, if we directly apply the VCG mechanism, the user would always declare a PoS equal to 1, leading the mechanism to be untruthful. Thus, the VCG-like mechanisms simply chooses the users with the lowest costs to satisfy the requirements of all the tasks, with the input

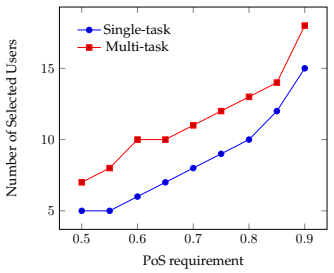


Fig. 8. PoS with Number of Users.

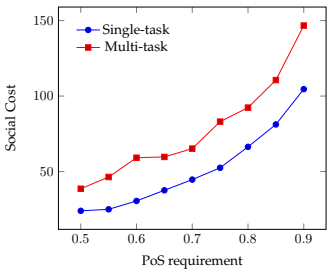


Fig. 9. PoS with Social Cost.

of untruthful PoS's. The ST-VCG mechanism would always select only one user with the lowest cost. As shown in Figure 7, the actual PoS's achieved by VCG mechanisms are lower than the required ones, especially in the single task setting, which demonstrates the infeasibility of VCG mechanism in the scenario of task execution uncertainty.

F. Effect of PoS Requirement

We further investigate the effect of different PoS requirements on the number of selected users and the resulting social cost. We fix the number of users as 100 in this evaluation. For the multi-task setting, we set the number of tasks as 50. The PoS requirement for the task is in the range $[0.5, 0.9]$ with an increase of 0.05.

Figure 8 presents the effect of PoS requirement on the number of selected users. Generally, the number of users required to complete tasks grows with the increase of PoS requirement. However, since the PoS's of users are relatively low in our prediction, the number of selected users increases fast when PoS requirements of tasks are high. The effect of PoS requirement on social cost is shown in Figure 9. Since the costs of users follow the same distribution, the effect on social cost coincides with that on the number of selected users.

V. Related Works

In this section, we briefly review the related works.

Incentive Mechanism for Mobile Crowdsensing. Incentive mechanism design have been extensively studied, mostly from a game-theoretic perspective. Paper [6] is the seminal work of designing incentive mechanisms for crowdsensing. The following works incorporated various issues with incentive mechanism design, such as task allocation in time and location dimensions [22], [23], data quality management [7], [24], [25], privacy preservation [10], budget feasibility [5]. Different optimization objectives have also been proposed, such as social welfare maximization, revenue maximization. However, these works all hold the implicit assumption that the users will definitely fulfill the tasks assigned to them, which is the main difference between our work and the previous.

Fault Tolerant Mechanism Design. Fault tolerant mechanisms are those that address the issue of execution uncertainty, such as whether, how long or how well will the assigned task be completed, guaranteeing the PoS, execution duration, quality, etc. It was initially investigated in [17], and the authors presented a novel payment scheme to achieve strategy-proofness with respective to the user's PoS. Stein *et al.* in [26] designed mechanisms under the uncertainty of execution duration. Papakonstantinou *et al.* from [27] tackled the issue of uncertain precision of data submitted by users. Similar to [17], we aim to achieve high PoS of tasks through fault-tolerance mechanism design. However, their mechanisms do not assign tasks redundantly, which we believe can potentially increase the PoS's of tasks. Furthermore, similar to the classical VCG mechanism, their mechanisms will fail due to the hight computation complexity to obtain optimal solution.

Uncertainty/Mobility in Mobile Crowdsensing. Mobility in crowdsensing has also received much attention in recent years. Ma *et al.* from [28] stated that mobility offers opportunities for data collection and transmission in crowdsensing, and investigates the features of mobility. He *et al.* in [29] designed strategy to recruit participants so that the system can keep sensing for a period of time at each location of interest. In [30], Zhang *et al.* aimed to select participants to satisfy probabilistic coverage constraint while minimizing incentive payments, In [31], the authors proposed to optimally select mobile users to form a path for collecting data from a set of fixed locations. In this paper we use mobility as one of the possible causes of uncertainty in crowdsensing. However, we emphasize that the causes are not limited to mobility, and our mechanisms can handle general execution uncertainty.

VI. Conclusion and Future Works

In this paper, we have studied the problem of incentive mechanism design for mobile crowdsensing with task execution uncertainty. We have considered a practical scenario of crowdsensing, where a user may fail to complete the assigned task, possibly due to her mobility pattern, unreliable network connection. We have examined the tools for tackling this problem, and considered a single task setting and a multi-task setting. For both setting, we have designed a mechanism with guaranteed approximation ratio and good economic properties. We have given theoretical analysis as well as simulation results to demonstrate the properties of our mechanisms.

In our future work, we will extend our mechanisms to adopt to more general and practical settings. Firstly, we will relax the assumption that we can verify the mobile user's cost, and consider the strategy-proofness with respect to both the PoS and the cost. Secondly, we will investigate more factors that cause the failure to complete the task, and incorporate them into the incentive mechanism for crowdsensing.

References

[1] N. Maisonneuve, M. Stevens, M. E. Niessen, and L. Steels, "Noisetube: Measuring and mapping noise pollution with mobile phones," in *Information technologies in environmental engineering*, 2009, pp. 215–228.

[2] M. Mun, S. Reddy, K. Shilton, N. Yau, J. Burke, D. Estrin, M. Hansen, E. Howard, R. West, and P. Boda, "Peir, the personal environmental impact report, as a platform for participatory sensing systems research," in *MobiSys*, 2009.

[3] C. Chen, D. Zhang, N. Li, and Z.-H. Zhou, "B-planner: Planning bidirectional night bus routes using large-scale taxi gps traces," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 4, pp. 1451–1465, 2014.

[4] Y. Chon, N. D. Lane, F. Li, H. Cha, and F. Zhao, "Automatically characterizing places with opportunistic crowdsensing using smartphones," in *UbiComp*, 2012.

[5] Z. Zheng, F. Wu, X. Gao, H. Zhu, G. Chen, and S. Tang, "A budget feasible incentive mechanism for weighted coverage maximization in mobile crowdsensing," *IEEE Transactions on Mobile Computing*, 2016, DOI: 10.1109/TMC.2016.2632721.

[6] D. Yang, G. Xue, X. Fang, and J. Tang, "Crowdsourcing to smartphones: incentive mechanism design for mobile phone sensing," in *Mobicom*, 2012.

[7] H. Jin, L. Su, D. Chen, K. Nahrstedt, and J. Xu, "Quality of information aware incentive mechanisms for mobile crowd sensing systems," in *MobiHoc*, 2015.

[8] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: current state and future challenges." *IEEE Communications Magazine*, vol. 49, no. 11, pp. 32–39, 2011.

[9] J. Hamm, A. C. Champion, G. Chen, M. Belkin, and D. Xuan, "Crowd-ml: A privacy-preserving learning framework for a crowd of smart devices," in *ICDCS*, 2015.

[10] W. Wang, L. Ying, and J. Zhang, "The value of privacy: Strategic data subjects, incentive mechanisms and fundamental limits," in *SIGMETRICS*, 2016.

[11] S. Chawla, J. D. Hartline, D. L. Malec, and B. Sivan, "Multi-parameter mechanism design and sequential posted pricing," in *STOC*, 2010.

[12] D. Lehmann, L. I. Oćallaghan, and Y. Shoham, "Truth revelation in approximately efficient combinatorial auctions," *Journal of the ACM*, vol. 49, no. 5, pp. 577–602, 2002.

[13] W. Vickrey, "Counterspeculation, auctions, and competitive sealed tenders," *The Journal of Finance*, vol. 16, no. 1, pp. 8–37, 1961.

[14] E. H. Clarke, "Multipart pricing of public goods," *Public Choice*, vol. 11, no. 1, pp. 17–33, 1971.

[15] T. Groves, "Incentives in teams," *Econometrica: Journal of the Econometric Society*, vol. 41, no. 4, pp. 617–631, 1973.

[16] P. Briest, P. Krysta, and B. Vöcking, "Approximation techniques for utilitarian mechanism design," in *STOC*, 2005.

[17] R. Porter, A. Ronen, Y. Shoham, and M. Tennenholtz, "Fault tolerant mechanism design," *Artificial Intelligence*, vol. 172, no. 15, pp. 1783–1799, 2008.

[18] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, *Algorithmic game theory*. Cambridge University Press Cambridge, 2007.

[19] H. Kellerer, U. Pferschy, and D. Pisinger, *Introduction to NP-Completeness of knapsack problems*. Springer, 2004.

[20] D.-Z. Du, K.-I. Ko, and X. Hu, *Design and analysis of approximation algorithms*. Springer Science & Business Media, 2011.

[21] M. M. GüNtzer and D. Jungnickel, "Approximate minimization algorithms for the 0/1 knapsack and subset-sum problem," *Operations Research Letters*, vol. 26, no. 2, pp. 55–66, 2000.

[22] M. H. Cheung, R. Southwell, F. Hou, and J. Huang, "Distributed time-sensitive task selection in mobile crowdsensing," in *MobiHoc*, 2015.

[23] L. Gao, F. Hou, and J. Huang, "Providing long-term participation incentive in participatory sensing," in *INFOCOM*, 2015.

[24] D. Peng, F. Wu, and G. Chen, "Pay as how well you do: A quality based incentive mechanism for crowdsensing," in *MobiHoc*, 2015.

[25] J. Wang, J. Tang, D. Yang, E. Wang, and G. Xue, "Quality-aware and fine-grained incentive mechanisms for mobile crowdsensing," in *ICDCS*, 2016.

[26] S. Stein, E. H. Gerding, A. Rogers, K. Larson, and N. R. Jennings, "Algorithms and mechanisms for procuring services with uncertain durations using redundancy," *Artificial Intelligence*, vol. 175, no. 14, pp. 2021–2060, 2011.

[27] A. Papakonstantinou, A. Rogers, E. H. Gerding, and N. R. Jennings, "Mechanism design for eliciting probabilistic estimates from multiple suppliers with unknown costs and limited precision," in *Agent-Mediated Electronic Commerce. Designing Trading Strategies and Mechanisms for Electronic Markets*. Springer, 2010, pp. 102–116.

[28] H. Ma, D. Zhao, and P. Yuan, "Opportunities in mobile crowd sensing," *IEEE Communications Magazine*, vol. 52, no. 8, pp. 29–35, 2014.

[29] Z. He, J. Cao, and X. Liu, "High quality participant recruitment in vehicle-based crowdsourcing using predictable mobility," in *INFOCOM*, 2015.

[30] D. Zhang, H. Xiong, L. Wang, and G. Chen, "Crowdrecruiter: selecting participants for piggyback crowdsensing under probabilistic coverage constraint," in *UbiComp*, 2014.

[31] M. Karaliopoulos, O. Telelis, and I. Koutsopoulos, "User recruitment for mobile crowdsensing over opportunistic networks," in *INFOCOM*, 2015.