A Utility-Optimal Reverse Posted Pricing Mechanism for Online Mobile Crowdsensing Task Allocation

Jixian Zhang, Xuelin Yang, Peng Chen, Zhemin Wang, Weidong Li, Zhenli He, *Senior Member, IEEE*, and Keqin Li, *Fellow, IEEE*

***Abstract*—In contrast to traditional mechanism design, the posted pricing mechanism can quickly determine the winning user and ensure the revenue of the seller through a predetermined price. Additionally, the posted pricing mechanism inherently possesses economic properties such as truthfulness and individual rationality. These properties make it an ideal method for solving online task allocation problems for mobile crowdsensing services (MCSs). The challenge in posted pricing mechanism design is being able to find reasonable posted prices under complex MCS task constraints. This article presents an innovative posted pricing mechanism to solve a general point of interest (POI)-based online MCS task allocation problem. We transform the problem into an integer programming model with the goal of maximizing the total utility of the system while satisfying various constraints. We prove that under any user arrival order, there must exist a posted price structure that can ensure that the total utility of the system is approximately optimal, with an approximation ratio of $1/(d+1)$ in the worst case. With the support of theoretical analysis, the posted price calculation can be completed using only a simple gradient descent algorithm. Compared with existing methods, our solution achieves very good results in terms of total utility and the task completion ratio, indicating that it can effectively improve the efficiency and service quality of MCSs.**

***Index Terms*—Mechanism design, mobile crowdsensing, online allocation, posted pricing.**

I. Introduction

Mobile crowdsensing services (MCSs) use many ubiquitous sensing devices through conscious [1] or unconscious [2] collaboration involving user participation to overcome the high costs of relying solely on actively deployed sensors. Mobile crowdsensing realizes the large-scale and low-cost potential of the Internet of Things, greatly expanding the depth and breadth of possible applications. At present, MCSs play an important role in the Internet of Vehicles (IoV) [3], edge-assisted crowdsensing [4], and digital twins [5]. Most early MCSs adopted a voluntary approach, and users expressed low enthusiasm for participation. Therefore, many subsequent studies have incorporated incentive mechanisms into MCSs to encourage more users to join by offering rewards or compensation, with the ultimate goal of improving the data collection quality or service quality of MCSs. Among the many mobile crowdsensing application models, point of interest (POI)-based models [6] have received considerable research attention because they can accurately reflect task collection requirement information.

Incentive mechanisms are often used to solve resource allocation or task scheduling problems [7] in cloud computing or edge computing scenarios [8]. When a traditional incentive mechanism is used in mobile crowdsensing services, it usually takes the form of a reverse auction mechanism [9]. The MCS provider collects user data, makes decisions in real time, and finally calculates the task allocation and payment solutions for the winning users. This method may consume considerable time because the user task allocation problem is NP-hard and cannot be solved in polynomial time via an optimal mechanism such as that of VCG theory [10] or AMA theory [11], and some approximation mechanisms also consume a great deal of exponential time [12]. Although some mechanisms [6] designed using monotonic allocation and critical price theory can ensure operational efficiency, they may reduce the utility obtained by the system or MCS provider.

The posted price mechanism, as a dynamic pricing method, attracted the attention of researchers very early on [13] and has a wide range of applications [14]. Unlike the above mechanisms, the posted price mechanism can collect user information in advance to set appropriate prices for items, ensuring that sellers can maximize their revenue when users arrive randomly. Moreover, the posted price mechanism executes quickly, can satisfy the requirements of real-time environments, and naturally provides economic characteristics such as truthfulness and individual rationality. In this paper, we integrate the posted pricing mechanism into the POI-based MCS context and explore a new way to solve online MCS task allocation problems. Fig. 1 shows the POI-based MCS under a posted pricing mechanism.

1) The MCS provider collects and evaluates the user data collection cost and data value information.

Received 25 February 2025; revised 20 May 2025; accepted 30 June 2025. Date of publication 24 July 2025; date of current version 9 October 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 62062065, Grant 62362068, and Grant 12071417, in part by the Project of the Yunnan Fundamental Research Projects under Grant 202501AS070076 and Grant 202401AS070416, in part by the Open Project Program of Yunnan Key Laboratory of Intelligent Systems and Computing under Grant ISC24Z01, in part by the Program for Excellent Young Talents, Yunnan, China, and in part bt the Open Foundation of Yunnan Key Laboratory of Software Engineering under Grant 2023SE208. *(Corresponding author: Zhenli He.)*

Jixian Zhang is with the School of Information Science and Engineering, Yunnan Key Laboratory of Intelligent Systems and Computing, Yunnan University, Kunming 650504, China (e-mail: zhangjixian@ynu.edu.cn).

Xuelin Yang, Peng Chen, and Zhemin Wang are with the School of Information Science and Engineering, Yunnan University, Kunming 650504, China (e-mail: yangxuelin@stu.ynu.edu.cn; pengchen@stu.ynu.edu.cn; wzm222@mail.ynu.edu.cn).

Weidong Li is with the School of Mathematics and Statistics, Yunnan University, Kunming 650504, China (e-mail: weidongmath@126.com).

Zhenli He is with the Yunnan Key Laboratory of Software Engineering, School of Software, Yunnan University, Kunming 650504, China (e-mail: hezl@ynu.edu.cn).

Keqin Li is with the Department of Computer Science, State University of New York, New Paltz, NY 12561 USA (e-mail: lik@newpaltz.edu).

Digital Object Identifier 10.1109/TSC.2025.3592426

Fig. 1. POI-based MCS under a posted pricing mechanism.

2) Based on the user data collection cost and data value, the MCS provider calculates the posted price for each POI.
3) The MCS provider notifies the users of the posted price.
4) Each user randomly enters the area where data collection is required and determines which POIs the data collection task needs to run.
5) Users perform data collection tasks at their own selected POIs and then transmit the data to the MCS provider.
6) The MCS provider confirms the task and pays the users.

In this scenario, users can enter the POI area at any time and either select tasks to execute or leave based on the current task execution situation and posted price. This method has high operability, high operating efficiency, and natural guarantees regarding the truthfulness and individual rationality of the designed mechanism. Different from the existing studies, in this paper, we assume that the cost of collecting user data can be determined in advance, which greatly reduces the complexity of the mechanism design process.

A. Motivation and Challenges

Although the posted price mechanism has many advantages, unfortunately, existing research cannot be directly applied to MCSs. The application of a posted pricing mechanism in an MCS faces three challenges. First, the greatest challenge is due to the characteristics of MCSs. The traditional posted pricing mechanism is used mostly for one seller and multiple buyers, and the pricing power is in the hands of the seller; in contrast, MCSs have one buyer (MCS provider) and multiple sellers (users), and the pricing power is in the hands of the buyer. This is a completely new problem for the design of posted pricing mechanisms, which we call reverse posted pricing mechanism design. The second challenge is how to determine the reward price (posted price) and how to evaluate whether this price is reasonable. Unlike the previously mentioned posted pricing mechanism that seeks to maximize revenue, in an MCS, the service provider is the only buyer that hopes to obtain greater data value with as little payment as possible. In short, it hopes to maximize the difference between the data value and payment. On the other hand, users incur costs when collecting data, so they hope to obtain more compensation. In this scenario, the MCS provider and the users aim to maximize their own utility. Thus, the reward price is the leverage between them. If the reward is too high, the utility gained by the MCS provider will decrease. If the reward is too low, no users will be willing to collect data. Therefore, it is necessary to design a reasonable algorithm for calculating reward prices that can maximize utility while achieving an approximation ratio that can be theoretically proven. The third challenge arises from the features of the MCS data collection task. Compared with general trading items, the data collection tasks of MCSs are subject to more constraints. For example, to ensure the service quality of MCSs, each user can collect data only once at the same POI. To reflect the importance of different POIs, the number of times data need to be collected at a POI differs among different POIs. Because the data value meets the diminishing marginal utility effect, the MCS data collection problem needs to be transformed into a submodular [15] or ordered submodular [16] problem. All of these features further complicate the problem. The above two challenges greatly increase the difficulty of designing a posted pricing mechanism for an MCS.

B. Main Contributions

This paper applies the concept of posted pricing to POI-based MCSs and determines the reward price for the data collection task for each POI based on the data collection cost and data value information of users. A reasonable reward price can ensure that the total system and MCS provider obtain the greatest possible utility while encouraging users to actively participate. To our knowledge, this study is the first attempt to apply a posted pricing mechanism to MCS data collection task allocation. In this paper, we do the following.

1) We propose a reverse posted pricing mechanism to solve a POI-based online MCS data collection and task allocation problem. We transform the problem into an integer programming model with the goal of maximizing the total utility of the system while satisfying various data collection constraints typical of actual situations.
2) We prove that in the case where users may arrive in any order (worst case), there must exist a posted price structure that can ensure that the total utility of the system is approximately optimal with an approximation ratio of $1/(d+1)$ in the worst case, where d is the maximum number of users who can perform data collection tasks. Then, we design a gradient descent algorithm to determine the optimal reward price to verify our theoretical analysis.
3) We compare our algorithm with the FixedPrice algorithm, the optimal algorithm, and an existing advanced online MCS algorithm (OMZ) [6]. The results show that our algorithm has advantages in terms of total utility and the task completion ratio, enabling it to effectively improve the efficiency and service quality of MCSs.

The remainder of this paper is structured as follows: In Section II, we analyze the existing research results obtained with respect to mobile crowdsensing and posted pricing mechanism designs. In Section III, we analyze the offline optimal utility model and the online utility model for an MCS operating under reward price constraints. In Section IV, we provide an approximation ratio proof for the online approximate optimization of utility under optimal reward price constraints. In Section V, we present an algorithm for determining the optimal reward price. In Section VI, we evaluate the performance of various algorithms through extensive experimental demonstrations. Finally, in Section VII, we conclude the paper and provide an overview of possible future research directions.

II. Related Work

Numerous scholars have conducted research in the field of MCSs, focusing mainly on incentive mechanism design, user recruitment strategies, and deep learning. Early MCS incentive mechanisms were primarily designed for offline scenarios. Yang et al. [17] proposed a user-centric model, utilizing the submodularity of utility functions to design an incentive mechanism based on reverse auctions. In this mechanism, users report executable tasks and bids, and the service provider collects information to determine task allocation and payment schemes. Cheng et al. [18] introduced a reverse auction incentive mechanism that considers the average age of information (AoI) under budget constraints. Zhao et al. [6] considered that users arrive in a random order and may leave at any time. Consequently, the service provider must make irrevocable decisions based solely on the information of users arriving before the current moment, without knowledge of future information. To address this, they designed two multi-stage online mechanisms, OMZ and OMG, which calculate thresholds based on past information to determine user allocation and payments. Hu et al. [19] employed a two-stage Stackelberg game to analyze the reciprocal relationship between service providers and mobile users, using backward induction to optimize incentive gains. Regarding user recruitment strategies, Wang et al. [20] estimated user contact probabilities based on a semi-Markov process model and proposed a predictive user recruitment algorithm for mobile crowdsensing, aiming to minimize data upload costs. With the increasing application of deep learning in various fields, researchers have begun to use neural networks to solve MCS problems. MCSs involve many issues worthy of investigation, and this paper focuses on pricing challenges. Unlike the reverse auction mechanism, in the scenario in this study, users can independently choose POIs rather than leaving the final task allocation to the service provider. The advantage of this design is that the service provider does not need to spend extra effort monitoring each user's behavior.

Posted pricing mechanisms are commonly used in the economics and computation literature due to their simplicity. The approximation guarantees of posted pricing mechanisms can be derived from prophet inequalities [21]. In 1978, Krengel et al. [22] showed that a player who selects a stopping time based on current and past observations can achieve at least half the return of a prophet. Subsequent research extended this result by allowing both the player and the prophet to select at most d elements. Alaei et al. [23] presented a factor-d optimal stopping prophet inequality with a $(1-1\ \sqrt{d+3})^{-1}$-approximation. Later, Kleinberg et al. [24] proved a prophet inequality for matroids, providing a $1\ (4d-2)$-approximation under the intersection of d matroid constraints, which was improved to $1\ (e(d+1))$ by Feldman et al. [25]. The prophet inequality problem for the intersection of d partition matroids can be generalized to the d-dimensional hypergraph prophet inequality. When the buyer is single-minded, the MPH-d combinatorial auction problem can be transformed into a d-dimensional hypergraph prophet inequality. Dütting et al. [26] generalized the previous results to settings where players make choices regarding multiple elements of a matroid and have submodular preferences over subsets of elements. They also obtained a polynomial-time $1\ (4d-2)$-approximate prophet inequality for MPH-d combinatorial auctions. Subsequently, Correa et al. [27] improved the bound to $1\ (d+1)$, which is tight. The posted pricing problem for MCSs is the inverse optimization problem of the posted pricing problem in combinatorial auctions and involves a redefinition of the utility function. As a result, designing the corresponding posted pricing mechanism becomes a significant challenge.

To date, there are still many aspects of the MCS problem based on posted pricing mechanisms that are worth further exploration. Singla et al. [28] and Balkanski et al. [29] explored how to determine posted pricing mechanisms in crowdsourcing problems, with the goal of maximizing the total valuation of the organizer under budget constraints. [28] assumed that each worker's task has a unit value, while [29]'s valuation function is submodular. Han et al. [30] studied a Bayesian-based quality-aware pricing problem, aiming to minimize payment prices under budget constraints. Unlike these studies, our proposed posted pricing mechanism starts with the goal of maximizing total utility and adjusts the posted prices to balance the utility between the service provider and the users, ensuring that both parties achieve satisfactory outcomes. Table II compares our approach with the main closely related methods from five perspectives.

III. Parameters and Problem Formulations

We assume that the MCS provider needs to collect data from m POIs within a given period. These POIs are represented by a set $\mathcal{M} = \{1, 2, \ldots, m\}$. There are n users, represented by $\mathcal{N} = \{1, 2, \ldots, n\}$. Within the given period, users can arrive in any order to perform data collection tasks. We use a vector $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{im})^{\mathrm{T}} \in \{0,1\}^m$ to represent user i's $(i = 1, 2, \ldots, n)$ data collection result, where $x_{ij} = 1$ means that user i should collect data at POI j; otherwise, $x_{ij} = 0$. For convenience, we let $\mathrm{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)$ represent the data collected by all users.

For each user, the cost of data collection is $c_i(\boldsymbol{x}_i)$, which is an m-dimensional function represented by $c_i : \{0,1\}^m \rightarrow \mathbb{R}^+ \cup \{0\}$. The value of $c_i(\boldsymbol{x}_i)$ depends on many factors, such as the energy consumption of the sensors used for data collection. We refer to [31] to calculate the data collection cost. Assume that user i's device has L_i sensors. We assume that the power per unit time of sensor $l \in (1, 2, \ldots, L_i)$ on user i's device when collecting data is η_{il} and that the time required to stay at POI j to collect data is t_{ij}. Thus, if user i collects data at POI j, the energy cost required is $\sum_{l=1}^{L_i} \eta_{il} \cdot t_{ij}$. In addition, we define a constant, λ_j, to represent the possible transportation cost of any user moving to POI j to collect data. Therefore, when the data collection result $\boldsymbol{x}_i$ of user i is determined, the user data collection cost is

$$c_i(\boldsymbol{x}_i) = \sum_{j=1}^{m} x_{ij} \cdot \left(\sum_{l=1}^{L_i} \eta_{il} \cdot t_{ij} + \lambda_j \right). \qquad (1)$$

By obtaining a user's device type, the MCS provider can approximately determine the data collection cost for that user in advance, thereby determining the cost function in advance (a simple method is for the MCS provider to ask users to download a piece of test data to evaluate the performance of their devices).

TABLE I
COMPARISON BETWEEN OUR APPROACH AND RELATED WORKS

Related works	Mechanism Features	Resource Features	Mechanism Design Goals	Algorithm Theory	Approximation Guarantee
Yang et al. [17]	Reverse Offline	Mobile crowdsensing Task	Seller's utility	Heuristic Auction	/
Cheng et al. [18]	Reverse Offline	Mobile crowdsensing Task	Seller's valuation	Heuristic Auction	/
Zhao et al. [6]	Reverse Online	Mobile crowdsensing Task	Seller's valuation	Heuristic Auction	/
Alaei et al. [23]	Forward Online	Items	Revenue	Approx. Prophet Inequality Bayesian	$(1-1/\sqrt{d+3})^{-1}$
Kleinberg et al. [24]	Forward Online	Items	Revenue	Approx. Prophet Inequality Bayesian	$1/(4d-2)$
Dütting et al. [26]	Forward Online	Items	Social welfare (Sum of the winning users' bids)	Approx. Prophet Inequality Posted Pricing	$1/(4d-2)$
Correa et al. [27]	Forward Online	Items	Social welfare (Sum of the winning users' bids)	Approx. Prophet Inequality Posted Pricing	$1/(d+1)$
Singla et al. [28]	Reverse Online	Tasks of workers	Regret	Posted Pricing	Regret Bound
Balkanski et al. [29]	Reverse Online	Tasks of workers	Buyer's valuation	Approx. Posted Pricing	$((1-\frac{1}{\sqrt{2\pi d}})(1-\frac{1}{d}))^{-1}$
Han et al. [30]	Reverse Online	Mobile crowdsensing Task	Seller's payment	Posted Pricing	Regret Bound
Our Work	Reverse Online	Mobile crowdsensing Task	Total system utility (Refer to formula 5,6,7)	Approx. Prophet Inequality Posted Pricing	$1/(d+1)$

Additionally, lambda can cover the transportation costs of most users. We therefore assume that the cost function is generated by the device and is truthful.

MCS providers hope that more different users will participate in data collection to meet the diversity of collected data. If the same user performs too many tasks, it may cause data pollution (for example, in an MCS, which measures signal strength in the same area, the hardware on different devices will show large differences). Therefore, we limit the upper limit of data collection tasks that each user can perform to d. That is, each user is restricted to collect data at no more than d POIs.

$$\|\boldsymbol{x}_i\|_1 = \sum_{j=1}^{m} x_{ij} \leq d \tag{2}$$

The importance of each POI is different, so the number of data collection tasks required to be performed on these POIs is also different. For example, for earthquake center POIs, more users are required to participate in data collection to ensure sufficient data for analysis [32]. Conversely, for unimportant POIs, only a small amount of data needs to be collected. We use k_j to represent the maximum number of users who should cover POI j (meaning that at most k_j data collection tasks should be performed at POI j) and define a vector $\boldsymbol{k} = (k_1, k_2, \ldots k_m)^{\mathrm{T}}$ to represent the coverage requirements for all the POIs.

In addition, a vector $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)^{\mathrm{T}}$ represents the results for the number of users performing tasks at each POI. For example, y_j denotes the total number of tasks performed by users at POI j. Thus, we have

$$y_j = \sum_{i=1}^{n} x_{ij} \leq k_j, \forall j = 1, 2, \ldots, m \tag{3}$$

We also define a valuation function for the MCS as $v_0(\boldsymbol{y})$, where $v_0 : \mathbb{N}^m \rightarrow \mathbb{R}^+ \cup \{0\}$ is a nondecreasing function. In this paper, we assume that the number of tasks performed by users at POI j does not necessarily have to reach k_j because the data value collected by each user is the same and independent. As long as a user performs a task at POI j, the MCS provider will obtain the data value of v_j. The definition of v_j varies across MCS tasks. For example, in [33], the data value depends on the semantic information in the collected video, while in [34], the age of the collected data is used. To facilitate subsequent analysis, we make the simple assumption that the value function is additive. Thus, we have

$$v_0(\boldsymbol{y}) = \sum_{j=1}^{m} y_j \cdot v_j = \sum_{j=1}^{m} \sum_{i=1}^{n} x_{ij} \cdot v_j \tag{4}$$

Through the value function, the MCS provider can evaluate how much value the data collected by users can generate.

To encourage more users to participate in the data collection tasks, the MCS provider offers rewards to users who complete the data collection tasks. Specifically, if user i completes a data collection task at POI j, the MCS provider will pay a reward p_j to user i. We use the vector $\boldsymbol{p} = (p_1, p_2, \ldots p_{\mathrm{m}})^{\mathrm{T}}$ to represent the reward price offered by the MCS provider for each POI (that is, the **posted price** structure for data collection tasks).

We assume that the MCS provider needs to announce the reward prices in advance to inform all users and that users will then choose which tasks to perform based on their own circumstances. Thus, the calculation of the reward prices is crucial.

A. *Offline Optimal Total Utility Model*

In the overall system, both the users and the MCS provider wish to obtain the greatest possible utility; thus, the maximization of user utility and the maximization of MCS provider utility are two objectives of a mutual game. The reward price $\boldsymbol{p}$ is a lever: if the reward price is too high, the utility of the MCS provider will be low, whereas that of users will be high; in contrast, if the reward price is too low, no user will wish to participate in the data collection tasks, and the utility of both users and the MCS provider will be 0. The offline model is used for verification, and we find that different reward prices affect the total utility of the system. In the experiment, the offline algorithm is used only as a baseline algorithm to evaluate the superiority of our solution.

Given a reward price vector $\boldsymbol{p}$ and an allocation solution X, the utility $u_0(\boldsymbol{p}, \mathrm{X})$ of the MCS provider can be viewed as the valuation obtained at each POI minus the compensation paid to users, that is,

$$u_0(\boldsymbol{p}, \mathrm{X}) = v_0(\boldsymbol{y}) - (\boldsymbol{y})^T \boldsymbol{p} = v_0\left(\sum_{i=1}^{n} \boldsymbol{x}_i\right) - \sum_{j=1}^{m} y_j p_j = v_0\left(\sum_{i=1}^{n} \boldsymbol{x}_i\right) - \sum_{j=1}^{m}\sum_{i=1}^{n} x_{ij} p_j \tag{5}$$

The utility of user i can be defined as follows:

$$u_i(\boldsymbol{p}, \mathrm{X}) = (\boldsymbol{x}_i)^T \boldsymbol{p} - c_i(\boldsymbol{x}_i) \tag{6}$$

Thus, the total utility of the system can be defined as

$$u(\boldsymbol{p}, \mathrm{X}) = u_0(\boldsymbol{p}, \mathrm{X}) + \sum_{i=1}^{n} u_i(\boldsymbol{p}, \mathrm{X}) = v_0(\boldsymbol{y}) - \sum_{i=1}^{n} c_i(\boldsymbol{x}_i) = v_0\left(\sum_{i=1}^{n} \boldsymbol{x}_i\right) - \sum_{i=1}^{n} c_i(\boldsymbol{x}_i) \tag{7}$$

where payments have been offset.

Therefore, under a given reward price vector $\boldsymbol{p} = (p_1, p_2, \ldots p_M)^{\mathrm{T}}$, the offline optimal total utility model can be formulated as follows:

$$Maximize \quad OPT(\boldsymbol{p}, \mathcal{M}) = v_0\left(\sum_{i=1}^{n} \boldsymbol{x}_i\right) - \sum_{i=1}^{n} c_i(\boldsymbol{x}_i) \tag{8}$$

$$s.t. \quad \sum_{i=1}^{n} x_{ij} \le k_j, \ \forall j = 1, 2, \ldots, m \tag{8a}$$

$$\sum_{j=1}^{m} x_{ij} \le d, \ \forall i = 1, 2, \ldots, n \tag{8b}$$

$$c_i(\boldsymbol{x}_i) \le \sum_{j=1}^{m} p_j x_{ij}, \forall i = 1, 2, \ldots, n \tag{8c}$$

$$x_{ij} \in \{0, 1\} \tag{8d}$$

$$\boldsymbol{p} \in (\mathbb{R}^+)^m \tag{8e}$$

Constraint (8a) indicates that the number of data collection tasks at POI m cannot exceed the specified limit. Constraint (8b) indicates that for any user i, the number of tasks performed cannot be greater than d. Constraint (8c) indicates that for any user, the total cost of the tasks performed must be less than or equal to the total reward received. Constraint (8d) states that x_{ij} is a 0-1 integer decision variable, and constraint (8e) indicates that the reward price vector is an m-dimensional real-valued decision variable.

Here, we provide a simple example to illustrate how the reward prices $\boldsymbol{p}$ impact the optimal total utility of the system. It is assumed that there is one POI for which data need to be collected once. The completion of data collection at this POI yields a value of 2 for the MCS provider, and the MCS provider is willing to reward the user with a fee of 1. At this time, there is only one user in the system, and the cost of collecting data at the POI is 1.2. Thus, the cost for this user to complete the data collection task is 1.2. Therefore, the optimal solution to the problem at this time is 0 because the user's data collection cost is greater than the reward, meaning that the condition for the user to be willing to participate in data collection is not met; thus, the user's utility is 0. The MCS provider's utility is also 0 because no data are collected. However, if the MCS provider increases the reward price to 1.5, then the user's utility will be $1.5 - 1.2 = 0.3$, the MCS provider's utility will be $2 - 1.5 = 0.5$, and the optimal total utility of the system will be $0.3 + 0.5 = 0.8$. Notably, as long as the reward price set by the MCS provider exceeds 1.2 and is less than 2, the total utility of the system is always 0.8. Therefore, it is crucial for the MCS provider to determine a reasonable reward price vector $\boldsymbol{p}$.

B. Online Total Utility Model $TU(\boldsymbol{p})$

Before we discuss how to determine the reward price, we need to describe user behavior in an online MCS system because the price is closely related to user behavior. In this paper, we consider that all users arrive in the MCS system in an arbitrary order, and this assumption satisfies the online characteristic of the problem.

When user i arrives in the system, she or he will choose an allocation solution $\boldsymbol{x}_i$ from the set of POIs for which tasks currently need to be performed to maximize her or his utility u_i. The corresponding maximization problem can be represented as follows:

$$Maximize \quad u_i = (\boldsymbol{x}_i)^T \boldsymbol{p} - c_i(\boldsymbol{x}_i) \tag{9}$$

$$s.t. \quad \sum_{j=1}^{m} x_{ij} \le d, \forall i = 1, 2, \ldots, n \tag{9a}$$

$$x_{ij} \le r_{ij}, \forall i = 1, 2, \ldots, n, j = 1, 2, \ldots, m, r_{ij} \in \mathbb{N} \tag{9b}$$

$$x_{ij} \in \{0, 1\} \tag{9c}$$

where $\boldsymbol{r}_i = (r_{i1}, r_{i2}, \ldots, r_{im})^{\mathrm{T}}$ represents the vector of the remaining numbers of tasks that need to be executed at the POIs when user i arrives in the system; thus, $k_j \ge r_{ij} \ge 1$. For example, $r_{ij} = 2$ means that when user i arrives, POI j still has 2 tasks that need to be executed. When all users have selected their allocation solutions in accordance with the order in which they entered the system, then we can calculate the total utility of the system after task completion and payment according to the formula (7) as $v_0(\sum_{i=1}^{n} \boldsymbol{x}_i) - \sum_{i=1}^{n} c_i(\boldsymbol{x}_i)$.

The user arrival order clearly affects the total utility of the system. Thus, we define σ as a given user arrival order

TABLE II
FREQUENTLY USED NOTATIONS

Notation	Meaning
$\mathcal{N}$	The set of users.
$\mathcal{M}$	The set of POIs.
$\mathcal{C}$	The subset of POIs.
$\mathcal{R}$	After all users have selected, the POI set for which the data collection tasks has still remaining.
v_j	The data value that MCS provider can be obtained due to a user performing data collection at POI j.
k_j	The maximum number of data collection tasks that POI j needs to perform.
σ	The arrival order of users.
$\boldsymbol{r}_i^\sigma$	$=(r_{i1}^\sigma, r_{i2}^\sigma, ..., r_{im}^\sigma)^{\mathrm{T}}$ represents the vector of the remaining task numbers under σ when user i arrives.
$\boldsymbol{x}_i$	$=(x_{i1}, x_{i2}, \cdots, x_{im})^{\mathrm{T}}$ represents user i's data collection result.
$\boldsymbol{x}_{i,\mathcal{C}}, \boldsymbol{x}_{i,\mathcal{C}}^*$	User i's data collection result and optimal data collection result under POI set $\mathcal{C}$.
$\boldsymbol{x}_i^\sigma$	$=(x_{i1}^\sigma, x_{i2}^\sigma, \cdots, x_{im}^\sigma)^{\mathrm{T}}$ represents the data collection result for user i under σ.
$\boldsymbol{y}$	$=(y_1, y_2, \cdots, y_m)^{\mathrm{T}}$ represents the results for the number of users performing tasks at each POI.
$\boldsymbol{p}, \boldsymbol{p}^*$	Reward price vector and optimal reward price vector.
d	The upper limit of data collection tasks that each user can perform.
$v_0(.)$	The data value function of the MCS provider.
$c_i(.)$	The cost function of user i.
$u_{i,\boldsymbol{p},\mathcal{C}}^*$	The positive utility of user i' under reward price vector $\boldsymbol{p}$ and POI set $\mathcal{C}$.
$OPT(\mathcal{M})$	The offline optimal total utility under POI set $\mathcal{M}$.
$OPT(\boldsymbol{p},\mathcal{M})$	The offline optimal total utility under reward price vector $\boldsymbol{p}$ and POI set $\mathcal{M}$.
$TU(\boldsymbol{p})$	the minimum value (worst case) of the total utility under an arbitrary user arrival order.

and $\boldsymbol{r}_i^\sigma = (r_{i1}^\sigma, r_{i2}^\sigma, \ldots, r_{im}^\sigma)^T$ as the vector of the remaining task numbers under σ when user i arrives. Similarly, $\boldsymbol{x}_i^\sigma = (x_{i1}^\sigma, x_{i2}^\sigma, \ldots, x_{im}^\sigma)^{\mathrm{T}} \in \{0,1\}^m$ is defined as the data collection result for user i under σ. Because there is a limit on the number of tasks that can be performed on POIs, high-reward tasks are completed in the early stage, and users who enter later can choose only small-reward tasks to perform. The individual rationality of the mechanism can ensure that the reward for performing tasks is greater than is the cost of user data collection. Therefore, for users, entering the system as early as possible may be a better choice because doing so can enable them to obtain greater utility.

Finally, we use $TU(\boldsymbol{p})$ to represent the minimum value (worst case) of the total utility under an arbitrary user arrival order when the reward prices $\boldsymbol{p}$ are determined.

$$TU(\boldsymbol{p}) = \min_{\sigma} \left\{ v_0 \left(\sum_{i=1}^{n} \boldsymbol{x}_i^\sigma \right) - \sum_{i=1}^{n} c_i(\boldsymbol{x}_i^\sigma) \right\},$$
$$\forall x_{ij}^\sigma \le r_{ij}^\sigma, j = 1, 2, \ldots, m, x_{ij}^\sigma \in \{0,1\} \tag{10}$$

For ease of reference, Table II lists the notations that are frequently used in this paper.

IV. THE PROPERTIES OF $TU(\boldsymbol{p})$

In this section, we prove that $TU(\boldsymbol{p})$ approximates the offline optimal total utility $OPT(\boldsymbol{p},\mathcal{M})$ within an approximation ratio of $1/(d+1)$. It is worth noting that this approximation ratio can be guaranteed only when all users enter the system to perform task selection.

We define $OPT(\mathcal{M})$ as the optimal solution when the POI set is $\mathcal{M}$, regardless of the reward price; that is, constraints (8c) and (8e) are not considered in formula (8). It is obvious that $OPT(\mathcal{M}) \ge OPT(\boldsymbol{p},\mathcal{M})$. $OPT(\mathcal{M})$ is described as follows.

$$Maximize \quad OPT(\mathcal{M}) = v_0 \left(\sum_{i=1}^{n} \boldsymbol{x}_{i,\mathcal{M}} \right) - \sum_{i=1}^{n} c_i(\boldsymbol{x}_{i,\mathcal{M}}) \tag{11}$$

$$s.t. \quad \sum_{i=1}^{n} x_{ij} \le k_j, \forall j = 1, 2, \ldots, m \tag{11a}$$

$$\sum_{j=1}^{m} x_{ij} \le d, \forall i = 1, 2, \ldots, n \tag{11b}$$

$$x_{ij} \in \{0,1\} \tag{11c}$$

where $\boldsymbol{x}_{i,\mathcal{M}}$ is a feasible allocation solution for user i under POI set $\mathcal{M}$ and $\boldsymbol{x}_{i,\mathcal{M}}^*$ is the optimal allocation solution for user i under POI set $\mathcal{M}$.

Similarly, for $\mathcal{C} \subseteq \mathcal{M}$, we define $OPT(\mathcal{C}) = Maximize(v_0(\sum_{i=1}^n \boldsymbol{x}_{i,\mathcal{C}}) - \sum_{i=1}^n c_i(\boldsymbol{x}_{i,\mathcal{C}}))$ as the optimal solution under POI set $\mathcal{C}$ and $\boldsymbol{x}_{i,\mathcal{C}}^*$ as the optimal allocation solution for user i under POI set $\mathcal{C}$. We use $u_{i,\boldsymbol{p},\mathcal{C}}^*$ to denote the positive part of the utility of user i, based on the above description and a reward price $\boldsymbol{p}$.

$$u_{i,\boldsymbol{p},\mathcal{C}}^* = \left[(\boldsymbol{x}_{i,\mathcal{C}}^*)^T \boldsymbol{p} - c_i(\boldsymbol{x}_{i,\mathcal{C}}^*) \right]_+, \forall i = 1, 2, \ldots, n \tag{12}$$

In the following, we prove a theorem that states that there must exist a determined reward price vector $\boldsymbol{p}^* \in R_+^M$ that satisfies $(d+1)TU(\boldsymbol{p}^*) \ge OPT(\boldsymbol{p}^*,\mathcal{M})$. Before this theorem is proven, we need to provide some definitions. First, we adopt the following notations:

$$\boldsymbol{k}_{\mathcal{C}} = (k_1', k_2', \ldots, k_m')^{\mathrm{T}},$$
$$k_j' = \begin{cases} k_j, \ j \in \mathcal{C}, \mathcal{C} \subseteq \mathcal{M} \\ 0, \ otherwise \end{cases} \tag{13}$$

Second, we define $\mathcal{R}$ ($\mathcal{R} \subseteq \mathcal{M}$) as the set of POIs for which tasks remain at the end of the task allocation process. Accordingly, $\mathcal{M} \backslash \mathcal{R}$ is the set of POIs for which the allocated data collection tasks have been completed.

Lemma 1: For any fixed reward price vector $\boldsymbol{p} \in R_+^M$, we can obtain

$$TU(\boldsymbol{p}) \ge v_0(\boldsymbol{k}_{\mathcal{M}\backslash\mathcal{R}}) - (\boldsymbol{k}_{\mathcal{M}\backslash\mathcal{R}})^T \boldsymbol{p} + \sum_{i=1}^{n} u_{i,\boldsymbol{p},\mathcal{R}}^* \tag{14}$$

Proof: We divide $TU(\boldsymbol{p})$ into two parts: the utility of the MCS provider, u_0, and the utility of all users, $\sum_{i=1}^n u_i$. That is, $TU(\boldsymbol{p}) = u_0 + \sum_{i=1}^n u_i$.

The MCS provider utility u_0 is equal to the utility gained from the POIs for which all data collection tasks have been either completed or partially completed. Therefore, we have

$$u_0 \ge v_0(\boldsymbol{k}_{\mathcal{M}\backslash\mathcal{R}}) - (\boldsymbol{k}_{\mathcal{M}\backslash\mathcal{R}})^T \boldsymbol{p} \tag{15}$$

According to formula (9), each user's strategy is to choose the set of remaining POIs that maximizes her or his utility when she or he enters the system. Therefore,

$$u_i = \max_{\boldsymbol{x}_i} \left((\boldsymbol{x}_i^{\sigma})^T \boldsymbol{p} - c_i(\boldsymbol{x}_i^{\sigma}) \right) \geq u^*_{i,\boldsymbol{p},\mathcal{R}} = (\boldsymbol{x}^*_{i,\mathcal{R}})^T \boldsymbol{p} - c_i(\boldsymbol{x}^*_{i,\mathcal{R}}) \quad (16)$$

Here, $\boldsymbol{x}^*_{i,\mathcal{R}}$ is the optimal allocation solution for user i under POI set $\mathcal{R}$. The reason is that for any fixed reward price vector $\boldsymbol{p}$, the utility of the optimal allocation solution (under the remaining POI set $\mathcal{R}$), $u^*_{i,\boldsymbol{p},\mathcal{R}}$, must be less than or equal to the maximal utility when user i arrives.

Formulas (15) and (16) are summed to obtain

$$TU(\boldsymbol{p}) \geq v_0(\boldsymbol{k}_{\mathcal{M}\setminus\mathcal{R}}) - (\boldsymbol{k}_{\mathcal{M}\setminus\mathcal{R}})^T \boldsymbol{p} + \sum_{i=1}^{n} u^*_{i,\boldsymbol{p},\mathcal{R}} \quad (17)$$

□

Lemma 2: For any fixed reward price vector $\boldsymbol{p} \in R_+^M$, we can obtain

$$OPT(\mathcal{M}) \leq v_0(\boldsymbol{k}) - \boldsymbol{k}^T \boldsymbol{p} + \sum_{i=1}^{n} u^*_{i,\boldsymbol{p},\mathcal{M}} \quad (18)$$

Proof: From formula (11), we know that

$$OPT(\mathcal{M}) = \max_{\boldsymbol{x}_{i,\mathcal{M}}} \left(v_0\left(\sum_{i=1}^{n} \boldsymbol{x}_{i,\mathcal{M}}\right) - \sum_{i=1}^{n} c_i(\boldsymbol{x}_{i,\mathcal{M}}) \right)$$

We define $\boldsymbol{x}^*_{i,\mathcal{M}}$ as the optimal allocation solution for user i. We divide $OPT(\mathcal{M})$ into two parts, namely, the optimal utility u_0^* of the MCS provider and the optimal utility $\sum_{i=1}^{n} u_i^*$ of all users, where

$$u_i^* = u^*_{i,\boldsymbol{p},\mathcal{M}} = \left[(\boldsymbol{x}^*_{i,\mathcal{M}})^T \boldsymbol{p} - c_i(\boldsymbol{x}^*_{i,\mathcal{M}}) \right]_+ \quad (19)$$

We can obtain

$$OPT(\mathcal{M}) = u_0^* + \sum_{i=1}^{n} u_i^* = v_0\left(\sum_{i=1}^{n} \boldsymbol{x}^*_{i,\mathcal{M}}\right) - \sum_{i=1}^{n} (\boldsymbol{x}^*_{i,\mathcal{M}})^T \boldsymbol{p} + \sum_{i=1}^{n} u^*_{i,\boldsymbol{p},\mathcal{M}} \quad (20)$$

Because the optimal utility u_0^* of the MCS provider must be less than or equal to the utility when all tasks at all the POIs are completed, it can be seen that

$$u_0^* = v_0\left(\sum_{i=1}^{n} \boldsymbol{x}^*_{i,\mathcal{M}}\right) - \sum_{i=1}^{n} (\boldsymbol{x}^*_{i,\mathcal{M}})^T \boldsymbol{p} \leq v_0(\boldsymbol{k}) - \boldsymbol{k}^T \boldsymbol{p} \quad (21)$$

Thus, according to formulas (19), (20), and (21), we can obtain

$$OPT(\mathcal{M}) \leq v_0(\boldsymbol{k}) - \boldsymbol{k}^T \boldsymbol{p} + \sum_{i=1}^{n} u^*_{i,\boldsymbol{p},\mathcal{M}}$$

□

Lemma 3: There must exist a reward price vector $\boldsymbol{p}^* \in R_+^M$ that satisfies

$$v_0(\boldsymbol{k}_{\{j\}}) - k_j p_j = \sum_{i=1}^{n} x^*_{ij} u^*_{i,\boldsymbol{p}^*,\mathcal{M}},\ j = 1, 2, \ldots, m \quad (22)$$

Proof: Here, $\boldsymbol{k}_{\{j\}}$ denotes a vector in which only the value of the jth element k_j in vector $\boldsymbol{k}$ is preserved, whereas the other elements' values are set to 0 (such as $\boldsymbol{k}_{\{3\}} = (0, 0, 2, 0, 0\ldots)^{\mathrm{T}}$). According to formula (13), $v_0(\boldsymbol{k}_{\{j\}})$ is a constant, so we can modify formula (22) as follows:

$$p_j = \frac{v_0(\boldsymbol{k}_{\{j\}}) - \sum_{i=1}^{n} x^*_{ij} u^*_{i,\boldsymbol{p},\mathcal{M}}}{k_j} \quad (23)$$

We define $\mathcal{D} = [0, v_0(\boldsymbol{k})]^m \subseteq (\mathbb{R}^+ \cup \{0\})^m$ as a domain of m-dimensional vectors, and we define a function $f(\boldsymbol{p}) : \mathcal{D} \to \mathcal{D}$ for any $\boldsymbol{p} \in \mathcal{D}$. For each POI j, we define the jth dimension function $f_j(\boldsymbol{p})$ of $f(\boldsymbol{p})$ as follows:

$$f_j(\boldsymbol{p}) = \left[\frac{v_0(\boldsymbol{k}_{\{j\}}) - \sum_{i=1}^{n} x^*_{ij} u^*_{i,\boldsymbol{p},\mathcal{M}}}{k_j} \right]_+ \quad (24)$$

Here, $[\cdot]_+$ represents the positive part of the argument in brackets. $f_j(\boldsymbol{p})$ is a continuously decreasing function of $\boldsymbol{p}$. Therefore, for all $\boldsymbol{p} \in \mathcal{D}$ and each POI j, we can obtain $0 \leq f_j(\boldsymbol{p}) \leq f_j(0) \leq \frac{v_0(\boldsymbol{k})}{k_j}$. It follows that if $\boldsymbol{p} \in \mathcal{D}$, then $f(\boldsymbol{p}) \in \mathcal{D}$. According to Brouwer's theorem [35], there must exist a reward price vector $\boldsymbol{p}^*$ that satisfies formula (22). □

Lemma 4: There must exist a reward price vector $\boldsymbol{p}^* \in R_+^M$ that satisfies

$$v_0(\boldsymbol{k}) - \boldsymbol{k}^T \boldsymbol{p}^* = \sum_{j=1}^{m} \sum_{i=1}^{n} x^*_{ij} u^*_{i,\boldsymbol{p}^*,\mathcal{M}} \quad (25)$$

Proof: By summing all the values of j in formula (22), we can obtain

$$v_0\left(\sum_{j=1}^{m} \boldsymbol{k}_{\{j\}}\right) - \sum_{j=1}^{m} k_j p_j = \sum_{j=1}^{m} \sum_{i=1}^{n} x^*_{ij} u^*_{i,\boldsymbol{p}^*,\mathcal{M}} \quad (26)$$

That is,

$$v_0(\boldsymbol{k}) - \boldsymbol{k}^T \boldsymbol{p}^* = \sum_{j=1}^{m} \sum_{i=1}^{n} x^*_{ij} u^*_{i,\boldsymbol{p}^*,\mathcal{M}}$$

□

Theorem 1: There must exist a determined reward price vector $\boldsymbol{p}^* \in R_+^M$ that satisfies $(d+1)TU(\boldsymbol{p}^*) \geq OPT(\boldsymbol{p}^*, \mathcal{M})$

Proof: According to Lemma 1, we can obtain

$$TU(\boldsymbol{p}) \geq v_0(\boldsymbol{k}_{\mathcal{M}\setminus\mathcal{R}}) - (\boldsymbol{k}_{\mathcal{M}\setminus\mathcal{R}})^T \boldsymbol{p} + \sum_{i=1}^{n} u^*_{i,\boldsymbol{p},\mathcal{R}}$$

Upon the replacement of $\mathcal{R}$ with $\mathcal{M}$, the following inequality is still satisfied:

$$TU(\boldsymbol{p}) \geq v_0(\boldsymbol{k}_{\mathcal{M}\setminus\mathcal{M}}) - (\boldsymbol{k}_{\mathcal{M}\setminus\mathcal{M}})^T \boldsymbol{p} + \sum_{i=1}^{n} u^*_{i,\boldsymbol{p},\mathcal{M}} = \sum_{i=1}^{n} u^*_{i,\boldsymbol{p},\mathcal{M}} \quad (27)$$

Thus,

$$TU(\boldsymbol{p}) \geq \sum_{i=1}^{n} u^*_{i,\boldsymbol{p},\mathcal{M}} \quad (28)$$

We use the reward price vector $\boldsymbol{p}^*$ calculated according to Lemma 3 and substitute it into Lemma 2. We can see that

$$OPT(\mathcal{M}) \leq v_0(\boldsymbol{k}) - \boldsymbol{k}^T \boldsymbol{p}^* + \sum_{i=1}^{n} u^*_{i,\boldsymbol{p}^*,\mathcal{M}} = \sum_{j=1}^{m} \sum_{i=1}^{n} x^*_{ij} u^*_{i,\boldsymbol{p}^*,\mathcal{M}} + \sum_{i=1}^{n} u^*_{i,\boldsymbol{p}^*,\mathcal{M}}$$

Algorithm 1: GD for Reward Prices.

```
Input: the optimal solution x*, v_0(·), c_i(·), k
Output: the reward price vector p
Initialize p ← 0^m, α
h(p) = Σ_{j=1}^{m} (v_0(k_{{j}}) − k_j p_j − Σ_{i=1}^{n} x*_{ij} u*_{i,p,M})^2
while h(p) > 10^{−6} do
  Optimize p using gradient descent
  p = p − α · ∇_p h(p)
end while
return p
```

$$= \sum_{i=1}^{n}\left(\sum_{j=1}^{m} x_{ij}^{*}+1\right) u_{i,\boldsymbol{p}^{*},\mathcal{M}}^{*}$$

$$\leq (d+1)\sum_{i=1}^{n} u_{i,\boldsymbol{p}^{*},\mathcal{M}}^{*} \leq (d+1) TU(\boldsymbol{p}^{*}) \quad (29)$$

where the first equation in formula (29) uses Lemma 4 and the last line of inequalities is based on $\sum_{j=1}^{m} x_{ij}^{*} \leq d$. Because $OPT(\mathcal{M}) \geq OPT(\boldsymbol{p},\mathcal{M})$, we obtain $(d+1)TU(\boldsymbol{p}^{*}) \geq OPT(\boldsymbol{p}^{*},\mathcal{M})$. □

V. Determining $\boldsymbol{p}^*$ Using a Gradient Descent Algorithm

Under the assumption that we have the optimal solution $OPT(\mathcal{M})$ (which can be obtained through CPLEX [36]), the valuation function $v_0(\cdot)$ and the user cost function $c_i(\cdot)$, we can determine the reward price vector $\boldsymbol{p}^*$ in accordance with Lemma 3. We design a gradient descent (GD) algorithm to obtain the optimal reward price solution. The basic idea is to construct a function

$$h(\boldsymbol{p}) = \sum_{j=1}^{m}\left(v_0(\boldsymbol{k}_{\{j\}}) - k_j p_j - \sum_{i=1}^{n} x_{ij}^{*} u_{i,\boldsymbol{p},\mathcal{M}}^{*}\right)^2 \quad (30)$$

and use GD to update $\boldsymbol{p}$ on the basis of $\nabla_{\boldsymbol{p}} h(\boldsymbol{p})$. When $h(\boldsymbol{p}) = 0$, $\boldsymbol{p}$ is the required reward price vector. Notably, formulas (12) and (25) indicate that $(v_0(\boldsymbol{k}_{\{j\}}) - k_j p_j - \sum_{i=1}^{n} x_{ij}^{*} u_{i,\boldsymbol{p},\mathcal{M}}^{*})$ in the $h(\boldsymbol{p})$ function is monotonic. Moreover, according to Lemma 3, there exists a reward price vector that makes $h(\boldsymbol{p})$ equal to 0. Therefore, theoretically, the function $h(\boldsymbol{p})$ should converge quickly, which is also demonstrated in subsequent experiments. Therefore, we can use gradient descent to solve the optimal price vector. The proposed algorithm can efficiently calculate the reward prices. The details are shown in Algorithm 1.

In this algorithm, we use the GD approach to calculate the optimal reward prices. The inputs are the optimal solution $\boldsymbol{x}^*$, the MCS valuation function $v_0(\cdot)$, the cost function $c_i(\cdot)$ for each user, and the POI coverage requirement vector $\boldsymbol{k}$. α is the learning rate. The design of the function $h(\boldsymbol{p})$ satisfies formula (22). The algorithm executes multiple rounds of updating $\boldsymbol{p}$. When the condition $h(\boldsymbol{p}) < 10^{-6}$ is met, a reward price vector $\boldsymbol{p}$ is found that approximately satisfies Lemma 3, and the algorithm terminates. The running time of this algorithm depends on the number of POIs, the number of users, the constraints of the data collection task, and the number of iterations. The number of iterations is related to the learning rate and the termination condition.

TABLE III
Example Results Under Different User Arrival Orders

User arrival order	Total system utility	MCS provider utility	User utility	Payment
$u1, u2, u3$	170.0	86.88	83.12	253.12
$u1, u3, u2$	165.0	86.88	78.12	253.12
$u2, u1, u3$	170.0	86.88	83.12	253.12
$u2, u3, u1$	175.0	86.88	88.12	253.12
$u3, u1, u2$	165.0	86.88	78.12	253.12
$u3, u2, u1$	175.0	86.88	88.12	253.12

A. A Simple Example

We consider a simple example with two POIs, A and B, and 3 users, $u1$, $u2$, and $u3$. Collecting data once at POI A can yield a value of 120 for the MCS provider. Similarly, collecting data once at POI B can yield a value of 100. At POI A, data need to be collected 2 times, and at POI B, data need to be collected 3 times. The data collection costs of $u1$ at POIs A and B are (70,40). Similarly, the collection costs of $u2$ are (60,75), and the collection costs of $u3$ are (65,80). The number of data collection tasks performed by each user cannot exceed $d = 2$.

According to Algorithm 1, the final reward prices for POIs A and B are 91.25 and 70.62, respectively. The optimal total utility of the system without reward price constraints is $OPT(\{A, B\}) = 220$, where $y_A = \{u_2, u_3\}$ and $y_B = \{u_1, u_2, u_3\}$. The optimal total system utility with reward price constraints is $OPT(\boldsymbol{p}, \{A, B\}) = 175$, where $y'_A = \{u_2, u_3\}$ and $y'_B = \{u_1\}$. Table III shows the results obtained for different user arrival orders with the reward prices determined by Algorithm 1.

For illustration, we describe in detail the case in which the user arrival order is ($u1, u2, u3$). First, $u1$ enters the system. At this time, data need to be collected at both POIs A and B. The reward prices are 91.25 and 70.62, respectively. The utilities that $u1$ can obtain at POIs A and B are $91.25 - 70 = 21.25$ and $70.62 - 40 = 30.62$, respectively. Therefore, $u1$ obtains a total utility of $21.25 + 30.62 = 51.87$. The utility obtained by the MCS provider is $120 - 91.25 + 100 - 70.62 = 58.13$, and the payment made is $91.25 + 70.62 = 161.87$. Next, $u2$ enters the system. At this time, data still need to be collected at both POIs A and B. The utility that $u2$ can obtain at POI A is $91.25 - 60 = 31.25$, whereas the utility that can be obtained at POI B is $70.62 - 75 = -4.38$; thus, $u2$ chooses not to collect data at POI B. Ultimately, the utility of $u2$ is 31.25, the utility obtained by the MCS provider is $120 - 91.25 = 28.75$, and the payment made is 91.25. Finally, $u3$ enters the system. At this time, data need to be collected only at POI B. Since the utility that can be obtained by $u3$ at POI B is $70.62 - 80 = -9.38$, $u3$ does not select any POI for data collection. The user utility, MCS provider utility, and payment are all 0. When the above results are summed, the total utility of the MCS provider is $58.13 + 28.75 = 86.88$, the total utility of the users is $51.87 + 31.25 = 83.12$, the total payment made by the MCS provider is $161.87 + 91.25 = 253.12$, and the total utility of the system is $86.88 + 83.12 = 170$. The analysis for other user orders is similar.

(a) Collection area (b) POIs

Fig. 2. Establishment of POIs on real terrain.

This example shows that $(u2, u3, u1)$ is the optimal user arrival order, resulting in a total system utility that is consistent with the optimal solution $OPT(\boldsymbol{p}, \{A, B\})$, whereas $(u1, u3, u2)$ and $(u3, u1, u2)$ are the worst cases but still result in a total system utility that is not much different from the optimal solution. In the next section, we present more detailed experimental analyses.

VI. Experiments and Analyses

A. Experimental Setting

1) To ensure the reliability of the experimental data, we set up various data collection scenarios on real terrain (Fig. 2). In the experimental area, we set up 6 POIs ($m = 6$), publish tasks for signal strength detection at these POIs to assist in choosing base station sites, and recruit users to collect data for these POIs. The data value of a POI is evaluated according to the population density of the selected area. For example, densely populated areas have higher data values and a greater number of data collection requirements.
2) We used data from 250 users and set the cost function for each user based on the mobile device type and hardware information (such as the power consumption of the CPU, GPS and baseband chips). Based on the above information, we can calculate the posted price of each POI.
3) Regarding the user's decision, we assume that when he or she enters the MCS system, he or she has an expected compensation for performing data collection tasks, which is used to evaluate whether to accept the reward prices for the POIs. Each user's expected compensation for each POI was generated according to a normal distribution with a mean of 50% of the POI data value and a variance of 5.
4) For each experimental indicator, 500 groups of data samples for different users were extracted from the dataset, and the user arrival order was determined for each group. These data were subsequently fed into different algorithms for experimental evaluation, and the average values of the results were plotted.
5) We compared the proposed PostedPrice algorithm with the $OPT(\boldsymbol{p}, \mathcal{M})$ solution as well as the OMZ [6] and FixedPrice algorithms. OMZ is an excellent algorithm for online MCS discrete task allocation. In the FixedPrice algorithm, a fixed reward price is set based on 40% of the data value for each POI task. (For example, if the data value obtained by the MCS provider for data collection at a POI is 100, then the reward price offered to the user is set to 40 for this POI). Notably, we have made some improvements to the OMZ algorithm. In OMZ, it is assumed that the number of users arriving at each timestamp follows a Poisson distribution with $\lambda = 5$; this assumption is used to calculate the winning users and the payment prices. Moreover, the data values that may be obtained from all the POIs are summed to serve as the budget in the OMZ algorithm.
6) All algorithms used in these experiments were implemented in Python. The hardware configuration of the experimental platform was as follows: the processor was an AMD Ryzen 7 5800H CPU with 16 GB of memory, and the storage device had a 512 GB SSD.
7) We have uploaded the code used in the experiments to https://github.com/YNU-DMC-ReversePostedPricing/ReversePostedPricing.

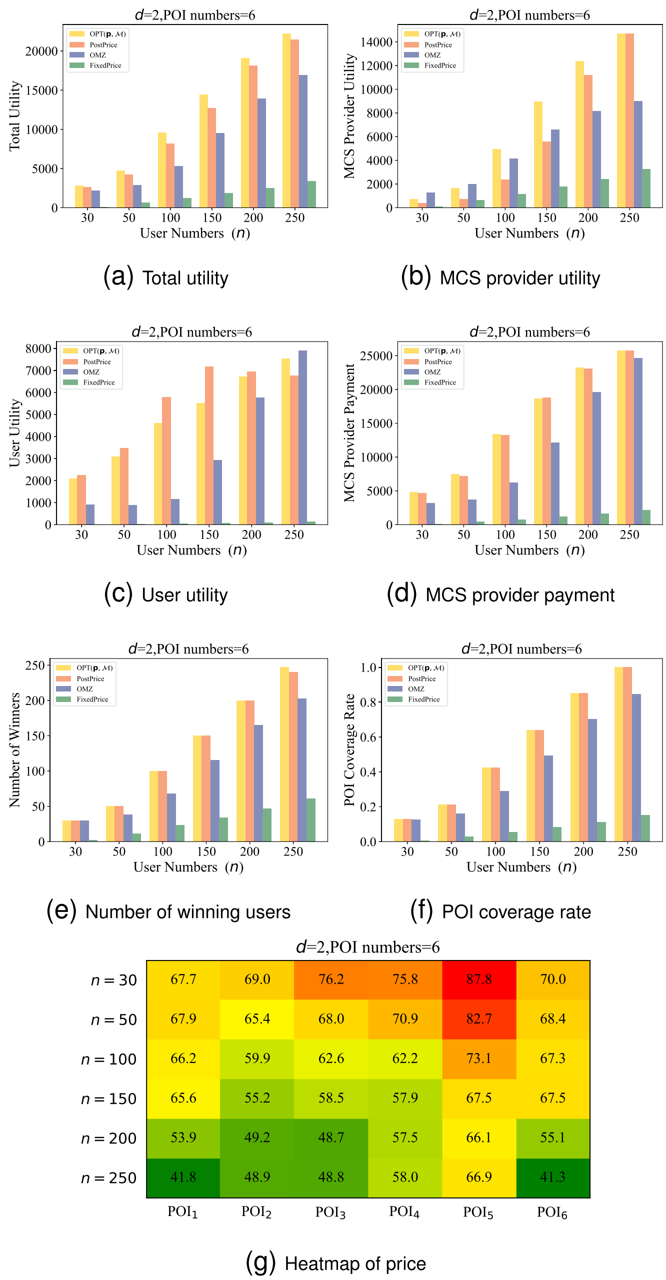
B. Experimental Results

1) Impact of the Number of Users, n: The main purpose of this experiment is to analyze how the total utility of the system, the MCS provider utility, the user utility, the total payment paid out by the MCS provider, the number of winning users and the POI coverage rate change under different algorithms as the number of users in the MCS system varies. The number of POIs is 6, and the POI coverage limit for each user is $d = 2$. The value of k_j for each POI lies within the interval [50,150].

Fig. 3(a) shows the differences in total utility between the different algorithms. The total utility is the sum of the user utility and the MCS provider utility and is one of the main optimization objectives in this article. The total utility of all four algorithms increases with the number of users, due primarily to the fact that the increase in the number of users leads to more winning users, thereby resulting in a greater number of completed transactions. The $OPT(\boldsymbol{p}, \mathcal{M})$ and PostedPrice solutions yield relatively high total utility. The reason is that both algorithms obtain the total utility based on the optimal reward price vector $\boldsymbol{p}^*$. $\boldsymbol{p}^*$ plays a significant role in driving the market, allowing the system to achieve high utility. The difference between PostedPrice and $OPT(\boldsymbol{p}, \mathcal{M})$ is that PostedPrice depends not only on the user set but also on the user order, whereas $OPT(\boldsymbol{p}, \mathcal{M})$ depends only on the user set. In contrast, FixedPrice results in low total system utility. This outcome is due primarily to its conservative pricing strategy, which strictly selects high-quality (i.e., low-cost) users. Such a strategy discourages broader participation, limiting the number of winning users and ultimately reducing total utility.

Fig. 3(b) shows the differences in the MCS provider utility. The MCS provider utility of all four algorithms increases with the number of users, due primarily to the rise in the number of winning users. The MCS provider utility of the PostedPrice mechanism is very small at the beginning, due mainly to the fact that the reward price on POIs is set high to attract more users to participate in task execution. As the number of users increases, the reward price will gradually decrease. The data value generated by the execution of more tasks plus the lower reward price (this can be seen in Fig. 3(g)) will cause the MCS provider utility to grow sharply. Conversely, the MCS provider utility of the OMZ and FixedPrice algorithms is linearly related to the growth of the number of users, which is related to the way in which their algorithms are designed.

Fig. 3(c) shows the differences in the user utility. The user utility under the PostedPrice mechanism shows a trend of initially increasing and then slightly decreasing. This phenomenon arises because, when the number of users is sufficient to cover



(a) Total utility (b) MCS provider utility (c) User utility (d) MCS provider payment (e) Number of winning users (f) POI coverage rate (g) Heatmap of price

Fig. 3. Impact of the number of users, n.

all POIs, continuing to add users will only reduce the reward price (because the MCS provider has more users from which to choose), thereby reducing the user's utility. This can also be seen from Fig. 3(e). When the number of users reaches 250, not all users can win. Theoretically speaking, although a larger user base results in more winners, the optimal reward prices in $\boldsymbol{p}^*$ are moderately reduced to balance formula (25). In contrast, FixedPrice results in near-zero user utility, reflecting an overly conservative pricing strategy that sets reward levels too low for most users to gain any meaningful profit from participation. Generally speaking, if the number of users is not enough to cover the POIs, then increasing the number of users will definitely lead to an increase in user utility. At the same time, each POI will offer a high price to attract users to participate in task execution. However, under the premise that the number of users is sufficient to cover the POIs, continuing to increase the number of users will not bring about an increase in total utility. Instead, it will lead to a decrease in the reward price at the POIs (Fig. 3(g)), thereby reducing user utility. This phenomenon is in line with market laws.

Fig. 3(d) shows the fees that the MCS provider needs to pay out under the different algorithms. As the number of users increases, the MCS provider payment also increases under all four algorithms. The $OPT(\boldsymbol{p}, \mathcal{M})$ and PostedPrice solutions result in nearly identical payment. The reason is that these two algorithms use the same reward prices $\boldsymbol{p}^*$. Furthermore, this situation also indicates that even though $OPT(\boldsymbol{p}, \mathcal{M})$ and PostedPrice have different allocation solutions, $\boldsymbol{p}^*$ provides nearly identical rewards to all users for each POI, which helps stabilize the market and prevents users from becoming overly biased toward any particular POI.

Fig. 3(e) shows the changes in the number of winning users. As the number of users increases, the number of winning users also increases under all four algorithms. It is noteworthy that when $n = 250$, the number of winners under the PostedPrice mechanism is lower than that of $OPT(\boldsymbol{p}, \mathcal{M})$. The PostedPrice mechanism allows users to freely select POIs—as long as the utility of executing a task is positive, users will choose as many POIs as possible to maximize their own benefit. In contrast, $OPT(\boldsymbol{p}, \mathcal{M})$ is designed solely to maximize total system utility and does not take truthfulness into account. Therefore, on the premise that the POI coverage requirements are met, each user in PostedPrice generally executes d POIs, whereas in $OPT(\boldsymbol{p}, \mathcal{M})$, some users may execute fewer than d POIs. As a result, PostedPrice has fewer winning users than does $OPT(\boldsymbol{p}, \mathcal{M})$.

Fig. 3(f) shows the POI coverage rate (task completion ratio). As the number of users increases, the POI coverage rate also increases under all four algorithms. Under reasonable reward prices $\boldsymbol{p}^*$, users actively participate in MCS tasks, and the $OPT(\boldsymbol{p}, \mathcal{M})$ and PostedPrice algorithms achieve high allocation efficiency. However, under the FixedPrice and OMZ algorithms, all MCS tasks still cannot be completed even when the number of users reaches 250. The reason for this is that the threshold in the OMZ algorithm is unstable, making it difficult to achieve 100% coverage, while the FixedPrice algorithm sets a very low price, requiring more users to achieve 100% coverage.

Fig. 3(g) presents the heatmap of pricing across different POIs. As shown in the figure, the overall trend in prices is decreasing, which is consistent with fundamental market dynamics—when the number of users is limited, the service provider tends to offer higher prices to incentivize participation. As the user population increases and competition intensifies, lower prices are sufficient to attract users. In addition, POI_5 exhibits the greatest data valuation, while POI_1 and POI_6 exhibit the lowest data valuation. This pattern is clearly reflected in the heatmap—POIs with higher data valuations are associated with higher prices, whereas those with lower valuations correspond to lower prices.

The above results prove that the PostedPrice algorithm can ensure high system performance and stability even with widely varying numbers of users.

2) Impact of the Per-User POI Coverage Limit, d: The main purpose of this experiment is to analyze how the total utility of the system, the MCS provider utility, the user utility, the

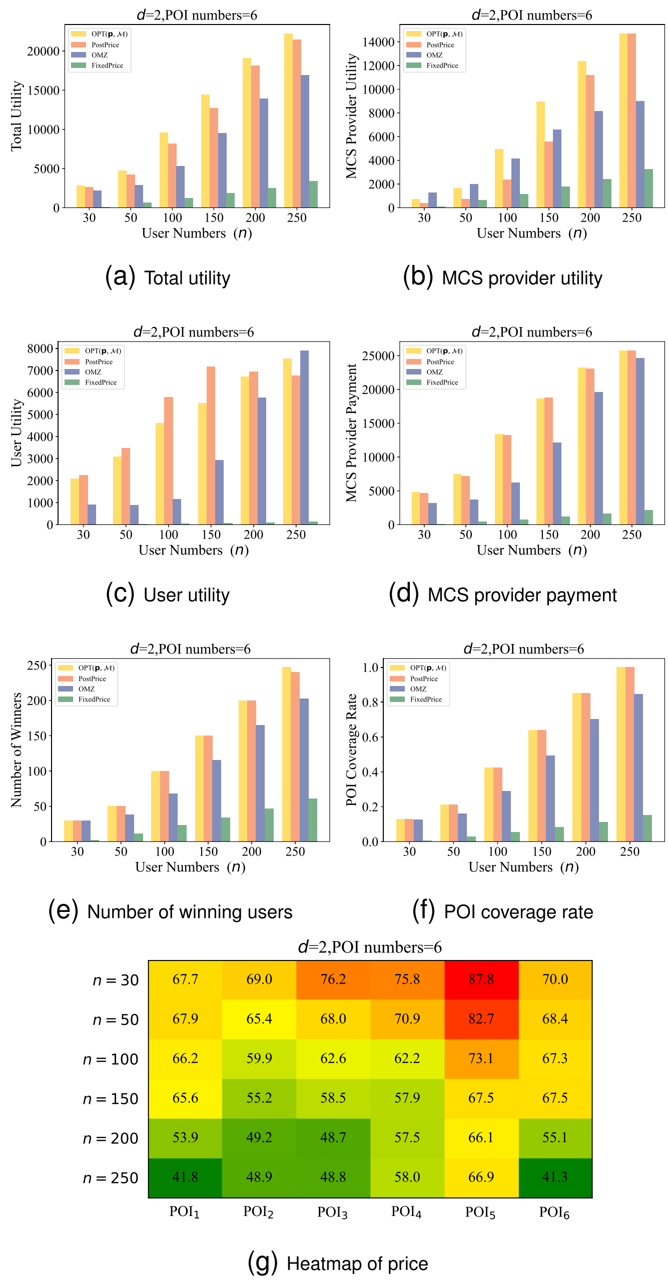


Fig. 4. Impact of the per-user POI coverage limit, d.

total payment paid out by the MCS provider, the number of winning users and the POI coverage rate change under different algorithms as the per-user POI coverage limit d varies. The number of POIs is 6, and the number of users is $n = 100$. The value of k_j for each POI lies within the interval [50,150].

Fig. 4(a) shows the changes in the total utility of the system. The total utility of $OPT(\boldsymbol{p}, \mathcal{M})$ continues to increase because some users incur low costs across multiple POIs. For the PostedPrice mechanism, total utility initially rises as each user becomes capable of covering a broader range of POIs. However, it subsequently declines. The reason for this is mainly that the users who enter early cover too many POIs, thus leaving those who enter later with no tasks to perform. This situation is a tradeoff inherent to the PostedPrice mechanism, which emphasizes truthfulness and online decision-making. Although the total utility gap between PostedPrice and OMZ narrows as d increases, PostedPrice still has the advantage. Under the FixedPrice strategy, due to its low pricing policy, increasing d yields a limited improvement in total utility.

Fig. 4(b) shows the MCS provider utility under the different algorithms. The trend observed for the $OPT(\boldsymbol{p}, \mathcal{M})$ and PostedPrice mechanisms can be attributed to the same underlying factors as those illustrated in Fig. 4(a). In contrast, the MCS provider utility fluctuates under the OMZ algorithm because of the multistage allocation process and algorithm threshold restrictions.

Fig. 4(c) shows the results for user utility. The user utility of $OPT(\boldsymbol{p}, \mathcal{M})$, PostedPrice and OMZ shows a trend of increasing first and then decreasing. When $d = 4$, the user utility of all algorithms begins to stabilize, as the coverage has approached saturation, and increasing d can only lead to more intense user competition and lower reward prices on POIs. Under the FixedPrice algorithm, because the fixed reward prices are lower, the users' utility is also lower.

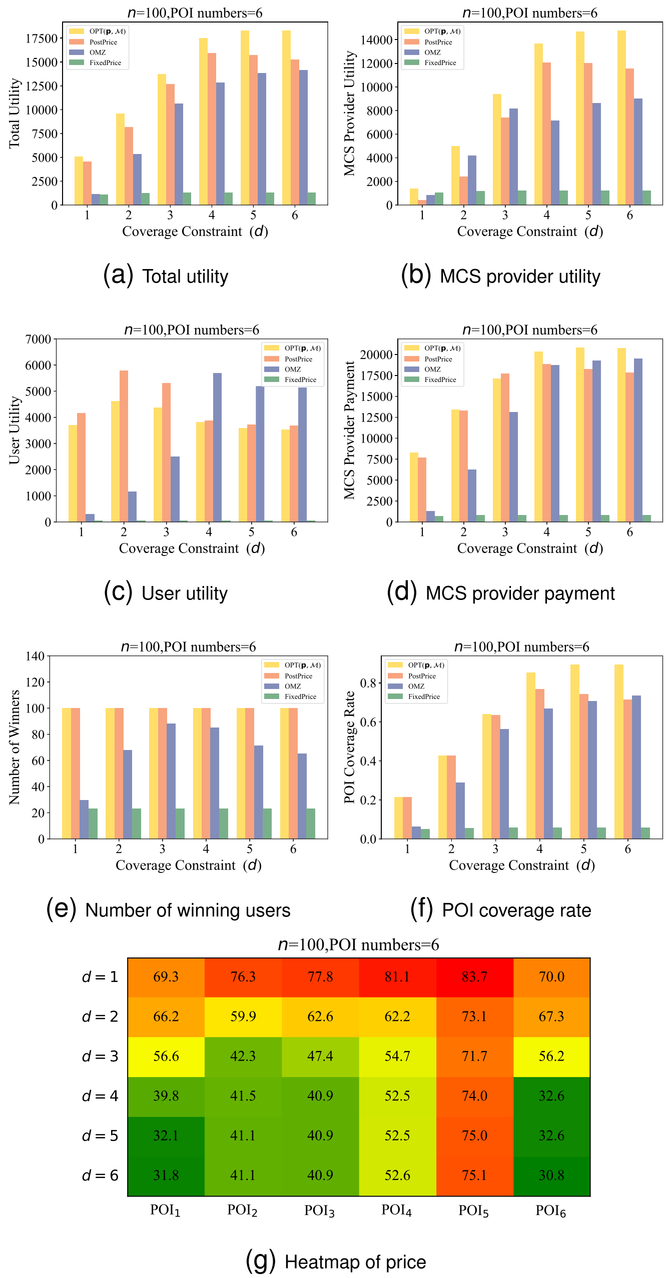
Fig. 4(d) shows the fees paid out by the MCS provider. The trend observed for the $OPT(\boldsymbol{p}, \mathcal{M})$ and PostedPrice mechanisms can be attributed to the same underlying factors as those illustrated in Fig. 4(a). For the OMZ algorithm, as the number of users increases, the threshold calculated by this algorithm increases, causing the payment fee to gradually decrease. For the FixedPrice algorithm, because the reward price offered for each POI is fixed, the overall payment fee shows minimal changes.

Fig. 4(e) shows the differences in the number of winning users. Both $OPT(\boldsymbol{p}, \mathcal{M})$ and PostedPrice result in the largest number of winning users. This outcome demonstrates that setting reasonable reward prices can significantly enhance market efficiency. As the number of users entering OMZ is phased and only some users can be selected in each round, the number of winning users is not high. For the FixedPrice algorithm, the lower fixed reward prices already result in a small number of winning users, and thus, there is not much change.

Fig. 4(f) shows the results for the POI coverage rate. The reason that $OPT(\boldsymbol{p}, \mathcal{M})$ does not achieve 100% coverage is that each user can cover only the same POI once, and the maximum value of k_j that we set (i.e., 150) is insufficient to fully cover all 100 users. Both $OPT(\boldsymbol{p}, \mathcal{M})$ and PostedPrice achieve high-level coverage. This result demonstrates that a well-designed reward price vector, $\boldsymbol{p}^*$, can effectively drive the market toward high-level coverage. In contrast, because the FixedPrice algorithms inherently result in few winning users, the task completion rate is also low.

Fig. 4(g) presents the heatmap of pricing across different POIs. As shown in the figure, the price vector $\boldsymbol{p}^*$ decreases. The main reason for this is that the increase in d leads to more intense competition among users, which reduces the reward price on POIs. Specifically, in formula (25), a user's utility is accumulated across multiple POIs. As a result, under the same coverage level, the right-hand side of formula (25) becomes larger when a user performs multiple POIs compared to when they perform only one.

The above results prove that the PostedPrice algorithm can ensure high system performance and stability even with widely



(a) Total utility (b) MCS provider utility

(c) User utility (d) MCS provider payment

(e) Number of winning users (f) POI coverage rate

(g) Heatmap of price

Fig. 5. Impact of the number of POIs, m.

varying limits d on the number of POIs that can be covered by each user.

3) Impact of the Number of POIs m: In this experiment, we further introduce 6 additional POIs to extend the evaluation. The main purpose of this experiment is to analyze how the total utility of the system, the MCS provider utility, the user utility, the total payment paid out by the MCS provider, the number of winning users and the POI coverage rate change under different algorithms as the number of POIs in the MCS system vary. The POI coverage limit for each user is $d = 2$, and the number of users is $n = 250$. The value of k_j for each POI lies within the interval [30,70].

Fig. 5(a) shows the changes in the total utility of the system. $OPT(\boldsymbol{p}, \mathcal{M})$ and PostedPrice achieve relatively high total utility, which is consistent with our optimization objectives. When $m = 11$, the total utilities of $OPT(\boldsymbol{p}, \mathcal{M})$, PostedPrice, and OMZ begin to stabilize, as all users have been fully utilized to execute POI tasks. In contrast, FixedPrice consistently yields the lowest total utility due to its low-pricing strategy.

Fig. 5(b) shows the differences in the MCS provider utility. For $OPT(\boldsymbol{p}, \mathcal{M})$, as the number of POIs increases, the MCS provider utility continues to increase, which is due to the optimal solution characteristics. For the PostedPrice mechanism, before the number of POIs is 11, the MCS provider utility continues to increase because the number of users is greater than the number of tasks to be performed, and thus, a larger number of users means greater utility. However, when the number of POIs reaches 12, the MCS provider's utility decreases because the number of tasks to be performed exceeds the number of users, causing the MCS provider to increase the reward price to incentivize more users. Conversely, FixPrice adopts a low-price strategy, which yields high MCS provider utility at the expense of user utility. OMZ shows noticeable fluctuations, attributed primarily to the threshold-based selection mechanism.

Fig. 5(c) shows the differences in user utility. For $OPT(\boldsymbol{p}, \mathcal{M})$, PostedPrice and FixedPrice algorithms, as the number of POIs increases, user utility tends to increase. The reason for this is that when the number of users is fixed, increasing the number of POIs means that more tasks need to be performed, and thus, user utility increases. However, the user utility of FixPrice is low and OMZ shows fluctuations, the reasons for which are analyzed in Fig. 5(b). At the same time, the experimental results show that both PostedPrice and $OPT(\boldsymbol{p}, \mathcal{M})$ maintain a balanced utility between the MCS provider and users, suggesting that a carefully selected reward price vector $\boldsymbol{p}^*$ can achieve a fair utility allocation in the market.

Fig. 5(d) shows the fees paid out by the MCS provider. As the number of POIs increases, both $OPT(\boldsymbol{p}, \mathcal{M})$ and PostedPrice exhibit a corresponding growth. The initial increase is attributed to users' more active participation in MCS tasks, while the subsequent growth is driven by the increase in the reward price vector $\boldsymbol{p}^*$.

Fig. 5(e) shows the number of winning users. As the number of POIs increases, both $OPT(\boldsymbol{p}, \mathcal{M})$ and PostedPrice are able to reach 250 winning users, indicating that a well-designed reward vector, $\boldsymbol{p}^*$, can effectively incentivize broader user participation. However, for the OMZ algorithm, the number of winning users grows slowly due to the threshold-based selection mechanism. In addition, FixedPrice fails to sufficiently incentivize users due to its low-pricing strategy, resulting in a limited number of winning users.

Fig. 5(f) shows the results for the POI coverage rate. Except for FixedPrice, the coverage rates of the other three algorithms exhibit a declining trend. The reason for this is that users' coverage capacity in these algorithms has reached its limit, while the number of coverable POIs continues to increase. Both $OPT(\boldsymbol{p}, \mathcal{M})$ and PostedPrice are still able to maintain the highest coverage levels. In contrast, the inherent design of the FixedPrice mechanism restricts the number of winning users, resulting in a lower overall task completion rate. For the OMZ algorithm, achieving greater coverage remains challenging due to its multistage allocation strategy and threshold-based constraints.

Fig. 5(g) presents the heatmap of pricing across different POIs. As shown in the figure, the reward prices remain stable at the beginning. When $m = 11$, the reward prices start to increase. The main reason for this is that the increase in the number of POIs requires the MCS system provider to increase reward prices to attract more users to perform tasks. This trend aligns well with general market principles—when users are able to fully cover all POIs, the market remains stable and no price adjustment is necessary, but when POIs cannot be sufficiently covered, the service provider raises the reward prices to increase user participation.

The above results prove that the PostedPrice algorithm can ensure high system performance and stability, even with a widely varying number of POIs.

VII. Conclusion

In practical applications, incorporating posted pricing theory into MCSs is an innovative approach. This approach can avoid the problems of low execution efficiency or unsatisfactory utility optimization of traditional mechanism design and naturally possesses the desired economic properties of mechanism design, such as truthfulness and individual rationality. Theoretically, the posted pricing mechanism proposed in this article allows MCS providers to find a theoretical basis and calculation method for reward pricing for data collection tasks and achieves total utility within a provable lower bound. In experiments, we observed that by setting reasonable reward prices, the posted pricing mechanism improves the operating efficiency of the market and achieves high utility for both the MCS provider and users. However, the work presented in this article still has limitations. For example, the mechanism currently relies on knowing the cost information of all users, which is a heavy burden for MCS providers. An effective improvement is to combine the dynamic pricing method with the posted pricing mechanism to determine the reward price in real time based on user information and environmental conditions. This approach can improve the execution efficiency and accuracy of task allocation. We will address this issue in future research.

Acknowledgment

The authors would like to thank the anonymous reviewers for their helpful remarks.

References

[1] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: Current state and future challenges," *IEEE Commun. Mag.*, vol. 49, no. 11, pp. 32–39, Nov. 2011, doi: 10.1109/MCOM.2011.6069707.

[2] J. Zhang, Y. Zhang, H. Wu, and W. Li, "An ordered submodularity-based budget-feasible mechanism for opportunistic mobile crowdsensing task allocation and pricing," *IEEE Trans. Mobile Comput.*, vol. 23, no. 2, pp. 1278–1294, Feb. 2024, doi: 10.1109/TMC.2022.3232513.

[3] L. Liu et al., "Evenness-aware data collection for edge-assisted mobile crowdsensing in internet of vehicles," *IEEE Internet Things J.*, vol. 10, no. 1, pp. 1–16, Jan. 2023.

[4] C. Ying, H. Jin, J. Li, X. Si, and Y. Luo, "Incentive mechanism design via smart contract in blockchain-based edge-assisted crowdsensing," *Front. Comput. Sci.*, vol. 19, no. 3, 2025, Art. no. 193802.

[5] J. Zhang, M. Zong, A. V. Vasilakos, and W. Li, "UAV base station network transmission-based reverse auction mechanism for digital twin utility maximization," *IEEE Trans. Netw. Service Manag.*, vol. 21, no. 1, pp. 324–340, Feb. 2024, doi: 10.1109/TNSM.2023.3301522.

[6] D. Zhao, X.-Y. Li, and H. Ma, "Budget-feasible online incentive mechanisms for crowdsourcing tasks truthfully," *IEEE/ACM Trans. Netw.*, vol. 24, no. 2, pp. 647–661, Apr. 2016, doi: 10.1109/TNET.2014.2379281.

[7] K. Li, "Scheduling independent tasks on multiple cloud-assisted edge servers with energy constraint," *J. Parallel Distrib. Comput.*, vol. 184, 2024, Art. no. 104781.

[8] K. Li, "Scheduling precedence constrained tasks for mobile applications in fog computing," *IEEE Trans. Serv Comput.*, vol. 16, no. 3, pp. 2153–2164, May/Jun. 2023.

[9] H. Zhou, T. Wu, X. Chen, S. He, D. Guo, and J. Wu, "Reverse auction-based computation offloading and resource allocation in mobile cloud-edge computing," *IEEE Trans. Mobile Comput.*, vol. 22, no. 10, pp. 6144–6159, Oct. 2023, doi: 10.1109/TMC.2022.3189050.

[10] L. Blumrosen and N. Nisan, *Algorithmic Game Theory*. New York, NY, USA: Cambridge Univ. Press, 2007.

[11] J. Zhang, P. Chen, X. Yang, H. Wu, and W. Li, "An optimal reverse affine maximizer auction mechanism for task allocation in mobile crowdsensing," *IEEE Trans. Mobile Comput.*, vol. 24, no. 8, pp. 7475–7488, Aug. 2025.

[12] L. Mashayekhy, M. M. Nejad, and D. Grosu, "A PTAS mechanism for provisioning and allocation of heterogeneous cloud resources," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 9, pp. 2386–2399, Sep. 2015, doi: 10.1109/TPDS.2014.2355228.

[13] R. Wang, "Auctions versus posted-price selling," *Amer. Econ. Rev.*, vol. 83, pp. 838–851, 1993.

[14] Y. Qu et al., "Posted pricing for chance constrained robust crowdsensing," *IEEE Trans. Mobile Comput.*, vol. 19, no. 1, pp. 188–199, Jan. 2020.

[15] Z. Zhang, E. K. P. Chong, A. Pezeshki, and W. Moran, "String submodular functions with curvature constraints," *IEEE Trans. Autom. Control*, vol. 61, no. 3, pp. 601–616, Mar. 2016, doi: 10.1109/TAC.2015.2440566.

[16] J. Kleinberg, E. Ryu, and E. Tardos, "Ordered submodularity and its applications to diversifying recommendations," 2022, *arXiv:2203.00233*.

[17] D. Yang, G. Xue, X. Fang, and J. Tang, "Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing," in *Proc. 18th Annu. Int. Conf. Mobile Comput. Netw.*, New York, NY, USA, 2012, pp. 173–184, doi: 10.1145/2348543.2348567.

[18] Y. Cheng, X. Wang, P. Zhou, X. Zhang, and W. Wu, "Freshness-aware incentive mechanism for mobile crowdsensing with budget constraint," *IEEE Trans. Serv. Comput.*, vol. 16, no. 6, pp. 4248–4260, Nov./Dec. 2023.

[19] C.-L. Hu, K.-Y. Lin, and C. K. Chang, "Incentive mechanism for mobile crowdsensing with two-stage stackelberg game," *IEEE Trans. Serv. Comput.*, vol. 16, no. 3, pp. 1904–1918, May/Jun. 2023.

[20] E. Wang, Y. Yang, J. Wu, W. Liu, and X. Wang, "An efficient prediction-based user recruitment for mobile crowdsensing," *IEEE Trans. Mobile Comput.*, vol. 17, no. 1, pp. 16–28, Jan. 2018.

[21] M. Hajiaghayi, R. Kleinberg, and T. W. Sandholm, "Automated mechanism design and prophet inequalities," in *Proc. 22nd Conf. Artif. Intell.*, 2007, pp. 58–65.

[22] U. Krengel and L. Sucheston, "On semiamarts, amarts, and processes with finite value," *Probab. Banach Spaces*, vol. 4, no. 197/266, pp. 1–2, 1978.

[23] S. Alaei, "Bayesian combinatorial auctions: Expanding single buyer mechanisms to many buyers," *SIAM J. Comput.*, vol. 43, no. 2, pp. 930–972, 2014.

[24] R. Kleinberg and S. M. Weinberg, "Matroid prophet inequalities," in *Proc. 44th Annu. ACM Symp. Theory Comput.*, New York, NY, USA, 2012, pp. 123–136, doi: 10.1145/2213977.2213991.

[25] M. Feldman, O. Svensson, and R. Zenklusen, "Online contention resolution schemes," in *Proc. 27th Annu. ACM-SIAM Symp. Discrete Algorithms*, SIAM, 2016, pp. 1014–1033.

[26] P. Dütting, M. Feldman, T. Kesselheim, and B. Lucier, "Prophet inequalities made easy: Stochastic optimization by pricing nonstochastic inputs," *SIAM J. Comput.*, vol. 49, no. 3, pp. 540–582, 2020, doi: 10.1137/20M1323850.

[27] J. Correa, A. Cristi, A. Fielbaum, T. Pollner, and S. M. Weinberg, "Optimal item pricing in online combinatorial auctions," *Math. Program.*, vol. 206, no. 1, pp. 429–460, 2024.

[28] A. Singla and A. Krause, "Truthful incentives in crowdsourcing tasks using regret minimization mechanisms," in *Proc. 22nd Int. Conf. World Wide Web*, New York, NY, USA, 2013, pp. 1167–1178, doi: 10.1145/2488388.2488490.

[29] E. Balkanski and J. D. Hartline, “Bayesian budget feasibility with posted pricing,” in *Proc. 25th Int. Conf. World Wide Web*, 2016, pp. 189–203, doi: 10.1145/2872427.2883032.

[30] K. Han, H. Huang, and J. Luo, “Quality-aware pricing for mobile crowdsensing,” *IEEE/ACM Trans. Netw.*, vol. 26, no. 4, pp. 1728–1741, Aug. 2018.

[31] W. Xu et al., “Reward maximization for disaster zone monitoring with heterogeneous UAVs,” *IEEE/ACM Trans. Netw.*, vol. 32, no. 1, pp. 890–903, Feb. 2024.

[32] J. Zhang, Z. Wang, H. Wu, and W. Li, “Ordered submodularity-based value maximization of UAV data collection in earthquake areas,” *IEEE Trans. Netw. Sci. Eng.*, vol. 11, no. 5, pp. 4886–4897, Sep./Oct. 2024.

[33] J. Xu et al., “Semantic-aware UAV swarm coordination in the metaverse: A reputation-based incentive mechanism,” *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 13821–13833, Dec. 2024.

[34] H. Gao et al., “Dynamic task pricing in mobile crowdsensing: An age-of-information-based queueing game scheme,” *IEEE Internet Things J.*, vol. 9, no. 21, pp. 21278–21291, Nov. 2022.

[35] R. B. Kellogg, T. Y. Li, and J. Yorke, “A Method of Continuation for Calculating a Brouwer Fixed Point,” *Academic Press: Fixed Points*, 1977, pp. 133–147. [Online]. Available: https://dx.doi.org/10.1016/B978-0-12-398050-2.50012-8

[36] S. Nickel, C. Steinhardt, H. Schlenker, W. Burkart, and M. Reuter-Oppermann, *IBM ILOG CPLEX Optimization Studio*. Berlin, Germany: Springer, 2021, pp. 9–23, doi: 10.1007/978-3-662-62185-1_2.

Jixian Zhang received the MS and PhD degrees in computer science from the University of Electronic Science and Technology of China, in 2006 and 2010, respectively. Currently, he is an associate professor with the School of Computer Science and Engineering, Yunnan University. He has published more than 50 articles in peer-reviewed journals and conference proceedings, including *IEEE Transactions on Mobile Computing*, *IEEE Transactions on Network and Service Management*, *IEEE Transactions on Network Science and Engineering*, *Future Generation Computer Systems*, *Journal of Obstetrics and Gynaecology Canada*, *Cluster Computing* and *Cloud Computing*. His research interests include cloud computing, edge computing and mechanism design.

Xuelin Yang received the BS degree in computer science and technology from Xidian University, in 2023. She is currently working toward the master’s degree with Yunnan University and is expected to received the master’s degree, in 2026. Her main research interests include deep learning, mechanism design, and combinatorial optimization.

Peng Chen is currently working toward the master’s degree with Yunnan University and is expected to received the master’s degree, in 2025. His main study and research interests include deep learning, deep reinforcement learning, reverse auctions, mobile crowdsensing and semantic communication.

Zhemin Wang is currently working toward the master’s degree with Yunnan University and is expected to received the master’s degree, in 2024. His main areas of study include resource allocation, deep learning, and UAV communication.

Weidong Li received the PhD degree from the Department of Mathematics, Yunnan University, in 2010. He is currently a professor with the School of Mathematics and Statistics, Yunnan University. He has published more than 90 articles in peer-reviewed journals and conference proceedings, including *IEEE Transactions on Mobile Computing*, *IEEE Transactions on Parallel and Distributed Systems*, *ALGO*, *Journal of Arthroplasty*, *Theoretical Computer Science*, *JOCO*, *Future Generation Computer Systems*, and *Journal of Obstetrics and Gynaecology Canada*. His main research interests include combinatorial optimization, approximation algorithms, randomized algorithms and cloud computing.

Zhenli He (Senior Member, IEEE) received the BS degree in software engineering from Yunnan University, Kunming, China, in 2010, and the MS and PhD degrees in software engineering and systems analysis and integration from Yunnan University, in 2012 and 2015, respectively. From 2019 to 2021, he was a postdoctoral researcher with Hunan University, Changsha, China. He is currently an associate professor and head of the Department of Software Engineering, School of Software, Yunnan University. He was selected as a *Young Talent of the Yunnan Provincial “Xingdian Talent Support Program”* and is a recipient of the *Hongyun Gardener Award* for teaching excellence. His research interests include edge computing, energy-efficient computing, heterogeneous computing, and edge-AI techniques.

Keqin Li (Fellow, IEEE) received the BS degree in computer science from Tsinghua University, in 1985, and the PhD degree in computer science from the University of Houston, in 1990. He is a SUNY distinguished professor with the State University of New York and a National distinguished professor with Hunan University, China. He has authored or co-authored more than 1110 journal articles, book chapters, and refereed conference papers. He holds more than 75 patents announced or authorized by the Chinese National Intellectual Property Administration. Since 2020, he has been among the world’s top few most influential scientists in parallel and distributed computing regarding single-year impact (ranked #2) and career-long impact (ranked #4) based on a composite indicator of the Scopus citation database. He is listed in *Scilit Top Cited Scholars* (2023–2024) and is among the top 0.02% out of more than 20 million scholars worldwide based on top-cited publications. He is listed in *ScholarGPS Highly Ranked Scholars* (2022–2024) and is among the top 0.002% out of more than 30 million scholars worldwide based on a composite score of three ranking metrics for research productivity, impact, and quality in the recent five years. He received the IEEE TCCLD *Research Impact Award* from the IEEE CS Technical Committee on Cloud Computing, in 2022 and the IEEE TCSVC *Research Innovation Award* from the IEEE CS Technical Community on Services Computing, in 2023. He won the IEEE Region 1 *Technological Innovation Award* (Academic), in 2023. He was a recipient of the 2022–2023 *International Science and Technology Cooperation Award* and the 2023 *Xiaoxiang Friendship Award* of Hunan Province, China. He is a member of the SUNY Distinguished Academy. He is an AAAS fellow, an AAIA fellow, an ACIS fellow, and an AIIA fellow. He is a member of the European Academy of Sciences and Arts. He is a member of Academia Europaea (Academician of the Academy of Europe).