OIC: An Online Incentivized and Context-aware Scheme for Task Assignment

Jiamei Ji, Kechao Cai, Zhuoyue Chen, Jinbei Zhang
School of Electronics and Communication Engineering, Sun Yat-sen University, Shenzhen, China
{jijm3,chenzhy225}@mail2.sysu.edu.cn, {caikch3,zhjinbei}@mail.sysu.edu.cn

***Abstract*—Online task assignment plays a central role in applications such as crowdsourcing and digital advertising, where both context awareness and worker incentives are essential for ensuring quality and long-term participation. However, existing studies either ignore context-aware task-worker matching or overlook incentive design, failing to adapt when both factors interact under changing conditions. To bridge the gap, we propose an Online Incentivized and Context-aware (OIC) task assignment scheme that jointly addresses the interdependence between context awareness and worker incentives in dynamic environments. We first formulate the task assignment problem as a Stackelberg game and derive the optimal incentive allocation under an oracle setting. Then we develop an online learning algorithm within OIC that approximates the oracle solution to address the unknown and dynamic task-worker relationships. We provide theoretical guarantees by proving that our OIC scheme achieves sublinear regret. Extensive experiments demonstrate that OIC consistently outperforms baseline methods in terms of effective real-time task assignment under changing conditions and strategic worker behaviors.**

***Index Terms*—Task assignment, context aware, online learning, incentive mechanism**

I. Introduction

Online task assignment plays a vital role in many real-world applications [1]–[3]. For example, in crowdsourcing platforms such as Amazon Mechanical Turk, assigning tasks to workers with appropriate expertise in real time can significantly reduce completion time and improve output quality. In online advertising, dynamic ad placement based on ad types and advertiser characteristics in an online manner can increase click-through rates and revenue. Effective online task assignment not only boosts productivity but also helps maintain long-term worker engagement.

Related work. There are two critical challenges in online task assignment. The first is context-aware matching of task requirements with worker capabilities under non-stationary conditions, such as changing demand patterns and evolving task types. Many studies address this matching problem [4]–[8] using contextual bandit frameworks [9]–[11] to align task and worker feature vectors. For example, Liu *et al.* [6] consider contextual information including task type, base station location, and service provider, proposing a contextual sleeping bandit learning algorithm with sublinear regret. Elahi *et al.* [8] propose an adaptive optimistic matching algorithm that discretizes the joint task–worker context space and achieves sublinear regret. The second challenge is designing incentive mechanisms that ensure non-negative utility and preserve workers' self-interest. Various game-theoretic and auction-based methods [12]–[15] have been developed for efficient incentivization. These mechanisms typically fall into two modes: *worker-selected tasks* and *server-assigned tasks*, each with different goals and methods [16]. In the *worker-selected tasks* mode, tasks are published openly, and workers choose tasks based on individual preferences or utility [17]. In contrast, the *server-assigned tasks* mode features centralized assignment to optimize system-level objectives such as social welfare [18], fairness [19], or truthfulness [20]. For instance, Yang *et al.* [12] propose a server-centric crowdsensing model that accounts for selfish worker behavior when optimizing payments. Wang *et al.* [21] extend this to federated learning by incorporating data quality into the model.

However, these studies often overlook strategic worker behavior (e.g., [8], [22]) or assume static or fully known environments, limiting their adaptability in dynamic and uncertain settings (e.g., [12], [21]). Joint optimization of context-aware task-worker matching and incentive reward allocation under dynamic conditions remains largely unexplored.

In this paper, we propose an Online Incentivized and Context-aware (OIC) task assignment scheme that considers context-aware task–worker matching and multiple heterogeneous strategic workers. We model the one-to-many task assignment problem under the *server-assigned tasks* mode as a Stackelberg game. Due to the context-dependent and unknown service quality of workers, computing the exact game equilibrium is intractable. To address this, we design an online learning algorithm to approximate the equilibrium, introducing an exploration–exploitation tradeoff. We manage this tradeoff by establishing an upper confidence bound (UCB) on the server's utility and show that our method achieves strong long-term performance.

Our main contributions are as follows: (i) We propose a novel incentivized and context-aware task assignment scheme that accounts for dynamic task contexts and worker heterogeneity. (ii) We formulate the incentive mechanism design as a Stackelberg game and derive the optimal solution in an oracle setting with full knowledge of worker service quality. This solution guides the design of our online scheme and serves as a performance benchmark. (iii) Based on the oracle strategy, we develop an online algorithm and prove that it achieves sublinear regret. The algorithm addresses the estimation of

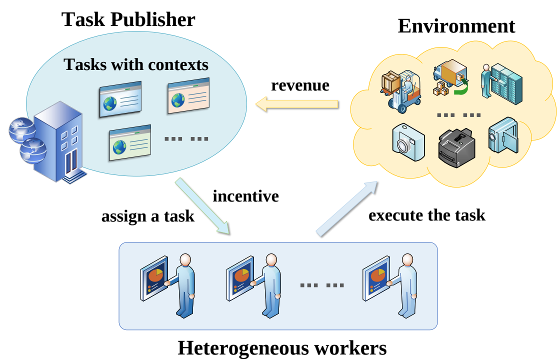


Fig. 1. Online incentivized and context-aware task assignment model.

unknown service quality, resource usage policy, optimal payment, and task assignment. (iv) We evaluate the proposed algorithm through extensive simulations and demonstrate its effectiveness in approximating the oracle benchmark while maximizing the server's long-term utility.

The remainder of this paper is organized as follows. Section II introduces our model and problem formulation. Section III presents our proposed scheme and its theoretical analysis. We evaluate the performance through numerical experiments in Section IV, and finally conclude the paper in Section V.

II. Model and Problem Formulation

In this section, we introduce our online task assignment model, define the utilities of both the server and clients, and formally present the objective of our online incentivized and context-aware task assignment scheme.

A. Online Task Assignment Model

Our online task assignment model, shown in Fig. 1, models a dynamic interaction between a task publisher (server) and multiple heterogeneous workers (clients). The server assigns a task to selected clients and offers corresponding incentives. Clients then execute the task, which involves interacting with the external environment.

Formally, the system consists of a server and a set of clients $\mathcal{N}$, where $|\mathcal{N}| = N$. At each round, the server selects one task from a set of K candidate tasks and assigns it to clients to maximize its own utility. At round t, each client $i \in \mathcal{N}$ incurs a unit cost c_i to execute a task, and provides service quality denoted as $q_{i,t}^k$ for task k, which could represent metrics such as click-through rate (CTR) or revenue in real world. This model operates in an online manner over a total of T rounds. At round $t \in [T]$, the following steps occur:

(1) The server observes a set of context vectors of the K candidate tasks: $\mathcal{A}_t = \{\boldsymbol{x}_t^1, \cdots, \boldsymbol{x}_t^k, \cdots, \boldsymbol{x}_t^K\} \subset \mathbb{R}^d$, where the d-dimensional vector $\boldsymbol{x}_t^k$ represents the context vector of task $k \in [K]$ at round t.

(2) The server selects a task k_t to be executed at round t, determines the total payment P_t for all clients, and assigns required resource usage $r_{i,t} \geq 0$ to client i based on client i's cost and service quality (defined later), for $i \in \mathcal{N}$. Client i would participate in the task if $r_{i,t} > 0$.

(3) Let $\mathcal{S}_t$ denote the set of participating clients. Client i executes the task k_t according to its required resource usage $r_{i,t}$ for $i \in \mathcal{S}_t$.

In real-world applications, the server does not have prior knowledge of each client's service quality. Hence, we assume that service quality of each client is an unknown but fixed linear function of the task context, with parameters that can be learned over time. Specifically, the actual service quality $q_{i,t}^k$ of client i for task k at round t is defined as:

$$q_{i,t}^k = \langle \boldsymbol{x}_t^k, \boldsymbol{\theta}_i \rangle + \eta_{i,t}^k, \tag{1}$$

where $\boldsymbol{\theta}_i \in \mathbb{R}^d$ is the unknown model parameter for client i and $\eta_{i,t}^k$ is a zero-mean σ-subgaussian noise term. Over time, client i can estimate its parameter $\boldsymbol{\theta}_i$ using historical data and thus compute an estimate $\hat{q}_{i,t}^k$. This estimate enables the server to assign tasks properly in the absence of complete information.

B. Utility Definitions

We define the utility functions of both the clients and the server as the metric used to design an effective incentive mechanism.

1) Client's Utility: The utility of a client is defined as the payment received from the server minus the incurred cost. When the server assigns task k to client i at round t, the client's utility is:

$$u_{i,t}^k = \frac{q_{i,t}^k r_{i,t}^k}{\sum_{j \in \mathcal{N}} q_{j,t}^k r_{j,t}^k} P_t^k - r_{i,t}^k c_i, \tag{2}$$

where P_t^k is the total payment to all clients if task k is selected at round t, and $r_{i,t}^k c_i$ is the total cost that client i requires to execute task k at round t.

2) Server's Utility: If the server requires clients to execute task k, its utility is defined as

$$u_{0,t}^k = \gamma \log(1 + \sum_{i \in \mathcal{N}} q_{i,t}^k r_{i,t}^k) - P_t^k, \tag{3}$$

where γ is a system parameter representing the ratio between virtual task execution performance and actual monetary payment. The logarithmic function captures diminishing returns with respect to the total contributed service quality.

C. Objective

Our online task assignment model is server-centric, aiming to maximize the server's utility while ensuring clients are incentivized to participate. Specifically, at round t, for each task $k \in [K]$, we compute the Nash Equilibrium (NE) of the clients' strategies and then determine the server's utility under that equilibrium. In particular, Nash equilibrium in our model is defined as follows.

Definition 1 (Nash Equilibrium (NE)). *A Nash equilibrium is a strategy profile* $\boldsymbol{r}_t^{k^{\mathrm{NE}}} = (r_{1,t}^{k^{\mathrm{NE}}}, r_{2,t}^{k^{\mathrm{NE}}}, ..., r_{N,t}^{k^{\mathrm{NE}}})$ *in which no* *client can increase its utility by unilaterally deviating from its strategy. Formally, for each client $i \in \mathcal{N}$ and any $r_{i,t}^k \geq 0$,*

$$u_{i,t}^k(r_{i,t}^{k^{\text{NE}}}, r_{-i,t}^{k^{\text{NE}}}) \geq u_{i,t}^k(r_{i,t}^k, r_{-i,t}^{k^{\text{NE}}}), \tag{4}$$

where $r_{-i,t}^{k^{\text{NE}}}$ denotes the equilibrium strategy profile of all clients except i.

Given the NE for each task, the server then selects the task that maximizes its utility. In the online learning literature, it is equivalent to the long-term objective of minimizing the cumulative regret, which is defined as:

$$R_T = \sum_{t=1}^{T} u_{0,t}^{k_t^*} - u_{0,t}^{k_t}, \tag{5}$$

where k_t^* is the optimal task at round t, and k_t is the task chosen by the server. Regret captures the performance gap between the oracle and the actual task selections across all rounds, so a lower regret implies better performance.

III. OIC: Online Incentivized and Context-aware Task Assignment Scheme

We formulate the task assignment problem as a Stackelberg game, where the server acts as the leader and the clients act as the followers. To derive the optimal solution to this task assignment problem, we decompose the scheme into the following two sub-problems:

P1: NE with known service quality. This sub-problem is for the oracle, where we assume that the server has perfect information about the client's service quality, i.e., $q_{i,t}^k$ is known for all clients and tasks. For a given task k and total payment P_t^k, the server needs to determine a unique strategy profile $\boldsymbol{r}_t^{k^{\text{NE}}}$ achieving NE to assign resource usages to clients. (Discussed in Section III-A)

P2: Online task selection with approximate NE. The server needs to obtain estimates of service quality via online learning. Using these estimates, the server computes an approximate NE strategy profile $\hat{\boldsymbol{r}}_t^{k^{\text{NE}}}$ for each task k to approximate the NE obtained in **P1**. Given the approximate NE strategy profiles, the server estimates the optimal payment $\hat{P}_t^k$ for each task $k \in [K]$ that maximizes its utility. The server then evaluates its estimated utility $\hat{u}_{0,t}^k$ for each task and selects the task with the highest estimated utility, i.e., $k_t = \arg\max_k \hat{u}_{0,t}^k$. (Discussed in Section III-B)

In the following paragraphs, we present detailed solutions to these sub-problems and provide a performance analysis of the proposed online incentive mechanism in Section III-C.

A. NE with Known Service Quality

We first derive the NE under the assumption that the system has perfect prior knowledge of service quality, referred to as the oracle setting. To compute the value of $r_{i,t}^k$ that satisfies the NE, we compute the derivative of $u_{i,t}^k$ with respect to $r_{i,t}^k$:

$$\frac{\partial u_{i,t}^k}{\partial r_{i,t}^k} = \frac{q_{i,t}^k \sum_{j\in\mathcal{N}\setminus\{i\}} q_{j,t}^k r_{j,t}^k}{\left(\sum_{j\in\mathcal{N}} q_{j,t}^k r_{j,t}^k\right)^2} P_t^k - c_i.$$

The second derivative of $u_{i,t}^k$ with respect to $r_{i,t}^k$ is:

$$\frac{\partial^2 u_{i,t}^k}{\partial (r_{i,t}^k)^2} = -\frac{2{q_{i,t}^k}^2 \Sigma_{j\in N\setminus\{i\}} q_{j,t}^k r_{j,t}^k}{(\Sigma_{j\in N} q_{j,t}^k r_{j,t}^k)^3} P_t^k < 0,$$

which implies that $u_{i,t}^k$ is concave with respect to $r_{i,t}^k$. Therefore, the optimal strategy of client i for task k corresponds to the solution of $\frac{\partial u_{i,t}^k}{\partial r_{i,t}^k} = 0$:

$$r_{i,t}^k = \frac{\sqrt{\frac{q_{i,t}^k P_t^k \sum_{j\in\mathcal{N}\setminus\{i\}} q_{j,t}^k r_{j,t}^k}{c_i}} - \sum_{j\in\mathcal{N}\setminus\{i\}} q_{j,t}^k r_{j,t}^k}{q_{i,t}^k}.$$

If the right-hand side of the above equation is negative, we set $r_{i,t}^k = 0$ to prevent negative resource usage. Therefore, the optimal strategy of client i for task k is:

$$\beta_{i,t}^k(r_{-i,t}^k) = \begin{cases} 0 & P_t^k \leq \frac{c_i}{q_{i,t}^k}\Sigma, \\ \frac{\sqrt{\frac{q_{i,t}^k P_t^k \Sigma}{c_i}} - \Sigma}{q_{i,t}^k} & \text{otherwise}, \end{cases} \tag{6}$$

where $\Sigma = \sum_{j\in\mathcal{N}\setminus\{i\}} q_{j,t}^k r_{j,t}^k$

However, the solution given in (6) is not explicit, as it contains terms $r_{j,t}^k$ for all clients $j \in \mathcal{N} \setminus \{i\}$ that remain unsolved. To facilitate the design of an algorithm for computing an NE, we establish that the NE strategy profile satisfies four statements in Theorem 1 as follows.

Theorem 1 (Properties of NE). *Assume that P_t^k is given. Let $\tilde{\boldsymbol{r}}_t^k = (\tilde{r}_{1,t}^k, \tilde{r}_{2,t}^k, ..., \tilde{r}_{N,t}^k)$ be the strategy profile of an NE for the Stackelberg game of task k, and let $\tilde{\mathcal{S}}_t^k$ denote the set of selected clients, i.e., $\tilde{\mathcal{S}}_t^k = \{i \in \mathcal{N} | \tilde{r}_{i,t}^k > 0\}$. The following four statements hold:*

(a) $|\tilde{\mathcal{S}}_t^k| \geq 2$.

(b) $\tilde{r}_{i,t}^k = \begin{cases} 0 & i \notin \tilde{\mathcal{S}}_t^k, \\ \frac{(|\tilde{\mathcal{S}}_t^k|-1)P_t^k}{q_{i,t}^k \sum_{j\in s_t^k} \frac{c_j}{q_{j,t}^k}}\left(1 - \frac{(|\tilde{\mathcal{S}}_t^k|-1)c_i}{q_{i,t}^k \sum_{j\in s_t^k} \frac{c_j}{q_{j,t}^k}}\right) & \textit{otherwise}. \end{cases}$

(c) If $\frac{c_i}{q_{i,t}^k} \leq \max_{j\in\tilde{\mathcal{S}}_t^k}\{\frac{c_j}{q_{j,t}^k}\}$, then $i \in \tilde{\mathcal{S}}_t^k$.

(d) Assume that the clients are ordered such that $\frac{c_{i(1)}}{q_{i(1),t}^k} < \frac{c_{i(2)}}{q_{i(2),t}^k} < \cdots < \frac{c_{i(N)}}{q_{i(N),t}^k}$. Let h be the largest integer in $[2, N]$ such that $\frac{c_{i(h)}}{q_{i(h),t}^k} < \frac{\sum_{j=1}^{h} \frac{c_{i(j)}}{q_{i(j),t}^k}}{h-1}$. Then $\tilde{\mathcal{S}}_t^k = \{i^{(1)}, i^{(2)}, ..., i^{(h)}\}$.

The proof of Theorem 1 is in Appendix A. Theorem 1 provides client selection criteria and an explicit form of the NE strategy profile. Statement (a) indicates that at least two clients are required to form an equilibrium. Statement (b) presents an explicit expression for the equilibrium. Statements (c) and (d) imply a preference for high-quality, low-cost clients in the equilibrium and confirm the uniqueness of the Stackelberg equilibrium. A unique strategy profile ensures that clients consistently adopt the same equilibrium behavior.

With the help of Theorem 1, we design Algorithm 1 to compute the NE for task k using Theorem 1. Algorithm 1 first sorts clients by their cost-to-quality ratio and then selects

Algorithm 1 Oracle: NE computation of task k

1: **Input:** P_t^k, $\{c_i, q_{i,t}^k\}_{i\in\mathcal{N}}$.
2: **Output:** $\boldsymbol{r}_t^{k^{\mathrm{NE}}} = (r_{1,t}^{k^{\mathrm{NE}}}, r_{2,t}^{k^{\mathrm{NE}}}, ..., r_{N,t}^{k^{\mathrm{NE}}})$
3: Sort clients according to their cost/quality ratio, $\frac{c_{i(1)}}{q_{i(1),t}^k} < \frac{c_{i(2)}}{q_{i(2),t}^k} < \cdots < \frac{c_{i(N)}}{q_{i(N),t}^k}$
4: $\mathcal{S}_t^k \leftarrow \{i^{(1)}, i^{(2)}\}, n \leftarrow 3, i \leftarrow i^{(n)}$
5: **while** $n \le N$ and $\frac{c_{i(n)}}{q_{i(n),t}^k} < \frac{\frac{c_{i(n)}}{q_{i(n),t}^k} + \sum_{j\in\mathcal{S}_t^k} \frac{c_j}{q_{j,t}^k}}{|\mathcal{S}_t^k|}$ **do**
6: $\quad\mathcal{S}_t^k \leftarrow \mathcal{S}_t^k \cup \{i^{(n)}\}, n \leftarrow n+1, i \leftarrow i^{(n)}$
7: **for all** $i \in \mathcal{N}$ **do**
8: $\quad$**if** $i \in \mathcal{S}_t$ **then**
9: $\quad\quad r_{i,t}^{k^{\mathrm{NE}}} \leftarrow \frac{(|\mathcal{S}_t^k|-1)P_t^k}{q_{i,t}^k \sum_{j\in\mathcal{S}_t^k} \frac{c_j}{q_{j,t}^k}} (1 - \frac{(|\mathcal{S}_t^k|-1)c_i}{q_{i,t}^k \sum_{j\in\mathcal{S}_t^k} \frac{c_j}{q_{j,t}^k}})$
10: $\quad$**else**
11: $\quad\quad r_{i,t}^{k^{\mathrm{NE}}} \leftarrow 0$

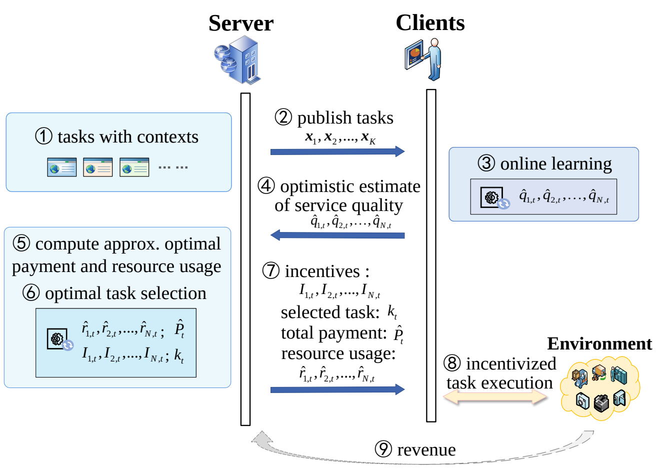


Fig. 2. Communication process between the server and clients in the online task assignment scheme.

clients with the lowest ratios until the condition in Line 5 is satisfied. We prove that the strategy profile $\boldsymbol{r}_t^{k^{\mathrm{NE}}}$ computed by Algorithm 1 is the NE for task k in Theorem 2 and provide the proof in Appendix B.

Theorem 2. *The strategy profile* $\boldsymbol{r}_t^{k^{\mathrm{NE}}}$ *computed by Algorithm 1 is the NE of the Stackelberg game of task* k*, which corresponds to the solution in* (6).

B. Online Task Selection with Approximate NE

Building upon the oracle solution provided in Algorithm 1, we propose our Online Incentivized and Context-aware (OIC) scheme for task assignment. The sequential communication between the server and clients in OIC is illustrated in Fig. 2. The detailed procedures of OIC are presented in Algorithms 2 and 3. Specifically, OIC consists of four core components, described as follows.

1) Optimistic estimate of service quality: To estimate service quality, we first estimate the model parameters $\boldsymbol{\theta}_i$ so that

Algorithm 2 OIC for task assignment

1: **[Clients]** Each client $i \in \mathcal{N}$ initializes $\boldsymbol{V}_{i,0} = \lambda \boldsymbol{I}$, $\boldsymbol{v}_{i,0} = \boldsymbol{0}^d$ and sends c_i to the server.
2: **for all** $t \in [T]$ **do**
$\quad$ */* 1) Optimistic estimate of service quality */*
3: $\quad$[Server] Observe task set $\mathcal{A}_t$ from the environment and publish tasks to clients.
4: $\quad$[Clients] Receive $\mathcal{A}_t$ from the server, compute optimistic estimate $\hat{q}_{i,t}^k$ according to (10), and send $\hat{q}_{i,t}^k$ to the server.
5: $\quad$[Server] Receive $\hat{q}_{i,t}^k$ from clients $i \in \mathcal{N}$.
$\quad$ */* 2) Approximate optimal payment & resource usage */*
6: $\quad$**for all** $k \in [K]$ **do**
7: $\quad\quad$[Server] Obtain $\hat{u}_{0,t}^k$, $\hat{P}_t^k$ and $\hat{\boldsymbol{r}}_t^{k^{\mathrm{NE}}} \leftarrow$ Algorithm 3.
$\quad$ */* 3) Optimal task selection */*
8: $\quad$[Server] Select the optimal task $k_t \leftarrow \arg\max_k \hat{u}_{0,t}^k$
9: $\quad$[Server] Assign task k_t to clients with resource usage $\hat{\boldsymbol{r}}_t \leftarrow \hat{\boldsymbol{r}}_t^{k_t^{\mathrm{NE}}}$ and payment $\hat{P}_t \leftarrow \hat{P}_t^{k_t}$.
$\quad$ */* 4) Incentivized task execution */*
10: $\quad$[Clients] Receive incentives $I_{i,t}$ according to (14), participate in the assigned task according to $\hat{\boldsymbol{r}}_t$ and observe the true service quality $q_{i,t}$.
11: $\quad$[Server] Get revenue from the environment.
12: $\quad$[Clients] Update $\boldsymbol{V}_{i,t}$ and $\boldsymbol{v}_{i,t}$ for each client $i \in \mathcal{N}$ according to (7) and (8).

the expected service quality can be computed according to the linear model defined in (1). To estimate the model $\boldsymbol{\theta}_i$, client i needs to maintains a covariance matrix $\boldsymbol{V}_{i,t}$ and a reward vector $\boldsymbol{v}_{i,t}$, which store the client's action and reward history respectively:

$$\boldsymbol{V}_{i,t} = \lambda \boldsymbol{I} + \sum_{\tau \le t, r_{i,\tau} > 0} \boldsymbol{x}_\tau \boldsymbol{x}_\tau^\top,$$
$$\boldsymbol{v}_{i,t} = \sum_{\tau \le t, r_{i,\tau} > 0} \boldsymbol{x}_\tau q_{i,\tau}.$$

At the beginning of the first round, we initialize $\boldsymbol{V}_{i,t} = \lambda \boldsymbol{I}$ and $\boldsymbol{v}_{i,t} = \boldsymbol{0}^d$, where λ the regularization parameter and $\boldsymbol{I}$ is the identity matrix (see Line 1 in Algorithm 2). Then the following two equations are used to iteratively update the covariance matrix and reward vector at each round t according to its assigned resource usage $\hat{r}_{i,t}$ (see Line 12 of Algorithm 2):

$$\boldsymbol{V}_{i,t} = \begin{cases} \boldsymbol{V}_{i,t-1} + \boldsymbol{x}_t \boldsymbol{x}_t^\top, & \hat{r}_{i,t} > 0, \\ \boldsymbol{V}_{i,t-1}, & \hat{r}_{i,t} = 0. \end{cases} \tag{7}$$

$$\boldsymbol{v}_{i,t} = \begin{cases} \boldsymbol{v}_{i,t-1} + \boldsymbol{x}_t q_{i,t}, & \hat{r}_{i,t} > 0, \\ \boldsymbol{v}_{i,t-1}, & \hat{r}_{i,t} = 0. \end{cases} \tag{8}$$

Using the historical data, we estimate $\boldsymbol{\theta}_i$ via ridge regression:

$$\hat{\boldsymbol{\theta}}_{i,t} = \boldsymbol{V}_{i,t-1}^{-1} \boldsymbol{v}_{i,t-1}. \tag{9}$$

Algorithm 3 Approximate optimal payment & resource usage of task k

1: **Input:** $\{c_i, \hat{q}_{i,t}^k\}_{i\in\mathcal{N}}$.
2: **Output:** $\hat{u}_{0,t}^k$, $\hat{P}_t^k$ and $\hat{\boldsymbol{r}}_t^{k^{\mathrm{NE}}} = (\hat{r}_{1,t}^{k^{\mathrm{NE}}}, \hat{r}_{2,t}^{k^{\mathrm{NE}}}, ..., \hat{r}_{N,t}^{k^{\mathrm{NE}}})$
3: Sort clients according to their cost/quality ratio, $\frac{c_{i^{(1)}}}{\hat{q}_{i^{(1)},t}^k} < \frac{c_{i^{(2)}}}{\hat{q}_{i^{(2)},t}^k} < \cdots < \frac{c_{i^{(N)}}}{\hat{q}_{i^{(N)},t}^k}$
4: $\hat{\mathcal{S}}_t^k \leftarrow \{i^{(1)}, i^{(2)}\}, n \leftarrow 3, i \leftarrow i^{(n)}$
5: **while** $n \le N$ and $\frac{c_{i^{(n)}}}{\hat{q}_{i^{(n)},t}^k} < \frac{\frac{c_{i^{(n)}}}{\hat{q}_{i^{(n)},t}^k} + \sum_{j\in\hat{\mathcal{S}}_t^k} \frac{c_j}{\hat{q}_{j,t}^k}}{|\hat{\mathcal{S}}_t^k|}$ **do**
6: $\quad \hat{\mathcal{S}}_t^k \leftarrow \hat{\mathcal{S}}_t^k \cup \{i^{(n)}\}, n \leftarrow n+1, i \leftarrow i^{(n)}$
7: Compute $\hat{X}_t^k \leftarrow \frac{(|\hat{\mathcal{S}}_t^k|-1)}{\sum_{j\in\hat{\mathcal{S}}_t^k} \frac{c_i}{\hat{q}_{i,t}^k}}$, $\hat{P}_t^k \leftarrow \gamma - \frac{1}{\hat{X}_t^k}$ and $\hat{u}_{0,t}^k \leftarrow \gamma\log(\gamma\hat{X}_t^k) + \frac{1}{\hat{X}_t^k} - \gamma$
8: **for all** $i \in \mathcal{N}$ **do**
9: $\quad$ **if** $i \in \hat{\mathcal{S}}_t$ **then**
10: $\quad\quad \hat{r}_{i,t}^{k^{\mathrm{NE}}} \leftarrow \frac{(|\hat{\mathcal{S}}_t^k|-1)\hat{P}_t^k}{\hat{q}_{i,t}^k \sum_{j\in\hat{\mathcal{S}}_t^k} \frac{c_j}{q_{j,t}^k}} (1 - \frac{(|\hat{\mathcal{S}}_t^k|-1)c_i}{\hat{q}_{i,t}^k \sum_{j\in\hat{\mathcal{S}}_t^k} \frac{c_j}{\hat{q}_{j,t}^k}})$
11: $\quad$ **else**
12: $\quad\quad \hat{r}_{i,t}^{k^{\mathrm{NE}}} \leftarrow 0$

With the estimated model $\hat{\boldsymbol{\theta}}_{i,t}$, client i can estimate its expected service quality as $\langle \boldsymbol{x}_t^k, \hat{\boldsymbol{\theta}}_{i,t}\rangle$.

However, directly using the expected service quality for client selection may exclude clients with high actual quality but limited exploration. In such cases, the variance of the estimate may be large, resulting in an underestimated service quality. To balance exploration and exploitation, we use an optimistic estimate of service quality by computing an upper confidence bound (UCB) (derived in [9], [10]) of the service quality (see Line 4 in Algorithm 2):

$$\hat{q}_{i,t}^k = \langle \boldsymbol{x}_t^k, \hat{\boldsymbol{\theta}}_{i,t}\rangle + \alpha_{i,t}\|\boldsymbol{x}_t^k\|_{\boldsymbol{V}_{i,t-1}^{-1}}, \tag{10}$$

where $\alpha_{i,t} = \sigma\sqrt{\log\frac{|\boldsymbol{V}_{i,t-1}|}{|\lambda\boldsymbol{I}|} + 2\log\frac{2}{\delta}} + \sqrt{\lambda}$ and δ denotes the confidence level. Such a UCB estimation provides a confidence interval of the actual service quality with probability at least $1-\delta$. Using optimistic estimates, our scheme naturally prioritizes clients with high expected quality (exploitation) or high uncertainty (exploration).

2) Approximate optimal payment and resource usage: After obtaining optimistic estimates of service quality, our goal is to compute approximate optimal resource usages of clients for each task k and compute the approximate optimal payment to maximize the server's utility. Specifically, we aim to determine the undetermined variables $\hat{\boldsymbol{r}}_t^{k^{\mathrm{NE}}}$ and $\hat{P}_t^k$. To extend the NE in the oracle setting to the online setting, we incorporate the optimistic estimate of service quality derived in (10). Replacing $q_{i,t}^k$ in Algorithm 1 with the optimistic counterpart $\hat{q}_{i,t}^k$, we obtain Algorithm 3, which computes an approximate NE strategy profile $\hat{\boldsymbol{r}}_t^{k^{\mathrm{NE}}}$ (see Line 10 in Algorithm 3).

Once $\hat{\boldsymbol{r}}_t^{k^{\mathrm{NE}}}$ is determined, we proceed to calculate the optimal payment $\hat{P}_t^k$ that maximizes the utility of the server. Applying the value of $\hat{\boldsymbol{r}}_t^{k^{\mathrm{NE}}}$ to the server's utility function, the server's approximate utility $\hat{u}_{0,t}^k$ is as follows:

$$\hat{u}_{0,t}^k = \gamma\log(1 + \frac{(|\hat{\mathcal{S}}_t^k|-1)\hat{P}_t^k}{\sum_{j\in\hat{\mathcal{S}}_t^k}\frac{c_i}{\hat{q}_{i,t}^k}}) - \hat{P}_t^k.$$

Let $\hat{X}_t^k = \frac{(|\hat{\mathcal{S}}_t^k|-1)}{\sum_{j\in\hat{\mathcal{S}}_t^k}\frac{c_i}{\hat{q}_{i,t}^k}}$, the server's utility can be rewritten as:

$$\hat{u}_{0,t}^k = \gamma\log(1 + \hat{X}_t^k\hat{P}_t^k) - \hat{P}_t^k.$$

We can analyze the server's utility $\hat{u}_{0,t}^k$ as a function of $\hat{P}_t^k$. The first derivative of $\hat{u}_{0,t}^k$ with respect to $\hat{P}_t^k$ is:

$$\frac{\partial\hat{u}_{0,t}^k}{\partial\hat{P}_t^k} = \frac{\gamma}{1+\hat{X}_t^k\hat{P}_t^k} - 1. \tag{11}$$

The second derivative of $\hat{u}_{0,t}^k$ with respect to $\hat{P}_t^k$ is:

$$\frac{\partial^2\hat{u}_{0,t}^k}{\partial(\hat{P}_t^k)^2} = -\frac{\gamma\hat{X}_t^k}{(1+\hat{X}_t^k\hat{P}_t^k)^2} < 0.$$

Hence, $\hat{u}_{0,t}^k$ is a concave function of $\hat{P}_t^k$. We set (11) to zero and obtain the optimal $\hat{P}_t^k$:

$$\hat{P}_t^k = \gamma - \frac{1}{\hat{X}_t^k}. \tag{12}$$

Applying (12) to compute the approximate server's utility:

$$\hat{u}_{0,t}^k = \gamma\log(\gamma\hat{X}_t^k) + \frac{1}{\hat{X}_t^k} - \gamma. \tag{13}$$

We apply (12) and (13) in Line 7 of Algorithm 3 to compute the approximate optimal payment and approximate server utility for each task.

3) Optimal task selection: After computing the approximate optimal resource usage and payment for each task k, we select the optimal task k_t with the highest server's utility to be assigned to clients. To balance the tradeoff between exploration and exploitation, we select the task with the highest UCB of the server's utility. Specifically, we prove that $\hat{u}_{0,t}^k$ computed by (13) serves as a valid UCB of the actual utility $u_{0,t}^k$, as shown in Theorem 3. Therefore, we select the task with the maximum $\hat{u}_{0,t}^k$ as the optimal task and assign it to clients (see Line 8 in Algorithm 2).

Theorem 3 (UCB of Server's Utility)**.** *If $\gamma \ge (1+\frac{1}{N-1})\frac{c_{\max}}{q_{\min}}$ holds, where $q_{\min}$ is the minimum value of data quality and $c_{\max}$ is the maximum value of cost, then $\hat{u}_{0,t}^k$ computed by (13) is a valid UCB of $u_{0,t}^k$ with probability at least $(1-\delta)^N$.*

Proof sketch: First, we prove that $\hat{X}_t^k$ is an upper confidence bound of $X_t^k = \frac{(|\mathcal{S}_t^k|-1)}{\sum_{j\in\mathcal{S}_t^k}\frac{c_i}{q_{i,t}^k}}$, where $\mathcal{S}_t^k$ is the set of selected clients in the oracle setting. Then, under the condition of sufficiently large γ, we reduce the problem of bounding $\hat{u}_{0,t}^k$ to that of bounding $\hat{X}_t^k$ being a UCB of X_t^k, thus completing the proof. The full proof is provided in Appendix C.

Once the optimal task k_t is selected, the server assigns the task k_t to the clients with resource usage $\hat{\boldsymbol{r}}_t = \hat{\boldsymbol{r}}_{i,t}^{k_t^{\mathrm{NE}}}$ and payment $\hat{P}_t = \hat{P}_t^{k_t}$.

4) Incentivized Task Execution: After server assigns task k_t, clients execute the task with corresponding incentives. Each client receives incentives:

$$I_{i,t} = \frac{\hat{q}_{i,t}^{k_t} \hat{r}_{i,t}^{k_t}}{\sum_{j \in \mathcal{N}} \hat{q}_{j,t}^{k_t} \hat{r}_{j,t}^{k_t}} \hat{P}_t^{k_t}. \tag{14}$$

Then each client participates in the task according to $\hat{\boldsymbol{r}}_t$, and observes their actual service quality $q_{i,t}$. After clients execute the task, the server receives revenue from the environment based on the client's service quality. The incentivized task execution procedure is shown in Lines 10-11 of Algorithm 2.

C. Performance Analysis

We analyze the performance of OIC and establish that its regret grows sublinearly with the number of rounds T, indicating that the scheme becomes increasingly effective over time. This result is formalized in Theorem 4.

Theorem 4 (Sub-linear Regret Upper Bound)**.** *Suppose that* $||\boldsymbol{x}_t||_2 \leq L$ *and* $\gamma \geq (1 + \frac{1}{N-1}) \frac{c_{\max}}{q_{\min}}$*. Then, the regret of OIC is upper bounded as follows with probability at least* $(1-\delta)^N$*:*

$$R_T \leq O(\frac{\gamma N \sigma d}{q_{\min}} \sqrt{T} \log(TL^2/\lambda d)). \tag{15}$$

Proof Sketch: Our proof can be decomposed into two steps. First, we bound the single-round regret. Specifically, we show that the regret in the server's utility in a given round can be bounded by the upper confidence bound of the clients' service quality estimates in that round. This result establishes a crucial connection between the server's performance and the quality of client-side task assignments. Second, we bound the cumulative regret of the server's utility. This is done by leveraging the confidence intervals on service quality to construct a martingale sequence, which enables us to show that the sum of the confidence intervals over time converges. Combining these two steps, we derive the overall regret bound. The complete proof is provided in Appendix D.

IV. Experiments

In this section, we conduct experiments using both synthetic and real-world datasets to evaluate the performance of OIC, aiming to: validate the correctness of the NE computation in the oracle setting; demonstrate convergence of the approximate NE in OIC; and evaluate the long-term performance and compare it with two baseline methods. We first illustrate our experiment settings and then demonstrate our experiment results.

A. Experiment Settings

We conduct experiments on two datasets: a synthetic dataset and a real-world dataset (Yahoo "Learning to Rank Challenge Dataset" [23]). We set the parameters as $T = 10^3$, $K = 10$, $N = 20$, $\gamma = 5$, $d = 9$, $L = 9$, $\sigma = 1$ and $\delta = 0.05$.

Synthetic dataset: The synthetic dataset is generated based on the linear model of service quality. Each task context $\boldsymbol{x}_t^k$ is sampled independently from a uniform distribution over

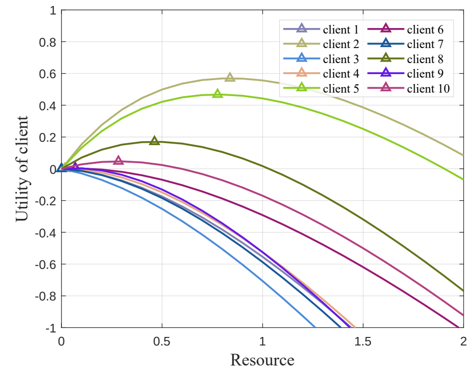


Fig. 3. Verification of NE computation.

$[0, 1]^d$. Each client's model parameter $\boldsymbol{\theta}_i$ is also sampled uniformly from $[0, 1]^d$. The true service quality is calculated as the inner product between the task context and the corresponding client's model. The cost is sampled uniformly from $[0.1, 1.1]$.

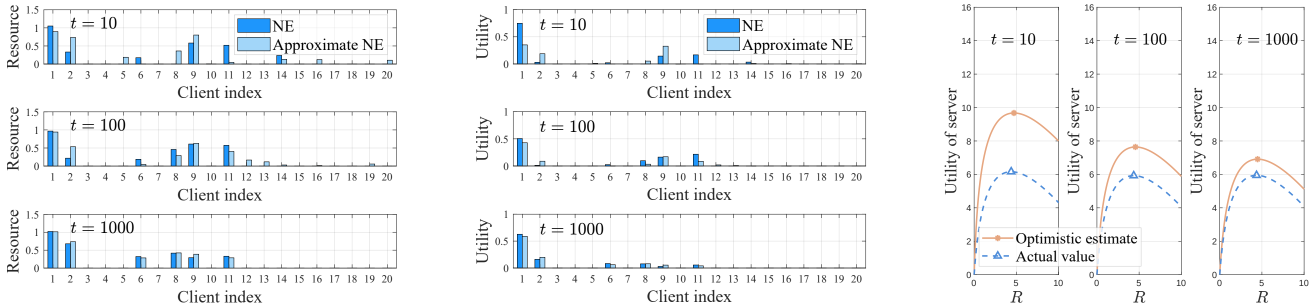
Yahoo dataset: We use "set2.test.txt" in the Yahoo dataset in our experiments, which provides ratings and normalized feature values. We select 9 feature columns with the highest number of valid entries. We cluster the features into 500 groups using K-means to simulate the model heterogeneity among different clients, and then randomly select N groups of features to construct N model parameters. For client $i \in [N]$, we compute its model parameter $\boldsymbol{\theta}_i$ by taking the ratio of the total relevance rating to the total feature sum in the group. The remaining feature groups are used to form task contexts over T rounds. Each client's actual service quality in a given round is computed as the inner product between their model parameter and the current task context. As the dataset does not contain data about costs, so we sample costs uniformly from $[0.1, 1.1]$.

B. Verification of NE Computation of Oracle

To verify the NE computation in Algorithm 1, we compute each client's utility as a function of its resource usage while fixing the resource usages of all other clients to the equilibrium values computed by Algorithm 1. The result is shown in Fig. 3, and the equilibrium points identified by Algorithm 1 are marked with triangles. The utility curve for each client exhibits a concave shape, and the utility reaches its maximum precisely at the point corresponding to the resource usage in Algorithm 1. When a client deviates, either increasing or decreasing its resource usage, its utility decreases. This behavior confirms that no client has motivation to unilaterally deviate from its equilibrium strategy, satisfying the definition of NE. Therefore, OIC effectively rationalizes clients' participation in the assigned task in alignment with the server's expectations.

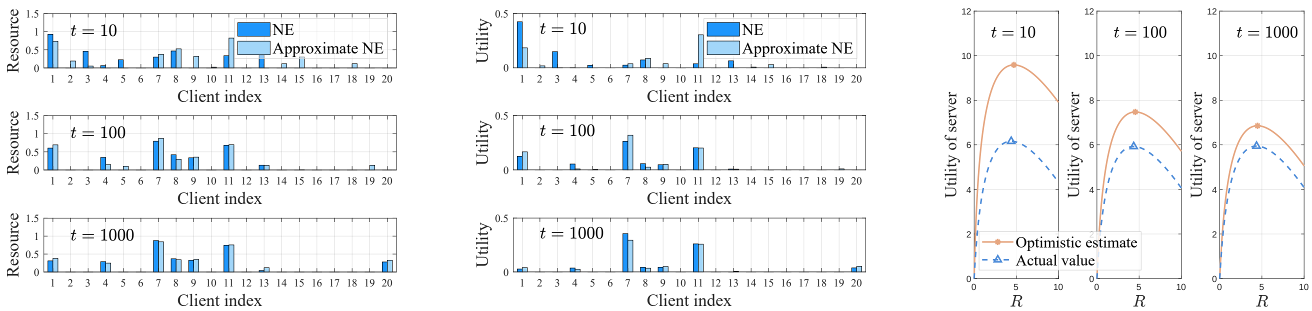
C. Convergence of Approximate NE in OIC

We compare the approximate NE computed via Algorithm 3 (OIC) with the exact NE computed via Algorithm 1 (oracle). We conduct this comparison across the following three metrics: (1) resource usage of each client; (2) utility of each client;



(a) Resource usage (b) Client's utility (c) Server's utility

Fig. 4. Gap between the approximate NE in OIC and the exact NE in oracle (synthetic dataset).



(a) Resource usage (b) Client's utility (c) Server's utility

Fig. 5. Gap between the approximate NE in OIC and the exact NE in oracle (Yahoo dataset).

(3) server utility (exact in the oracle; optimistic estimate in OIC). The closer the results given by OIC are to those given by the oracle, the better our online scheme converges.

As shown in Fig. 4 and Fig. 5, the three metrics in the online scheme all become increasingly aligned with those of the oracle. This validates the efficacy of the online scheme in approximating the optimal solution in the oracle setting over time. Additionally, the optimistic estimates of the server utility consistently exceed the actual utility values, thereby confirming the validity of Theorem 3.

D. Long-term Performance

We evaluate the long-term performance of OIC by analyzing its regret. According to Theorem 4, the regret upper bound depends on the number of clients N and the parameter γ. Hence, we analyze how varying N and γ affects regret, and the results are shown in Fig. 6. As shown in Fig. 6(a) and Fig. 6(c), the regret increases with N, but not linearly. This is because the theoretical regret bound is derived by relaxing the actual number of selected clients and the number of times each client is selected. Due to the dynamic nature of service quality, the number of selected clients per round varies and is difficult to bound precisely. Thus, we conservatively bound the regret using the total number of clients N and total rounds T, which results in a loose dependency on N in the regret bound. As shown in Fig. 6(b) and Fig. 6(d), regret increases linearly with γ, aligning with Theorem 4. Since γ represents the ratio between virtual task execution performance and actual monetary payment, a higher γ amplifies the regret bound, reflecting increased sensitivity to errors in task performance estimation.

To further evaluate the robustness of our approach, we compare the proposed online scheme against two baselines:

- OIC without UCB (OIC w/o UCB): This variant uses ridge regression for service quality estimation but does not incorporate exploration, limiting its ability to identify high-performing clients.
- ϵ-greedy: This strategy explores with probability ϵ by randomly assigning tasks to all clients, and exploits the estimated best task otherwise. We set $\epsilon = \min\{1, \frac{K}{t}\}$ following [24] to ensure sublinear regret.

As shown in Fig. 7, OIC outperforms both baselines. OIC w/o UCB performs the worst due to the absence of exploration, which leads to the exclusion of potentially high-quality clients based on initial poor performance. The ϵ-greedy approach performs better by incorporating exploration, but its naive strategy incurs linear regret increase during some exploratory rounds. In contrast, OIC achieves a more efficient exploration-exploitation tradeoff, resulting in consistently lower regret.

V. Conclusion

In this paper, we proposed OIC, a novel online incentivized and context-aware task assignment scheme designed for environments with unknown, context-dependent service quality. Assuming a linear relationship between service quality and dynamic task contexts, we formulated the task assignment

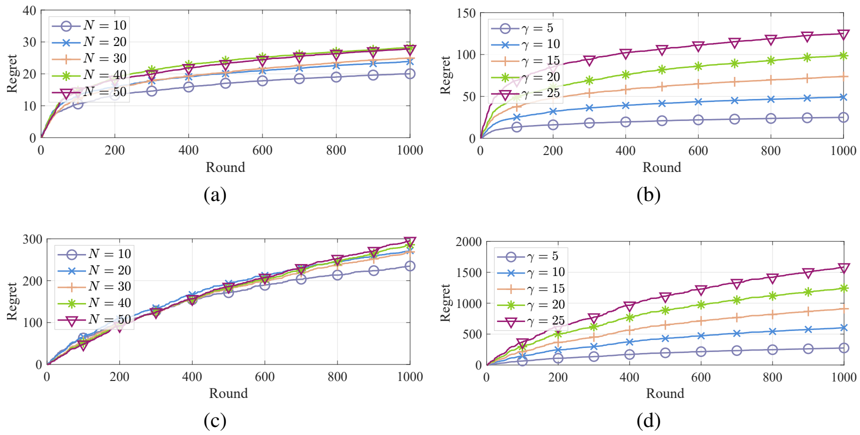


Fig. 6. Regret performance. Experiments using synthetic dataset: (a) regret versus N, (b) regret versus γ; experiments using Yahoo dataset: (c) regret versus N, (d) regret versus γ.

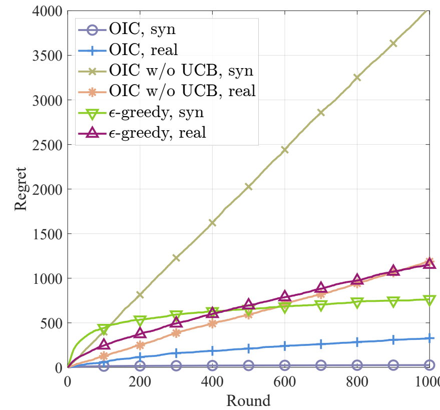


Fig. 7. Regret comparison.

problem as a Stackelberg game and applied online learning to approximate the optimal task assignment scheme. By deriving an upper confidence bound on the server's utility, we managed the exploration–exploitation tradeoff and established a sublinear regret bound, ensuring strong long-term performance. Extensive experiments on both synthetic and real-world datasets validated the effectiveness of OIC in incentivizing client participation and maximizing the server's expected utility over time.

References

[1] R. J. L. Dawn and A. Giovanidis, "Social influencer selection by budgeted portfolio optimization," in *2021 19th International Symposium on Modeling and Optimization in Mobile, Ad hoc, and Wireless Networks (WiOpt)*. IEEE, 2021, pp. 1–8.

[2] X. Wang, J. Ye, and J. C. Lui, "Decentralized task offloading in edge computing: A multi-user multi-armed bandit approach," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*. IEEE, 2022, pp. 1199–1208.

[3] B. Simon, A. Ortiz, W. Saad, and A. Klein, "Decentralized online learning in task assignment games for mobile crowdsensing," *IEEE Transactions on Communications*, 2024.

[4] S. K. née Müller, C. Tekin, M. van der Schaar, and A. Klein, "Context-aware hierarchical online learning for performance maximization in mobile crowdsourcing," *IEEE/ACM Transactions on Networking*, vol. 26, no. 3, pp. 1334–1347, 2018.

[5] Y. Wu, F. Li, L. Ma, Y. Xie, T. Li, and Y. Wang, "A context-aware multiarmed bandit incentive mechanism for mobile crowd sensing systems," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 7648–7658, 2019.

[6] S. Liu, P. Cheng, Z. Chen, W. Xiang, B. Vucetic, and Y. Li, "Contextual user-centric task offloading for mobile edge computing in ultra-dense network," *IEEE Transactions on Mobile Computing*, vol. 22, no. 9, pp. 5092–5108, 2022.

[7] R. Zhang, P. Cheng, Z. Chen, S. Liu, B. Vucetic, and Y. Li, "Calibrated bandit learning for decentralized task offloading in ultra-dense networks," *IEEE Transactions on Communications*, vol. 70, no. 4, pp. 2547–2560, 2022.

[8] S. Elahi, A. Nika, and C. Tekin, "Online context-aware task assignment in mobile crowdsourcing via adaptive discretization," *IEEE Transactions on Network Science and Engineering*, vol. 10, no. 1, pp. 305–320, 2022.

[9] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proceedings of the 19th international conference on World wide web*, 2010, pp. 661–670.

[10] Y. Abbasi-Yadkori, D. Pál, and C. Szepesvári, "Improved algorithms for linear stochastic bandits," *Advances in neural information processing systems*, vol. 24, 2011.

[11] S. Agrawal and N. Goyal, "Thompson sampling for contextual bandits with linear payoffs," in *International conference on machine learning*. PMLR, 2013, pp. 127–135.

[12] D. Yang, G. Xue, X. Fang, and J. Tang, "Incentive mechanisms for crowdsensing: Crowdsourcing with smartphones," *IEEE/ACM transactions on networking*, vol. 24, no. 3, pp. 1732–1744, 2015.

[13] Z. Cai, Z. Duan, and W. Li, "Exploiting multi-dimensional task diversity in distributed auctions for mobile crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 20, no. 8, pp. 2576–2591, 2020.

[14] Y. Li, F. Li, S. Yang, P. Zhou, L. Zhu, and Y. Wang, "Three-stage stackelberg long-term incentive mechanism and monetization for mobile crowdsensing: An online learning approach," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 1385–1398, 2021.

[15] Q. Xu, Z. Su, D. Fang, and Y. Wu, "BASIC: Distributed task assignment with auction incentive in uav-enabled crowdsensing system," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 2, pp. 2416–2430, 2023.

[16] B. Zhao, Y. Wang, Y. Li, Y. Gao, and X. Tong, "Task allocation model based on worker friend relationship for mobile crowdsourcing," *Sensors*, vol. 19, no. 4, p. 921, 2019.

[17] M. H. Cheung, F. Hou, J. Huang, and R. Southwell, "Distributed time-sensitive task selection in mobile crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 20, no. 6, pp. 2172–2185, 2020.

[18] Z. Wang, L. Gao, and J. Huang, "Socially-optimal mechanism design for incentivized online learning," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*. IEEE, 2022, pp. 1828–1837.

[19] Y. Zhao, K. Zheng, J. Guo, B. Yang, T. B. Pedersen, and C. S. Jensen, "Fairness-aware task assignment in spatial crowdsourcing: Game-theoretic approaches," in *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE, 2021, pp. 265–276.

[20] E. Wang, H. Wang, Y. Yang, and W. Liu, "Truthful incentive mechanism for budget-constrained online user selection in mobile crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 21, no. 12, pp. 4642–4655, 2021.

[21] X. Wang, Y. Zhao, C. Qiu, Z. Liu, J. Nie, and V. C. Leung, "Infedge: A blockchain-based incentive mechanism in hierarchical federated learning for end-edge-cloud communications," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 12, pp. 3325–3342, 2022.

[22] H. Zhang, Y. Ma, and M. Sugiyama, "Bandit-based task assignment for heterogeneous crowdsourcing," *Neural computation*, vol. 27, no. 11, pp. 2447–2475, 2015.

[23] O. Chapelle and Y. Chang, "Yahoo! learning to rank challenge overview," in *Proceedings of the learning to rank challenge*. PMLR, 2011, pp. 1–24.

[24] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine learning*, vol. 47, pp. 235–256, 2002.

Appendix A
Proofs of Theorem 1

Proof. **Proof of (1)**: We prove this statement by contradiction. Suppose that $|\tilde{\mathcal{S}}_t^k| = 0$, then client i can increase its utility from 0 to $\frac{P_t^k}{2}$ by unilaterally changing its computational resource from 0 to $\frac{P_t^k}{2c_i}$, contradicting the NE assumption. This proves that $|\tilde{\mathcal{S}}_t^k| \geq 1$. Suppose that $|\tilde{\mathcal{S}}_t^k| = 1$. This means $\tilde{r}_{i,t}^k > 0$ for some $i \in \mathcal{N}$, and $\tilde{r}_{j,t}^k = 0$ for all $j \in \mathcal{N} \setminus \{i\}$. The current utility of client i is $P_t^k - \tilde{r}_{i,t}^k c_i$. Client i can increase its utility by unilaterally changing its sensing time from $\tilde{r}_{i,t}^k$ to $\frac{\tilde{r}_{i,t}^k}{2}$, again contradicting the NE assumption. Therefore $|\tilde{\mathcal{S}}_t^k| \geq 2$.

Proof of (2): The optimal strategy can be obtained by letting the derivative of the utility of client be 0, i.e.,

$$\begin{aligned}\frac{\partial u_{i,t}^k}{\partial r_{i,t}^k} &= \frac{q_{i,t}^k}{\sum_{j\in\mathcal{N}} q_{j,t}^k r_{j,t}^k} P_t^k - \frac{q_{i,t}^k}{(\sum_{j\in\mathcal{N}} q_{j,t}^k r_{j,t}^k)^2} P_t^k - c_i \\ &= \frac{q_{i,t}^k}{\sum_{j\in\tilde{\mathcal{S}}_t^k} q_{j,t}^k r_{j,t}^k} P_t^k - \frac{q_{i,t}^k}{(\sum_{j\in\tilde{\mathcal{S}}_t^k} q_{j,t}^k r_{j,t}^k)^2} P_t^k - c_i = 0.\end{aligned} \tag{16}$$

Summing up over the clients in $\tilde{\mathcal{S}}_t^k$, we have

$$n_0 P_t^k - P_t^k = \sum_{j\in\tilde{\mathcal{S}}_t^k} q_{j,t}^k r_{j,t}^k \sum_{j\in\tilde{\mathcal{S}}_t^k} \frac{c_j}{q_{j,t}^k}, \tag{17}$$

thus we have

$$\sum_{j\in\tilde{\mathcal{S}}_t^k} q_{j,t}^k r_{j,t}^k = \frac{(n_0-1)P_t^k}{\sum_{j\in\tilde{\mathcal{S}}_t^k} \frac{c_j}{q_{j,t}^k}}. \tag{18}$$

Substituting (18) into (16) and considering $\tilde{r}_{i,t}^k = 0$ for any $j \in \mathcal{N} \setminus \{\tilde{\mathcal{S}}_t^k\}$, we obtain the optimal strategy:

$$\tilde{r}_{i,t}^k = \frac{(n_0-1)P_t^k}{q_{i,t}^k \sum_{j\in\tilde{\mathcal{S}}_t^k} \frac{c_j}{q_{j,t}^k}} \left(1 - \frac{(n_0-1)c_i}{q_{i,t}^k \sum_{j\in\tilde{\mathcal{S}}_t^k} \frac{c_j}{q_{j,t}^k}}\right). \tag{19}$$

Therefore, the statement (2) is proved.

Proof of (3): We prove this statement by contradiction. By definition of $\tilde{\mathcal{S}}_t^k$, we know that $\tilde{r}_{i,t}^k > 0$ for every $i \in \tilde{\mathcal{S}}_t^k$. From statement (2), $\tilde{r}_{i,t}^k > 0$ implies $\frac{(n_0-1)c_i}{q_{i,t}^k \sum_{j\in\tilde{\mathcal{S}}_t^k} q_{j,t}^k} < 1$, i.e., $\forall i \in \tilde{\mathcal{S}}_t^k$, $\frac{c_i}{q_{i,t}^k} < \frac{\sum_{j\in\tilde{\mathcal{S}}_t^k} \frac{c_j}{q_{j,t}^k}}{n_0-1}$. Therefore we have $\max_{j\in\tilde{\mathcal{S}}_t^k}\{\frac{c_j}{q_{j,t}^k}\} < \frac{\sum_{j\in\tilde{\mathcal{S}}_t^k} \frac{c_j}{q_{j,t}^k}}{n_0-1}$. Assume that $\frac{c_i}{q_{i,t}^k} \leq \max_{j\in\tilde{\mathcal{S}}_t^k}\{\frac{c_j}{q_{j,t}^k}\} < \frac{\sum_{j\in\tilde{\mathcal{S}}_t^k} \frac{c_j}{q_{j,t}^k}}{n_0-1}$ but $i \notin \tilde{\mathcal{S}}_t^k$, i.e., $\tilde{r}_{i,t}^k = 0$.

$$\begin{aligned}\frac{\partial u_{i,t}^k}{\partial r_{i,t}^k} &= \frac{q_{i,t}^k \sum_{j\in N\setminus\{i\}} q_{j,t}^k r_{j,t}^k}{(\sum_{j\in N} q_{j,t}^k r_{j,t}^k)^2} P_t^k - c_i \\ &= \frac{q_{i,t}^k \sum_{j\in\tilde{\mathcal{S}}_t^k} q_{j,t}^k r_{j,t}^k}{(\sum_{j\in\tilde{\mathcal{S}}_t^k} q_{j,t}^k r_{j,t}^k)^2} P_t^k - c_i \\ &= \frac{q_{i,t}^k}{\sum_{j\in\tilde{\mathcal{S}}_t^k} q_{j,t}^k r_{j,t}^k} P_t^k - c_i \\ &= \frac{q_{i,t}^k \sum_{j\in\tilde{\mathcal{S}}_t^k} \frac{c_j}{q_{j,t}^k}}{n_0-1} - c_i \\ &> q_{i,t}^k \max_{j\in\tilde{\mathcal{S}}_t^k}\{\frac{c_j}{q_{j,t}^k}\} - c_i \geq 0,\end{aligned}$$

where the third equation comes from (18). This means that client i can increase its utility by unilaterally increasing its sensing time from $\tilde{r}_{i,t}^k = 0$, contradicting the NE assumption.

Proof of (4): Statements (1) and (3) imply that $\tilde{\mathcal{S}}_t^k = \{i^{(1)}, i^{(2)}, ..., i^{(g)}\}$ for some integer g in $[2, N]$. From $\frac{c_{i^{(g)}}}{q_{i^{(g)},t}^k} < \frac{\sum_{j\in\tilde{\mathcal{S}}_t^k} \frac{c_j}{q_{j,t}^k}}{n_0-1}$, we have $g \leq h$. Assume that $g < h$, then $\frac{c_{i^{(g+1)}}}{q_{i^{(g+1)},t}^k} \geq \frac{\sum_{j=1}^{g+1} \frac{c_{i^{(j)}}}{q_{i^{(j)},t}^k}}{g}$, which implies $\sum_{j=1}^{g} \frac{c_{i_j^{Sort}}}{q_{i_j^{Sort},t}^k} \frac{c_{i^{(g+1)}}}{q_{i^{(g+1)},t}^k} > 0$.

$$\begin{aligned}\frac{\partial u_{i^{(g+1)},t}^k}{\partial r_{i^{(g+1)},t}^k} &= \frac{q_{i^{(g+1)},t}^k \sum_{j\in\tilde{\mathcal{S}}_t^k} q_{j,t}^k r_{j,t}^k}{(\sum_{j\in\tilde{\mathcal{S}}_t^k} q_{j,t}^k r_{j,t}^k)^2} P_t^k - c_{i^{(g+1)}} \\ &= \frac{q_{i^{(g+1)},t}^k}{\sum_{j\in\tilde{\mathcal{S}}_t^k} q_{j,t}^k r_{j,t}^k} P_t^k - c_{i^{(g+1)}} \\ &= \frac{q_{i_{g+1}^{sort},t}^k \sum_{j=1}^{g} q_{i^{(j)},t}^k}{g-1} - c_{i^{(g+1)}} > 0,\end{aligned}$$

where the third equation comes from (18). This contradiction proves $g = h$, and thus proves statement (4).

Appendix B
Proof of Theorem 2

Proof. The following three observations are used to prove Algorithm 1 finds the Nash equilibrium of task k.

(1) For any $i \in \mathcal{S}_t^k$, $\frac{c_i}{q_{i,t}^k} \geq \frac{\sum_{j\in\mathcal{S}_t^k} \frac{c_j}{q_{j,t}^k}}{n_0-1}$, where $n_0 = |\mathcal{S}_t^k|$.

(2) $\sum_{j\in\mathcal{S}_t^k} q_{j,t}^k {r_{j,t}^k}^{\mathrm{NE}} = \frac{(n_0-1)P_t^k}{\sum_{j\in\mathcal{S}_t^k} \frac{c_j}{q_{j,t}^k}}$.

(3) $\sum_{j\in\mathcal{S}_t^k\setminus\{i\}} q_{j,t}^k {r_{j,t}^k}^{\mathrm{NE}} = \frac{(n_0-1)^2 c_i P_t^k}{q_{j,t}^k (\sum_{j\in\mathcal{S}_t^k} \frac{c_j}{q_{j,t}^k})^2}$

Based on the above observations, we prove two cases respectively: client $i \in \mathcal{S}_t^k$ and client $i \notin \mathcal{S}_t^k$. For any $i \notin \mathcal{S}_t^k$,

$$\begin{aligned}\frac{c_i}{q_{i,t}^k}\sum_{j\in\mathcal{S}_t^k} q_{j,t}^k {r_{j,t}^k}^{\text{NE}} &= \frac{c_i}{q_{i,t}^k}\sum_{j\in\mathcal{S}_t^k} q_{j,t}^k {r_{j,t}^k}^{\text{NE}} \\ &\overset{(2)}{=} \frac{c_i}{q_{i,t}^k}\frac{(n_0-1)P_t^k}{\sum_{j\in\mathcal{S}_t^k}\frac{c_j}{q_{j,t}^k}} \\ &\overset{(1)}{\geq} \frac{\sum_{j\in\mathcal{S}_t^k}\frac{c_j}{q_{j,t}^k}}{n_0-1}\frac{(n_0-1)P_t^k}{\sum_{j\in\mathcal{S}_t^k}\frac{c_j}{q_{j,t}^k}} = P_t^k, \quad (20)\end{aligned}$$

thus $\beta_{i,t}^k({r_{-i,t}^k}^{\text{NE}}) = 0$. For any $i \in \mathcal{S}_t^k$,

$$\sum_{j\in\mathcal{N}\setminus\{i\}} q_{j,t}^k {r_{j,t}^k}^{\text{NE}} = \sum_{j\in\mathcal{S}_t^k\setminus\{i\}} q_{j,t}^k {r_{j,t}^k}^{\text{NE}} \quad (21)$$

$$\overset{(3)}{=} \frac{c_i}{q_{i,t}^k}\frac{(n_0-1)^2 c_i P_t^k}{q_{j,t}^k(\Sigma_{j\in\mathcal{S}_t^k}\frac{c_j}{q_{j,t}^k})^2} = (\frac{c_i}{q_{i,t}^k})^2\frac{(n_0-1)^2P_t^k}{(\Sigma_{j\in\mathcal{S}_t^k}\frac{c_j}{q_{j,t}^k})^2}, \quad (22)$$

$$\begin{aligned}(n_0-1)\frac{c_i}{q_{i,t}^k} &= (i-1)\frac{c_i}{q_{i,t}^k} + (n_0-i)\frac{c_i}{q_{i,t}^k} \\ &< \sum_{j=1}^{i}\frac{c_i}{q_{j,t}^k} + \sum_{j=i+1}^{n_0}\frac{c_i}{q_{j,t}^k} = \sum_{j=1}^{n_0}\frac{c_i}{q_{j,t}^k}, \quad (23)\end{aligned}$$

where the second inequality holds because $\frac{c_i}{q_{i,t}^k} < \frac{\frac{c_i}{q_{i,t}^k}+\sum_{j\in\mathcal{S}_t^k}\frac{c_j}{q_{j,t}^k}}{|\mathcal{S}_t^k|}$ holds for client $i \in \mathcal{S}_t^k$, which comes from Line 5 of Algorithm 1. Therefore, we have

$$\begin{aligned}&\frac{c_i}{\hat{q}_{i,t}^k}\sum_{j\in\mathcal{N}\setminus\{i\}}\hat{q}_{j,t}^k {r_{j,t}^k}^{\text{NE}} \\ <&(\frac{\sum_{j=1}^{n_0}\frac{c_j}{\hat{q}_{j,t}^k}}{n_0-1})^2\frac{(n_0-1)^2P_t^k}{(\Sigma_{j\in S_t^k}\frac{c_j}{\hat{q}_{j,t}^k})^2} = P_t^k, \quad (24)\end{aligned}$$

and thus the strategy of client $i \in \mathcal{S}_t^k$ is

$$\begin{aligned}\beta_{i,t}^k({r_{-i,t}^k}^{\text{NE}}) &= \frac{\sqrt{\frac{q_{i,t}^k P_t^k\sum_{j\in\mathcal{N}\setminus\{i\}}q_{j,t}^k r_{j,t}^k}{c_i}} - \sum_{j\in\mathcal{N}\setminus\{i\}}q_{j,t}^k r_{j,t}^k}{q_{i,t}^k} \\ &= \frac{(|S_t^k|-1)P_t^k}{q_{i,t}^k\sum_{j\in S_t^k}\frac{c_j}{q_{j,t}^k}}(1-\frac{(|S_t^k|-1)c_i}{q_{i,t}^k\sum_{j\in S_t^k}\frac{c_j}{q_{j,t}^k}}) = {r_{i,t}^k}^{\text{NE}}. \\ &\quad (25)\end{aligned}$$

Therefore, the strategy in Algorithm 1 is the NE of the Stackelberg game of task k, which corresponds to the solution in (6). □

□

APPENDIX C
PROOFS OF THEOREM 3

In this section, we provide the proof of Theorem 4.

Lemma 1. *Suppose that there are three sequences:*

1. $\{a_1, a_2, \cdots, a_n\}$, *where* $a_1 < a_2 < \cdots < a_n$,
2. $\{b_1, b_2, \cdots, b_n\}$, *where* $b_1 \leq a_1$, $b_2 \leq a_2$, $\cdots$, $b_n \leq a_n$,
3. *Sort sequence 2 in ascending order and denoted as* $\{b_1', b_2', \cdots, b_n'\}$, *where* $b_1' < b_2' < \cdots < b_n'$.

For sequence 1 and sequence 3, we select elements according to the following rules:

- *First select the two smallest elements in the sequence 1 and 3 and denote the selected sets of sequence 1 and 3 as* $\mathcal{S}_a$ *and* $\mathcal{S}_b$ *respectively (note that sequences 1 and 3 are sorted in ascending order, so the first two elements are the two smallest elements).*
- *Then for the elements in sequences 1, we verify in order if* $a_i < \frac{a_i+\sum_{j\in\mathcal{S}_a}a_j}{|\mathcal{S}_a|}$. *If the condition is satisfied, then* a_i *will be selected into* $\mathcal{S}_a$ *and* $\mathcal{S}_a$ *will be updated. If not, then the selection stops.*
- *The same rule* $b_h' < \frac{b_h'+\sum_{j\in\mathcal{S}_b}b_j'}{|\mathcal{S}_b|}$ *is used to determine whether to select* b_h' *and update* $\mathcal{S}_b$.

Finally, we denote the size of final result of $\mathcal{S}_a$ *and* $\mathcal{S}_b$ *as* $n_a = |\mathcal{S}_a|$ *and* $n_b = |\mathcal{S}_b|$ *respectively. We have* $\frac{\sum_{j=1}^{n_a}a_j}{n_a-1} \geq \frac{\sum_{j=1}^{n_b}b_j'}{n_b-1}$.

Proof. We prove this lemma in three cases: (1) $n_a = n_b$, (2) $n_a < n_b$, and (3) $n_a > n_b$.

We first prove the case of $n_a = n_b$. Obviously, we have

$$\frac{\sum_{j=1}^{n_b}b_j'}{n_b-1} \leq \frac{\sum_{j=1}^{n_b}b_j}{n_b-1} \leq \frac{\sum_{j=1}^{n_b}a_j}{n_b-1} = \frac{\sum_{j=1}^{n_a}a_j}{n_a-1}.$$

Then we prove the case of $n_a < n_b$. For any $h < n_b$, we have $b_h' < \frac{\sum_{j=1}^{h-1}b_j'}{h-2}$ from the selection condition $b_h' < \frac{b_h'+\sum_{j\in\mathcal{S}_b}b_j'}{|\mathcal{S}_b|}$. Thus we have

$$\begin{aligned}\frac{\sum_{j=1}^{n_b}b_j'}{n_b-1} &= \frac{b_{n_b}'+\sum_{j=1}^{n_b-1}b_j'}{n_b-1} < \frac{\frac{\sum_{j=1}^{n_b-1}b_j'}{n_b-2}+\sum_{j=1}^{n_b-1}b_j'}{n_b-1} \\ &= \frac{\sum_{j=1}^{n_b-1}b_j'}{n_b-2} = \frac{b_{n_b-1}'+\sum_{j=1}^{n_b-2}b_j'}{n_b-2} \\ &< \frac{\frac{\sum_{j=1}^{n_b-2}b_j'}{n_b-3}+\sum_{j=1}^{n_b-2}b_j'}{n_b-2} = \frac{\sum_{j=1}^{n_b-2}b_j'}{n_b-3} \\ &< \cdots < \frac{\sum_{j=1}^{n_b-\Delta n}b_j'}{n_b-\Delta n-1} = \frac{\sum_{j=1}^{n_a}b_j'}{n_a-1} \\ &\leq \frac{\sum_{j=1}^{n_a}b_j}{n_a-1} \leq \frac{\sum_{j=1}^{n_a}a_j}{n_a-1}. \quad (26)\end{aligned}$$

Finally, we prove the case of $n_a > n_b$. For any $i \leq n_a$, we have $a_i \leq a_{n_a} < \frac{a_{n_a}+\sum_{j=1}^{n_a-1}a_j}{n_a-1} < \frac{\sum_{j=1}^{n_a}a_j}{n_a-1}$ from the selection condition $a_i < \frac{a_i+\sum_{j\in\mathcal{S}_a}a_j}{|\mathcal{S}_a|}$. For any $h > n_b$, we have $b_h' \geq \frac{\sum_{j=1}^{n_b}b_j'}{n_b-1}$ since it does not satisfy the selection condition $b_h' < \frac{b_h'+\sum_{j\in\mathcal{S}_b}b_j'}{|\mathcal{S}_b|}$. Since $n_a > n_b$, there must be at least one element in sequence 1 which is selected into $\mathcal{S}_a$, but its corresponding lower bound in sequence 3 is not selected into $\mathcal{S}_b$ Denote any of these special elements in sequence 1 as a_{i_0}, and denote its corresponding lower bound in sequence 2 and sequence 3 as b_{i_0} and b_{h_0}', we have $a_{i_0} \geq b_{i_0} = b_{h_0}'$. With the help of this special element, we have $\frac{\sum_{j=1}^{n_b}b_j'}{n_b-1} \leq b_{h_0}' \leq a_{i_0} < \frac{\sum_{j=1}^{n_a}a_j}{n_a-1}$. □

Proof.

$$u_{0,t}^k = \gamma \log(\gamma X_t^k) + \frac{1}{X_t^k} - \gamma, X_t^k = \frac{n_0 - 1}{\sum_{j\in \mathcal{S}_t^k} \frac{c_j}{q_{j,t}^k}} \tag{27}$$

$$\hat{u}_{0,t}^k = \gamma \log(\gamma \hat{X}_t^k) + \frac{1}{\hat{X}_t^k} - \gamma, \hat{X}_t^k = \frac{\hat{n}_0 - 1}{\sum_{j\in \hat{\mathcal{S}}_t^k} \frac{c_j}{q_{j,t}^k}}. \tag{28}$$

For function $f(x) = \gamma \log(\gamma x) + \frac{1}{x} - \gamma$, we compute the first derivative of $f(x)$ as $\frac{\partial f(x)}{\partial x} = \frac{\gamma}{x} - \frac{1}{x^2}$. Let $\frac{\partial f(x)}{\partial x} = 0$, we have $x = \frac{1}{\gamma}$. Since $\frac{\partial f(x)}{\partial x}|_{x=\frac{2}{\gamma}} = \frac{\gamma^2}{4} > 0$ and $\frac{\partial f(x)}{\partial x}|_{x=\frac{1}{2\gamma}} = -2\gamma^2 < 0$, we have $f(x)$ is a convex function and $f(x)$ reaches its minimum at $x = \frac{1}{\gamma}$. Therefore, for $x_1 \geq x_2 \geq \frac{1}{\gamma}$, we have $f(x_1) \geq f(x_2)$. Therefore, proving $\hat{u}_{0,t}^k \geq u_{0,t}^k$ is equivalent to proving $\hat{X}_t^k \geq X_t^k$.

We utilizes Lemma 1 to prove $\hat{X}_t^k \geq X_t^k$, where sequence 1 can be referred to as the sorted sequence $\{\frac{c_1}{q_{1,t}^k}, \frac{c_2}{q_{2,t}^k}, \cdots, \frac{c_i}{q_{i,t}^k}\}$, where $\frac{c_1}{q_{1,t}^k} \leq \frac{c_2}{q_{2,t}^k} \leq \cdots \leq \frac{c_i}{q_{i,t}^k}$. Sequence 2 can be referred to as $\{\frac{c_1}{\hat{q}_{1,t}^k}, \frac{c_2}{\hat{q}_{2,t}^k}, \cdots, \frac{c_i}{\hat{q}_{i,t}^k}\}$ and $\frac{c_1}{\hat{q}_{1,t}^k} \leq \frac{c_1}{q_{1,t}^k}, \frac{c_2}{\hat{q}_{2,t}^k} \leq \frac{c_1}{q_{1,t}^k}, \cdots, \frac{c_i}{\hat{q}_{i,t}^k} \leq \frac{c_1}{q_{1,t}^k}$ with probability at least $(1-\delta)^N$. Sequence 3 can be referred to as an ascending sort of $\{\frac{c_1}{\hat{q}_{1,t}^k}, \frac{c_2}{\hat{q}_{2,t}^k}, \cdots, \frac{c_i}{\hat{q}_{i,t}^k}\}$.

According to Lemma 1, we have

$$\frac{\sum_{j\in \hat{\mathcal{S}}_t^k} \frac{c_j}{q_{j,t}^k}}{\hat{n}_0 - 1} \leq \frac{\sum_{j\in \mathcal{S}_t^k} \frac{c_j}{q_{j,t}^k}}{n_0 - 1}. \tag{29}$$

Since $X_t^k = \frac{n_0-1}{\sum_{j\in \mathcal{S}_t^k} \frac{c_j}{q_{j,t}^k}}$ and $\hat{X}_t^k = \frac{\hat{n}_0-1}{\sum_{j\in \hat{\mathcal{S}}_t^k} \frac{c_j}{q_{j,t}^k}}$, we have $\hat{X}_t^k \geq X_t^k$ with probability at least $(1-\delta)^N$. Thus we prove Lemma 3. □

APPENDIX D
PROOFS OF THEOREM 4

In this section, we provide the proof of Theorem 4. First, we bound the **single-round regret**. Specifically, we show in Lemma 2 that the regret in the server's utility in a given round can be bounded by the uncertainty—i.e., the confidence bounds—of the clients' utility estimates in that round. This result establishes a crucial connection between the server's performance and the quality of client-side task assignments.

Lemma 2. *For a single round t, the regret of Algorithm 2 is upper bounded by*

$$r_t \leq \gamma \log(1 + \frac{1}{\hat{n}_0 - 1} + \frac{\sum_{j\in \hat{\mathcal{S}}_t^k} \alpha_{j,t} ||\boldsymbol{x}_t||_{\boldsymbol{V}_{j,t-1}^{-1}}}{q_{\min}}), \tag{30}$$

where $q_{\min}$ denotes the minimum of data quality.

Proof.

$$\begin{aligned} r_t &= u_{0,t}^{k^*} - u_{0,t}^{k_t} \\ &= (u_{0,t}^{k^*} - \hat{u}_{0,t}^{k^*}) + (\hat{u}_{0,t}^{k^*} - \hat{u}_{0,t}^{k_t}) + (\hat{u}_{0,t}^{k_t} - u_{0,t}^{k_t}) \\ &\leq \hat{u}_{0,t}^{k_t} - u_{0,t}^{k_t}, \end{aligned}$$

where the inequality is due to the fact that $\hat{u}_{0,t}^{k^*} \geq u_{0,t}^{k^*}$ according to Lemma 3 and $\hat{u}_{0,t}^{k_t} \geq u_{0,t}^{k_t}$ since the server selects the estimated optimal arm k_t.

Next, we analyze the difference between $\hat{u}_{0,t}^{k_t}$ and $u_{0,t}^{k_t}$. To simplify the notation, we analyze the difference between $\hat{u}_{0,t}^k$ and $u_{0,t}^k$ in the following steps, which is also the bound of the difference between $\hat{u}_{0,t}^{k_t}$ and $u_{0,t}^{k_t}$.

$$\begin{aligned} &\hat{u}_{0,t}^k - u_{0,t}^k \\ &= \gamma \log(\gamma \hat{X}_t^k) + \frac{1}{\hat{X}_t^k} - \gamma - (\gamma \log(\gamma X_t^k) + \frac{1}{X_t^k} - \gamma) \\ &= \gamma \log(\frac{\gamma \hat{X}_t^k}{\gamma X_t^k}) + \frac{1}{\hat{X}_t^k} - \frac{1}{X_t^k} \\ &\leq \gamma \log(\frac{\hat{X}_t^k}{X_t^k}), \end{aligned}$$

where the inequality is due to the fact that $\hat{X}_t^k \geq X_t^k$ we proved in Lemma 3. Therefore, we need to analyze the bound of $\frac{\hat{X}_t^k}{X_t^k}$. We analyze the bound from three cases: $\mathcal{S}_t^k = \hat{\mathcal{S}}_t^k$, $\mathcal{S}_t^k \subset \hat{\mathcal{S}}_t^k$, and $\mathcal{S}_t^k \not\subset \hat{\mathcal{S}}_t^k$.

For the case of $\mathcal{S}_t^k = \hat{\mathcal{S}}_t^k$, we have

$$\begin{aligned} \frac{\hat{X}_t^k}{X_t^k} &= \frac{\frac{\hat{n}_0-1}{\sum_{h\in \hat{\mathcal{S}}_t^k} \frac{c_h}{\hat{q}_{h,t}^k}}}{\frac{n_0-1}{\sum_{j\in \mathcal{S}_t^k} \frac{c_j}{q_{j,t}^k}}} = \frac{\sum_{j\in \mathcal{S}_t^k} \frac{c_j}{q_{j,t}^k}}{\sum_{h\in \hat{\mathcal{S}}_t^k} \frac{c_h}{\hat{q}_{h,t}^k}} \\ &= \sum_{j\in \mathcal{S}_t^k} \frac{c_j}{q_{j,t}^k \sum_{h\in \hat{\mathcal{S}}_t^k} \frac{c_h}{\hat{q}_{h,t}^k}} \leq \sum_{j\in \hat{\mathcal{S}}_t^k} \frac{c_j}{q_{j,t}^k \frac{c_j}{\hat{q}_{j,t}^k}(\hat{n}_0 - 1)} \\ &= \frac{1}{\hat{n}_0 - 1} \sum_{j\in \hat{\mathcal{S}}_t^k} \frac{\hat{q}_{j,t}^k}{q_{j,t}^k} = \frac{1}{\hat{n}_0 - 1} \sum_{j\in \hat{\mathcal{S}}_t^k} \frac{q_{j,t}^k + \alpha_{j,t}||\boldsymbol{x}_t||_{\boldsymbol{V}_{j,t-1}^{-1}}}{q_{j,t}^k} \\ &= \frac{1}{\hat{n}_0 - 1}(\hat{n}_0 + \frac{\sum_{j\in \hat{\mathcal{S}}_t^k} \alpha_{j,t}||\boldsymbol{x}_t||_{\boldsymbol{V}_{j,t-1}^{-1}}}{q_{\min}}) \\ &\leq \frac{\hat{n}_0}{\hat{n}_0 - 1} + \frac{\sum_{j\in \hat{\mathcal{S}}_t^k} \alpha_{j,t}||\boldsymbol{x}_t||_{\boldsymbol{V}_{j,t-1}^{-1}}}{q_{\min}} \\ &= 1 + \frac{1}{\hat{n}_0 - 1} + \frac{\sum_{j\in \hat{\mathcal{S}}_t^k} \alpha_{j,t}||\boldsymbol{x}_t||_{\boldsymbol{V}_{j,t-1}^{-1}}}{q_{\min}}. \end{aligned}$$

For the case of $\hat{\mathcal{S}}_t^k \subset \mathcal{S}_t^k$, we obtain $\frac{\sum_{j\in \mathcal{S}_t^k} \frac{c_j}{q_{j,t}^k}}{n_0-1} \leq \frac{\sum_{j\in \hat{\mathcal{S}}_t^k} \frac{c_j}{q_{j,t}^k}}{\hat{n}_0-1}$ by using the same method as (26) when proving case (2) in proof of lemma 1. Thus we have

$$\begin{aligned} \frac{\hat{X}_t^k}{X_t^k} &= \frac{\frac{\hat{n}_0-1}{\sum_{h\in \hat{\mathcal{S}}_t^k} \frac{c_h}{\hat{q}_{h,t}^k}}}{\frac{n_0-1}{\sum_{j\in \mathcal{S}_t^k} \frac{c_j}{q_{j,t}^k}}} \leq \frac{\frac{\hat{n}_0-1}{\sum_{h\in \hat{\mathcal{S}}_t^k} \frac{c_h}{\hat{q}_{h,t}^k}}}{\frac{\hat{n}_0-1}{\sum_{j\in \hat{\mathcal{S}}_t^k} \frac{c_j}{q_{j,t}^k}}} = \frac{\sum_{j\in \hat{\mathcal{S}}_t^k} \frac{c_j}{q_{j,t}^k}}{\sum_{h\in \hat{\mathcal{S}}_t^k} \frac{c_h}{\hat{q}_{h,t}^k}} \\ &\leq 1 + \frac{1}{\hat{n}_0 - 1} + \frac{\sum_{j\in \hat{\mathcal{S}}_t^k} \alpha_{j,t}||\boldsymbol{x}_t||_{\boldsymbol{V}_{j,t-1}^{-1}}}{q_{\min}}. \end{aligned}$$

For the case of $\hat{\mathcal{S}}_t^k \not\subset \mathcal{S}_t^k$, there must be at least one client that is selected into $\hat{\mathcal{S}}_t^k$, but is not selected into $\mathcal{S}_t^k$. Thus we can find the special client $j_0 = h_0$ that satisfies $j_0 \in \hat{\mathcal{S}}_t^k$ and

$h_0 \notin \mathcal{S}_t^k$. We have $\frac{c_{j_0}}{\hat{q}_{j_0,t}^k} < \frac{\sum_{j\in\hat{\mathcal{S}}_t^k}\frac{c_j}{\hat{q}_{j,t}^k}}{\hat{n}_0-1}$ and $\frac{c_{h_0}}{q_{h_0,t}^k} \geq \frac{\sum_{j\in\mathcal{S}_t^k}\frac{c_j}{q_{j,t}^k}}{n_0-1}$ from the selection condition. Therefore, we have

$$\frac{\hat{X}_t^k}{X_t^k} = \frac{\frac{\hat{n}_0-1}{\sum_{h\in\hat{\mathcal{S}}_t^k}\frac{c_h}{\hat{q}_{h,t}^k}}}{\frac{n_0-1}{\sum_{j\in\mathcal{S}_t^k}\frac{c_j}{q_{j,t}^k}}} \leq \frac{\frac{\hat{q}_{j_0,t}^k}{c_{j_0}}}{\frac{q_{h_0,t}^k}{c_{h_0}}} = \frac{\frac{\hat{q}_{j_0,t}^k}{c_{j_0}}}{\frac{q_{j_0,t}^k}{c_{j_0}}} = \frac{\hat{q}_{j_0,t}^k}{q_{j_0,t}^k}$$
$$= 1 + \frac{\alpha_{j_0,t}||\boldsymbol{x}_t||_{\boldsymbol{V}_{j_0,t-1}^{-1}}}{q_{\min}}$$
$$\leq 1 + \frac{1}{\hat{n}_0-1} + \frac{\sum_{j\in\hat{\mathcal{S}}_t^k}\alpha_{j,t}||\boldsymbol{x}_t||_{\boldsymbol{V}_{j,t-1}^{-1}}}{q_{\min}}.$$

Therefore, the regret in single round t is bounded by

$$r_t \leq \gamma\log(\frac{\hat{X}_t^k}{X_t^k})$$
$$\leq \gamma\log(1 + \frac{1}{\hat{n}_0-1} + \frac{\sum_{j\in\hat{\mathcal{S}}_t^k}\alpha_{j,t}||\boldsymbol{x}_t||_{\boldsymbol{V}_{j,t-1}^{-1}}}{q_{\min}}).$$

□

Next, we bound the **cumulative regret** of the server's utility. This is done by leveraging the confidence bounds on service quality to construct a martingale sequence, which enables us to show that the sum of the confidence bounds over time converges, as shown in Lemma 3.

Lemma 3 (Lemma 11 in [10])**.** *Let $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n \in \mathbb{R}^d$ be a sequence in $\mathbb{R}^d$, $\boldsymbol{U}_0 \in \mathbb{R}^{d\times d}$ a positive definte matrix and define $\boldsymbol{U}_n = \boldsymbol{U}_0 + \sum_{t=1}^n \boldsymbol{x}_t\boldsymbol{x}_t^\top$ for all t.Then, we have that*

$$\log\left(\frac{\det(\boldsymbol{U}_n)}{\det(\boldsymbol{U}_0)}\right) \leq \sum_{t=1}^n ||\boldsymbol{x}_t||^2_{\boldsymbol{U}_{t-1}^{-1}}$$

Further, if $||\boldsymbol{x}_t||_2 \leq L$, then

$$\sum_{t=1}^n \min\{1, ||\boldsymbol{x}_t||^2_{\boldsymbol{U}_{t-1}^{-1}}\} \leq 2(\log\det(\boldsymbol{U}_t) - \log\det\boldsymbol{U}_0),$$

and finally, if $\lambda_{\min}(\boldsymbol{U}_0) \geq \max(1, L^2)$ then

$$\sum_{t=1}^n ||\boldsymbol{x}_t||^2_{\boldsymbol{U}_{t-1}^{-1}} \leq 2\log\frac{\det(\boldsymbol{U}_n)}{\det(\boldsymbol{U}_0)}.$$

With the help of Lemma 2 and 3, we can analyze the regret of Algorithm 2 in multiple rounds.

Proof. According to Lemma 2, the regret in single round t is bounded by

$$r_t \leq \gamma\log(1 + \frac{1}{\hat{n}_0-1} + \frac{\sum_{j\in\hat{\mathcal{S}}_t^k}\alpha_{j,t}||\boldsymbol{x}_t||_{\boldsymbol{V}_{j,t-1}^{-1}}}{q_{\min}}). \quad (31)$$

Therefore, the total regret in T rounds is bounded by

$$R_T = \sum_{t=1}^T r_t$$
$$\leq \sum_{t=1}^T \gamma\log(1 + \frac{1}{\hat{n}_0-1} + \frac{\sum_{j\in\hat{\mathcal{S}}_t^k}\alpha_{j,t}||\boldsymbol{x}_t||_{\boldsymbol{V}_{j,t-1}^{-1}}}{q_{\min}})$$

Denote $Y_t^k = \frac{\sum_{j\in\hat{\mathcal{S}}_t^k}\alpha_{j,t}||\boldsymbol{x}_t||_{\boldsymbol{V}_{j,t-1}^{-1}}}{q_{\min}}$. Since $\frac{1}{\hat{n}_0-1}$ is small enough, it is resonable to assume that Y_t^k is large enough so that $1 + \frac{1}{\hat{n}_0-1} \leq e^{Y_t^k} - Y_t^k$. Therefore, we can safely assume that $\log(1 + \frac{1}{\hat{n}_0-1} + Y_t^k) \leq Y_t^k$. Thus, the total regret is bounded by

$$R_T \leq \gamma\sum_{t=1}^T \frac{\sum_{j\in\hat{\mathcal{S}}_t^k}\alpha_{j,t}||\boldsymbol{x}_t||_{\boldsymbol{V}_{j,t-1}^{-1}}}{q_{\min}}$$
$$= \frac{\gamma}{q_{\min}}\sum_{j\in\mathcal{N}}\sum_{t\in\mathcal{T}_j}\alpha_{j,t}||\boldsymbol{x}_t||_{\boldsymbol{V}_{j,t-1}^{-1}}$$
$$\leq \frac{\gamma\alpha_T}{q_{\min}}\sum_{j\in\mathcal{N}}\sqrt{T\sum_{t\in\mathcal{T}_j}||\boldsymbol{x}_t||^2_{\boldsymbol{V}_{j,t-1}^{-1}}}$$
$$\leq \frac{\gamma\alpha_T}{q_{\min}}\sum_{j\in\mathcal{N}}\sqrt{2Td\log(\frac{\det(\boldsymbol{V}_{j,t-1})}{\det(\boldsymbol{V}_{j,0})})},$$

where the last inequality is due to Lemma 3. Obviously, $\det(\boldsymbol{V}_{j,0}) = \lambda^d$ and thus $\log(\det(\boldsymbol{V}_{j,0})) = d\log(\lambda)$. Denote the eigenvalues of $\boldsymbol{V}_{j,t-1}$ as λ_s. With the fact that $\det(\boldsymbol{V}_{j,t-1}) = \prod_s \lambda_s \leq (\frac{\text{trace}(\boldsymbol{V}_{j,0})+tL^2}{d})^d$ and $\text{trace}(\boldsymbol{V}_{j,0}) = \lambda d$, we have $\log(\det(\boldsymbol{V}_{j,t-1})) \leq d\log(\frac{\lambda d+tL^2}{d})$. Therefore,

$$R_T \leq \frac{\gamma\alpha_T}{q_{\min}}\sum_{j\in\mathcal{N}}\sqrt{2Td\log(\frac{\lambda d+TL^2}{\lambda d})}$$
$$\leq \frac{\gamma\sigma N}{q_{\min}}\sqrt{2Td\log(1+\frac{TL^2}{\lambda d})}$$
$$\cdot(\sqrt{2d\log(1+\frac{TL^2}{\lambda d}) + 2\log(\frac{2}{\delta})} + \sqrt{\lambda})$$
$$= O(\frac{\gamma\sigma dN}{q_{\min}}\sqrt{T}\log(TL^2/\lambda d)).$$

□