Online Worker Scheduling for Maximizing Long-Term Utility in Spatiotemporal Crowdsensing

Jiajun Wang[1], Xingjian Ding[1], Jianxiong Guo[2*], Zhiqing Tang[2], and Deying Li[3]
[1] Faculty of Information, Beijing University of Technology, Beijing, China
[2] Advanced Institute of Natural Sciences, Beijing Normal University, Zhuhai, China
[3] School of Information, Renmin University of China, Beijing, China
*Corresponding Author: Jianxiong Guo
S202375129@emails.bjut.edu.cn, dxj@bjut.edu.cn, {jianxiongguo,zhiqingtang}@bnu.edu.cn, deyingli@ruc.edu.cn

2024 IEEE/CIC International Conference on Communications in China (ICCC) | 979-8-3503-7841-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICCC62479.2024.10682026

***Abstract*—With the continuous development of mobile networks and sensing devices, spatiotemporal crowdsourcing has gradually become a new intelligent sensing paradigm for data acquisition and sharing in the Industrial Internet of Things. How to reasonably allocate tasks to workers in a dynamic environment to maximize the platform utility has become a research hotspot. Many past works have made great efforts in this regard, but most of them only consider the long-term constraints of resources, and ignore the short-term ability constraints of workers to provide resources. In this paper, we consider a platform-centered online spatiotemporal crowdsourcing system, where mobile workers have long-term and short-term resource constraints, while the platform has a long-term budget constraint. We aim to find an online worker scheduling scheme to maximize the platform's long-term utility without violating the constraints of workers and the platform. To address the problem, we first transform the long-term utility maximization problem into a real-time utility maximization problem by leveraging the Lyapunov optimization. Then, we design a centralized algorithm based on Markov approximation to solve the real-time optimization problem. Furthermore, we demonstrate that our proposed approach can achieve near-optimal performance for our problem. Finally, we evaluate our designs by numerical simulation experiments, and the results demonstrate the effectiveness of our algorithms.**

I. Introduction

With the rapid development of smart devices, spatiotemporal crowdsourcing stands out for its exceptional capabilities in task processing and real-time responsiveness. It has great potential to assist the Industrial Internet of Things (IIoT) systems, especially those requiring high levels of real-time responsiveness, and has attracted widespread attention from both academia and industry. For example, crowdsourcing helps transform most traditional services' passive response models for smart manufacturing applications into context-aware and proactive services, thus improving serviceability and productivity [1].

In a typical spatiotemporal crowdsourcing system, the platform dynamically publishes tasks [2], then selects appropriate mobile workers to perform the corresponding tasks, and specifies the amount of resources that the workers need to provide, at the same time, the platform will also pay corresponding recruitment fees to workers. The profit of the platform is related to the quality of task completion, which is based on the amount of resources provided by workers. Generally speaking, the total amount of resources of workers and the payment budget of the platform are limited, thus online worker scheduling (which is also known as online task allocation) plays an important role in improving the platform's long-term utility.

In the past few years, many attempts have been made in previous works on the online worker scheduling problem [2]–[10]. Diverse methods have been adopted in the previous works, such as online bipartite graph matching [6], [7], multi-armed bandit [3], Q-Learning [4], approximation algorithm [3], [8], heuristic algorithm [5]. For example, [6] modeled the problem as a dynamic delayed bipartite graph matching problem and designed two adaptive threshold frameworks based on Policy Gradient and Proximal Policy Optimization respectively to acquire the approximate optimal solution for allocating tasks. Considering timeliness and fairness, [5] propose the Utility-Fairness Index and address the Fairness-Aware Task Planning problem, making task planning for workers in real-time, such that the total utility is maximized and the fairness among workers is maintained. [3] focussed on solving the Mobile Crowdsourcing (MCS) task assignment problem by exploiting both task and worker context and established contextual combinatorial volatile multi-armed bandit which encapsulates a wide range of crowdsourcing problems. Although these efforts model the online task assignment problem from a variety of perspectives and come to inspirational conclusions, there are still several problems to be solved. Firstly, they did not take the long-term consumption of workers' resources for task completion into account, if resource consumption is not properly controlled, the crowdsourcing system may not run for a long time. How to maximize the long-term utility of the platform while rationally consuming worker resources is a challenge. Secondly, in online scenarios, tasks may be rapidly published in large quantities within a short period, which results in short-term resource consumption exceeding the capacity that workers can bear. How to effectively address this situation is another challenge.

In this paper, we investigate the online worker scheduling problem by jointly considering the long-term and short-term resource constraints of workers, as well as the long-term recruitment budget of the platform. The main challenges of our problem are twofold. First, as both workers and the platform have long-term constraints, how to allocate resources and budget to each time slot is a very crucial issue as it is almost impossible for us to get complete future information about tasks. Second, as each worker has a short-term resource constraint, how to assign tasks to corresponding workers to maximize the total utility is very tough as it falls into the category of the multi-knapsack problem. To address the above issues, we combine Lyapunov optimization and Markov approximation theory and design a framework for online worker scheduling to maximize platform utility while satisfying long-term and short-term constraints. The main contributions of our work are summarized below:

- We investigate the online worker scheduling problem for Maximizing Platform's Long-term utility in Platform-centric spatiotemporal crowdsourcing systems (MPLP), and give the formal formulation of the problem.
- To deal with the challenges of our problem, we first transform the MPLP problem into a real-time problem by leveraging the Lyapunov optimization, and then design a centralized algorithm based on Markov approximation to address the problem.
- We demonstrate that the proposed algorithms can achieve near-optimal performance for the MPLP problem, and prove the excellent performance of our designs by conducting extensive simulations.

II. System Model and Problem Formulation

A. System Model

We consider a stable spatiotemporal MCS system that contains a fixed number of m mobile workers denoted by $\mathcal{W} = \{w_1, w_2, \ldots, w_m\}$. The MCS platform will periodically publish tasks to workers. To conveniently represent the periodicity of system release tasks, we assume the MCS system works in a slotted model, and the timeline is divided T time slots, $i.e.$, $\mathcal{T} = \{1, 2, \ldots, T\}$. At the beginning of each time slot $t \in \mathcal{T}$, there is a set N_t of tasks published by the platform, which is denoted by $\mathcal{A}_t = \{a_1, a_2, \ldots, a_{N_t}\}$. We assume that each task is indivisible and thus can be executed by at most one worker, while one worker may perform multiple tasks within his/her capability.

For any worker $w_i \in \mathcal{W}$, there is a long-term time-average resource budget $\overline{B_i}$, which means that the average amount of resources invested by worker w_i in executing tasks in each time slot cannot exceed $\overline{B_i}$. In addition, considering the capability of each worker to perform tasks is limited, we assume the resource budget of the worker $w_i \in \mathcal{W}$ is b_i^{max} in a single time slot. For any task $a_j \in \mathcal{A}_t$, there is a minimal resource requirement $r_{j,t}^{min}$, which means that the task can only be executed if the amount of resources allocated to it is more than $r_{j,t}^{min}$. We use a $M \times N_t$ allocation matrix $\boldsymbol{R}_t$ to represent the worker scheduling strategy in time slot t, where each element R_{ij}^t denotes the amount of resources that worker w_i allocates to task a_j, and $R_{ij}^t = 0$ indicates that the worker w_i does not perform the task a_j in time slot t.

In a time slot t, the profit obtained by the platform from worker w_i performing task a_j can be calculated as $P_{ij}^t = \alpha_j \log(1 + \beta_j R_{ij}^t)$, if $R_{ij}^t \geq r_{j,t}^{min}$, otherwise, $P_{ij}^t = 0$, where $\alpha_j, \beta_j \in \mathbb{R}_+$ are coefficients. Assume that the unit resource price of workers is identical and is denoted as τ, then the platform will pay $C_i^t = \tau \cdot \sum_{a_j \in \mathcal{A}_t} R_{ij}^t$ for worker w_i in time slot t. The long-term time-average recruitment cost of the platform is limited on the entire timeline and is represented by $\overline{C_{bgt}}$.

B. Constraints of Problem

Allocation decision constraint: As each task can be executed by at most one worker, the following constraint must be satisfied:

$$\sum\nolimits_{w_i \in \mathcal{W}} \mathbb{I}\{R_{ij}^t > 0\} \leq 1, \quad \forall a_j \in \mathcal{A}_t, t \in \mathcal{T}, \tag{1}$$

where $\mathbb{I}\{R_{ij}^t > 0\}$ is the indicator function.

Worker resource consumption constraints: The Worker resource consumption must satisfy the long-term and short-term constraints:

$$\lim_{T \to \infty} \frac{1}{T} \sum\nolimits_{t=1}^{T} \sum\nolimits_{a_j \in \mathcal{A}_t} \mathbb{E}\{R_{ij}^t\} \leq \overline{B_i}, \quad \forall w_i \in \mathcal{W}, \tag{2}$$

$$\sum\nolimits_{a_j \in \mathcal{A}_t} R_{ij}^t \leq b_i^{max}, \quad \forall w_i \in \mathcal{W}, t \in \mathcal{T}. \tag{3}$$

Platform recruitment budget constraints: The time-average recruitment cost of the platform on the entire timeline must not exceed the budget $\overline{C_{bgt}}$:

$$\lim_{T \to \infty} \frac{1}{T} \sum\nolimits_{t=1}^{T} \sum\nolimits_{w_i \in \mathcal{W}} \mathbb{E}\{C_i^t\} \leq \overline{C_{bgt}}. \tag{4}$$

In the above constraints, the expectation function $\mathbb{E}\{\cdot\}$ is used to eliminate the influence of stochastic in the dynamic spatiotemporal MCS system.

C. Problem Definition

We aim to find an online worker scheduling strategy to maximize the long-term utility of the platform while satisfying the above constraints. The utility U_t of the platform in a time slot t is defined as the total profits of tasks minus the total recruitment cost on the entire timeline, that is, $U_t = \sum_{w_i \in \mathcal{W}} \sum_{a_j \in \mathcal{A}_t} P_{ij}^t - \sum_{w_i \in \mathcal{W}} C_i^t$. The problem is formally defined as follows.

Problem 1. Online worker scheduling for Maximizing Platform's Long-term utility in Platform-centric spatiotemporal crowdsourcing systems (MPLP). *Given the time slot sequence $\mathcal{T}$, the mobile worker set $\mathcal{W}$, the task set $\mathcal{A}_t$ for each time slot $t \in \mathcal{T}$, the MPLP problem aims to find a worker scheduling strategy $\boldsymbol{R}_t$ for each time slot to maximize platform's long-term utility under constraints (1)-(4), which can be written as*

$$(\boldsymbol{P1}) \quad \max_{\boldsymbol{R}_1, \ldots, \boldsymbol{R}_T} \lim_{T \to \infty} \frac{1}{T} \sum\nolimits_{t=1}^{T} \mathbb{E}\{U_t\},$$
$$\boldsymbol{s.t.} \quad (1) - (4).$$

III. Online Worker Scheduling Framework

A. Problem Transformation and Online Framework

The core challenge of the original MPLP problem $\boldsymbol{P1}$ is that the optimal worker scheduling strategy relies on future information which is impossible to obtain due to the dynamic and stochastic property of the spatiotemporal MCS system. To address the issue, we decompose the long-term optimization object under long-term constraints into each single time slot based on Lyapunov optimization, and transform the original problem into a queue stability control problem.

For clarity, we define $E_i^R(t) = \sum_{a_j \in \mathcal{A}_t} R_{ij}^t - \overline{B_i}$ and $E^C(t) = \sum_{w_i \in \mathcal{W}} C_i^t - \overline{C_{bgt}}$. Then, we define virtual queues with initial values of 0 for each long-term constraint of (2) and (4) respectively, that is,

$$Q_i^R(t+1) = \max\{Q_i^R(t) + E_i^R(t), 0\}, \forall w_i \in \mathcal{W} \quad (5)$$

$$Q^C(t+1) = \max\{Q^C(t) + E^C(t), 0\}. \quad (6)$$

Each virtual queue represents the exceeded budget of each constraint.

From Eq. (5), we have $Q_i^R(t+1) - Q_i^R(t) \geq \sum_{a_j \in \mathcal{A}_t} R_{ij}^t - \overline{B_i}$, for each $w_i \in \mathcal{W}$. By summing the inequality on each $t \in \mathcal{T}$ and take expectation, we have $\frac{1}{T}\sum_{t=1}^{T}\sum_{a_j \in \mathcal{A}_t} \mathbb{E}\{R_{ij}^t\} - \overline{B_i} \leq \mathbb{E}\{Q_i^R(T)\}/T$. Therefore, to satisfy the constraint (2), we only need to satisfy the following constraint.

$$\lim_{T \to \infty} \mathbb{E}\{Q_i^R(T)\}/T \leq 0, \forall w_i \in \mathcal{W}. \quad (7)$$

Similarly, constraint (4) can be satisfied by meeting the following constraint.

$$\lim_{T \to \infty} \mathbb{E}\{Q^C(T)\}/T \leq 0. \quad (8)$$

Constraints (7) and (8) suggest that we only need to control the stability of the virtual queues to satisfy the long-term constraints. Next, we introduce the *Lyapunov function* and *one-slot conditional Lyapunov drift* [11] to stabilize the virtual queues.

Define $\Theta(t)$ as the vector of all virtual queues at time slot t, i.e., $\Theta(t) \triangleq \{Q_1^R(t), Q_2^R(t), \ldots, Q_m^R(t), Q^C(t)\}$, the Lyapunov function is defined as follows.

$$L(\Theta(t)) \triangleq \frac{1}{2}\sum_{i=1}^{m} Q_i^R(t)^2 + \frac{1}{2}Q^C(t)^2. \quad (9)$$

The one-slot conditional Lyapunov drift is defined as $\Delta(\Theta(t)) \triangleq \mathbb{E}[L(\Theta(t+1)) - L(\Theta(t))|\Theta(t)]$, which reflects the expected backlog increment of all virtual queues over one slot. Then, we leverage the Lyapunov drift-plus-penalty function to approximately solve our problem on each time slot t, and we get the following problem:

$$(\boldsymbol{P2}) \quad \max_{\boldsymbol{R}_t} \mathbb{E}[V \cdot U_t|\Theta(t)] - \Delta(\Theta(t)), \forall t \in \mathcal{T},$$
$$\boldsymbol{s.t.} \quad (1), (3), (7), (8),$$

where V is a positive weight that balances utility and virtual queue backlogs.

According to the Lemma 4.6 in [11], we can derive that: $\Delta(\Theta(t)) \leq B + \sum_{w_i \in \mathcal{W}} Q_i^R(t)E_i^R(t) + Q^C(t)E^C(t)$, where B is a positive constant value for all $t \in \mathcal{T}$. Define that $\Omega(t) = \sum_{w_i \in \mathcal{W}} Q_i^R(t)E_i^R(t) + Q^C(t)E^C(t)$, then, problem $\boldsymbol{P2}$ can be approximately solved by addressing the following problem.

$$(\boldsymbol{P3}) \quad \max_{\boldsymbol{R}_t} \mathbb{E}[V \cdot U_t - \Omega(t)|\Theta(t)], \forall t \in \mathcal{T},$$
$$\boldsymbol{s.t.} \quad (1), (3).$$

The first component in the objective function of $\boldsymbol{P3}$ is about maximizing the platform's utility in each time slot, corresponding to the objective function of MPLP. The second component is about controlling the virtual queue backlogs, which reflects the exceeded budget of each time-average constraint. The positive weight V is used to adjust the trade-off between the two components. By solving $\boldsymbol{P3}$ on each time slot t, we get a feasible solution for the original MPLP problem. The proposed online algorithm is described in Algorithm 1.

Algorithm 1 Online MPLP Algorithm

Input: $\mathcal{W}$, $\mathcal{T}$, A_t for $t \in \mathcal{T}$, and control parameter V.
Output: Worker scheduling strategies $\boldsymbol{R}_1^*, \ldots, \boldsymbol{R}_T^*$.

1: $Q^C(0) = 0$, $Q_i^R(0) = 0$ for each $w_i \in \mathcal{W}$;
2: **for** $t = 0$ to $T-1$ **do**
3: Find the optimal solution $\boldsymbol{R}_t^*$ of $\boldsymbol{P3}$;
4: Calculate virtual queues $Q_i^R(t+1)$ and $Q^C(t+1)$ for the next time slot by Eqs. (5) and (6);
5: **end for**
6: **return** $\boldsymbol{R}_1^*, \ldots, \boldsymbol{R}_T^*$;

In Algorithm 1, we need to find the optimal solution of $\boldsymbol{P3}$, however, $\boldsymbol{P3}$ is an NP-hard problem due to its multi-knapsack property [12]. Therefore, in the next subsection, we propose a Markov approximation method to approximately solve $\boldsymbol{P3}$.

Notice that we omit constraints (7) and (8) in $\boldsymbol{P3}$, the two constraints are hidden in the second component of $\boldsymbol{P3}$'s optimization objective function, and the solution obtained by Algorithm 1 can satisfy these two constraints, which will be proved in next section.

B. Markov Approximation Method

In this subsection, we design a Markov approximation-based algorithm to approximately solve $\boldsymbol{P3}$ for Algorithm 1, which is inspired by [13]. We use $G(\boldsymbol{R}_t)$ to denote the objective function of $\boldsymbol{P3}$, then $\boldsymbol{P3}$ can be transformed into the following form:

$$(\boldsymbol{P4}) \ \max \ \sum_{\boldsymbol{R}_t \in \boldsymbol{\mathcal{F}}_t} p(\boldsymbol{R}_t) \cdot G(\boldsymbol{R}_t),$$
$$\boldsymbol{s.t.} \ \sum_{\boldsymbol{R}_t \in \boldsymbol{\mathcal{F}}_t} p(\boldsymbol{R}_t) = 1, \forall t \in \mathcal{T},$$

where $\boldsymbol{\mathcal{F}}_t$ is the collection of all feasible solutions, and $p(\boldsymbol{R}_t)$ represent the probability of the solution $\boldsymbol{R}_t$ is adopted at time slot t. Obviously, the optimal solution of $(\boldsymbol{P4})$ is to set $p(\boldsymbol{R}_t) = 1$ for $\boldsymbol{R}_t$ that maximize $G(\boldsymbol{R}_t)$.

Let $\Gamma = \frac{1}{\gamma}\sum_{\boldsymbol{R}_t \in \boldsymbol{\mathcal{F}}_t} p(\boldsymbol{R}_t) \cdot \log p(\boldsymbol{R}_t)$, where γ denote a positive constant that controls the approximation ratio of the entropy term. The problem can be approximated as log-sum-exp problem [14]:

$$(\boldsymbol{LSE-P4})\ \min\ \sum_{\boldsymbol{R}_t\in\boldsymbol{\mathcal{F}}_t} p(\boldsymbol{R}_t)\cdot G\left(\boldsymbol{R}_t\right)+\Gamma,$$
$$\boldsymbol{s.t.}\ \sum_{\boldsymbol{R}_t\in\boldsymbol{\mathcal{F}}_t} p(\boldsymbol{R}_t)=1, \forall t\in\mathcal{T}.$$

The optimality gap between $\boldsymbol{LSE-P4}$ and $\boldsymbol{P4}$ is bounded by $\frac{1}{\gamma}\log|\boldsymbol{\mathcal{F}}_t|$ according to [14]. Actually, the problem $\boldsymbol{LSE-P4}$ converges to the problem $\boldsymbol{P4}$ when γ approaches infinity. By utilizing the Karush-Kuhn-Tucker (KKT) condition [15], we can get the optimal solution of $\boldsymbol{LSE-P4}$ for any $t\in\mathcal{T}, \boldsymbol{R}_t\in\boldsymbol{\mathcal{F}}_t$:

$$p(\boldsymbol{R}_t)=\frac{\exp\left(\gamma\cdot G\left(\boldsymbol{R}_t\right)\right)}{\sum_{\widetilde{\boldsymbol{R}}_t\in\boldsymbol{\mathcal{F}}_t}\exp\left(\gamma\cdot G\left(\widetilde{\boldsymbol{R}}_t\right)\right)}. \tag{10}$$

Then we can find the solution for $\boldsymbol{P4}$ by choosing $\boldsymbol{R}_t$ with the maximum probability $p(\boldsymbol{R}_t)$ got from Eq. (10). Next, we design a Markov chain-based algorithm to solve the problem $\boldsymbol{LSE-P4}$, which also returns a feasible solution for $\boldsymbol{P4}$.

The key idea of the Markov chain-based algorithm is to create a time-reversible ergodic Markov chain [14] that achieves the stationary distribution as shown in (10). The constructed Markov chain should be irreducible, that is, any state is reachable from any other state. Also, the following balance equation should be satisfied: $p(\boldsymbol{R}_t)\cdot p(\boldsymbol{R}_t,\boldsymbol{R}_t')=p(\boldsymbol{R}_t'))\cdot p(\boldsymbol{R}_t'),\boldsymbol{R}_t),\forall\boldsymbol{R}_t,\boldsymbol{R}_t'\in\boldsymbol{\mathcal{F}}_t$, and $\boldsymbol{R}_t\neq\boldsymbol{R}_t'$, where $\boldsymbol{\mathcal{F}}_t$ is the state space, and $p(\boldsymbol{R}_t,\boldsymbol{R}_t')$ is the transition probability from state $\boldsymbol{R}_t$ to $\boldsymbol{R}_t'$. Based on Lemma 1 of [14], we could construct such a Markov chain as follows. First, we treat the solution space $\boldsymbol{\mathcal{F}}_t$ of $\boldsymbol{LSE-P4}$ as the state space, and the transition probability $p(\boldsymbol{R}_t,\boldsymbol{R}_t')$ for any two states $\boldsymbol{R}_t,\boldsymbol{R}_t'\in\boldsymbol{\mathcal{F}}_t$ and $\boldsymbol{R}_t\neq\boldsymbol{R}_t'$ is set as follows.

$$p(\boldsymbol{R}_t,\boldsymbol{R}_t')=\rho\cdot\exp\left(\frac{\gamma}{2}\left(G\left(\boldsymbol{R}_t'\right)-G\Big(\boldsymbol{R}_t\Big)\right)\right), \tag{11}$$

where ρ is a positive constant.

The designed Markov chain-based algorithm is described in Algorithm 2. In the algorithm, we randomly choose a state, i.e., a worker scheduling strategy $\boldsymbol{R}_t$ from the solution space. Then we constantly update the state according to the transition probabilities, thus forming a Markov chain, and iterate this process until the Markov chain converges. Note that during the iteration, the best strategy has been recorded.

When the Markov chain reaches the stationary distribution, or equivalently, satisfies the balance equation, we can get the optimal strategy. Recall that the optimality gap between $\boldsymbol{LSE-P4}$ and $\boldsymbol{P4}$ is bounded by $\frac{1}{\gamma}\log|\boldsymbol{\mathcal{F}}_t|$, we can set γ as large as possible to get a better solution. Assume that the algorithm achieves convergence within I_c iterations, we need to calculate $|\boldsymbol{\mathcal{F}}_t|$ transition probabilities in each iteration, then, the time complexity of Algorithm 2 is $\mathcal{O}(I_c|\boldsymbol{\mathcal{F}}_t|)$.

Algorithm 2 MPLP-Centralized (MPLP-C)

Input: $\mathcal{W}$, $\mathcal{T}$, A_t for time slot t, $\boldsymbol{\mathcal{F}}_t$, iteration number I_c.
Output: The optimal strategy $\boldsymbol{R}_t^*$ in time slot t.

```
R_t^* = ∅, and G(R_t^*) = 0;
Randomly select R_t form F_t;
while I_c > 0 do
    Calculate G(R_t);
    if G(R_t) > G(R_t^*) then
        R_t^* = R_t;
    end if
    Select a new strategy R_t' based on the transition probability (11);
    Update R_t by R_t';
    I_c = I_c − 1;
end while
return R_t^*;
```

IV. Performance Analysis

In this section, we analyze the convergence and approximation properties of Algorithm 1. It's easy to know that U_t is a bounded function due to constraints, for clarity, we let U_{min} and U_{max} be the upper and lower bounds of U_t on all time slots, respectively.

Theorem 1. *The solution obtained by Algorithm 1 meets constraints (7) and (8).*

Proof. Based on Lemma 4.6 in [11], we have $\Delta\left(\Theta(t)\right)\leq B+\Omega(t)$, where B is a positive constant value, thus we can get the following inequation:

$$V\cdot U_{max}-\Delta\left(\Theta(t)\right)\geq V\cdot U_{min}-B-\Omega(t). \tag{12}$$

As $\Delta(\Theta(t))\triangleq\mathbb{E}[L(\Theta(t+1))-L(\Theta(t))|\Theta(t)]$ and $L(\Theta(t))\triangleq\frac{1}{2}\sum_{i=1}^{m}Q_i^R(t)^2+\frac{1}{2}Q^C(t)^2$, then, taking the summation of both sides of (12) on $\mathcal{T}$, and combining with the Cauchy-Bunyakovsky-Schwarz inequality, we obtain:

$$\begin{aligned}&\left(\sum_{w_i\in\mathcal{W}}Q_i^R(T)+Q^C(T)\right)^2\leq\\&2T\left(B+V(U_{max}-V_{min})\right)+2\sum_{t=1}^{T}\Omega(t).\end{aligned} \tag{13}$$

Then, dividing both sides of (13) by T^2 and taking the square root of it, we have:

$$\begin{aligned}&\frac{\left(\sum_{w_i\in\mathcal{W}}Q_i^R(T)+Q^C(T)\right)}{T}\leq\\&\sqrt{\frac{2\left(B+V(U_{max}-V_{min})\right)}{T}+\frac{2\sum_{t=1}^{T}\Omega(t)}{T^2}}.\end{aligned} \tag{14}$$

As is proved in Theorem 4.8 in [11], all queues are mean rate stable, thus $\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}\Omega(t)$ has a constant upper bound. Then, taking expectations on both sides of (14) and letting T tend to infinity, we can obtain

$$\lim_{T\to\infty}\frac{\mathbb{E}\{\left(\sum_{w_i\in\mathcal{W}}Q_i^R(T)+Q^C(T)\right)\}}{T}\leq 0. \tag{15}$$

And because of $Q_i^R(T)\geq 0,\forall w_i\in\mathcal{W},Q^C(T)\geq 0$, we have $\lim\limits_{T\to\infty}\mathbb{E}\{Q_i^R(T)\}/T=0,\forall w_i\in\mathcal{W}$ and $\lim\limits_{T\to\infty}\mathbb{E}\{Q^C(T)\}/T=0.$ □

Let $\boldsymbol{R}_t^*$ be the optimal strategy for $\boldsymbol{P1}$ for time slot t, and $\boldsymbol{R}_t^p$ denote the strategy determined by Algorithm 1 and 2 in time slot t. Then, we have the following theorem.

Theorem 2. *For any $\delta > 0$ and positive control parameter $V \geq 0$, we have:*

$$U_{OPT} - \lim_{T\to\infty} \frac{1}{T} \sum\nolimits_{t=1}^{T} \mathbb{E}\{U_t(\boldsymbol{R}_t^p)\} \leq \frac{B'}{V} - \delta, \tag{16}$$

where $U_{OPT} = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\{U_t(\boldsymbol{R}_t^*)\}$, and $B' = B + \frac{1}{\gamma}\log|\boldsymbol{\mathcal{F}}_t|$. Recall that $\frac{1}{\gamma}\log|\boldsymbol{\mathcal{F}}_t|$ is the Markov approximation optimal gap.

Proof. Let's recall the model description in Section II. Platform utility on each time slot is related to the process of task arrival. According to Theorem 4.5 in [11], if the process of task arrival is stationary, then for any $\delta > 0$ we have:

$$U_{OPT} \leq \mathbb{E}\{U_t(\boldsymbol{R}_t^p)\} + \delta, \tag{17}$$

$$\mathbb{E}\{E_i^R(t)\} \leq \delta, \forall i \in \mathcal{W},\ \mathbb{E}\{E^C(t)\} \leq \delta. \tag{18}$$

Combine the Lemma 4.6 in [11], the following inequality can be obtained:

$$\begin{aligned} &V \cdot \mathbb{E}\{U(\boldsymbol{R}_t^p)\} - \Delta(\Theta(t)) \geq V \cdot \mathbb{E}\{U(\boldsymbol{R}_t^p)\} - B' \\ &- \Omega(t) \geq V \cdot (U_{OPT} + \delta) - B'. \end{aligned} \tag{19}$$

As $\Delta(\Theta(t)) \triangleq \mathbb{E}[L(\Theta(t+1)) - L(\Theta(t))|\Theta(t)]$, by summing the time slots t over $\mathcal{T}$ on both sides of equation (19) and rearranging the terms, we have $U_{OPT} - \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\{U_t(\boldsymbol{R}_t^p)\} \leq \frac{B'}{V} - \delta$. □

Theorem 3. *For any positive control parameter $V \geq 0$, the time average expected virtual queue satisfies:*

$$\limsup_{T\to\infty} \frac{1}{T} \sum\nolimits_{t=1}^{T} \mathbb{E}\{\|\Theta(t)\|_1\} \leq \frac{V \cdot (U_{max} - U_{min}) + B'}{\epsilon}. \tag{20}$$

Proof. Suppose $\exists \epsilon \geq 0$ such that for all time slot $t \in \mathcal{T}$ and all possible values of $\Theta(t)$, according to Theorem 4.2 in [11], we have:

$$V \cdot \mathbb{E}\{U_t|\Theta(t)\} - \Delta(\Theta(t)) \geq V \cdot U_{min} - B' + \epsilon \|\Theta(t)\|_1. \tag{21}$$

Then, as $\Delta(\Theta(t)) \triangleq \mathbb{E}[L(\Theta(t+1)) - L(\Theta(t))|\Theta(t)]$ and U_{max} is the upper bound of U_t. Summing over $\mathcal{T}$ on both sides of (21) and rearranging terms, we have:

$$\frac{1}{T} \sum\nolimits_{t=1}^{T} \mathbb{E}\{\|\Theta(t)\|_1\} \leq \frac{V \cdot (U_{max} - U_{min}) + B'}{\epsilon} + C, \tag{22}$$

where $C = \frac{\mathbb{E}\{L(\Theta(0))\}}{\epsilon T}$. The theorem can be proved by setting $T \to \infty$ in (22). □

Theorems 2 and 3 show that the gap between the utility obtained by MPLP-C and the optimal utility can be measured by $\mathcal{O}(1/V)$, and the size of the time average queue can be measured by $\mathcal{O}(V)$, which implies that we can adjust the control parameter V to achieve the balance between the optimal goal and the queue stability.

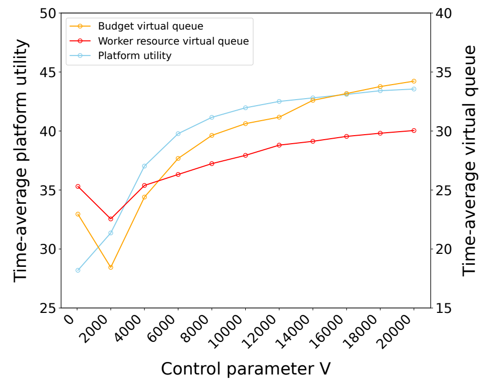
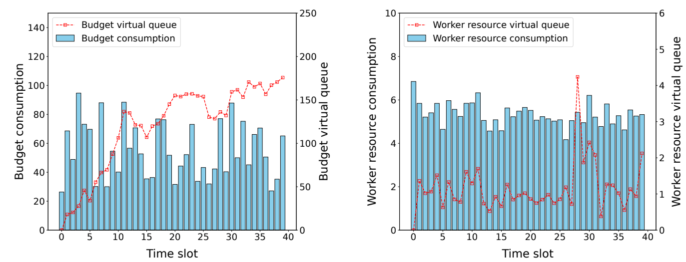


Fig. 1. Impact of control parameter V



(a) Virtual budget queues (b) Virtual resource queues

Fig. 2. Virtual queue analysis

V. Simulations

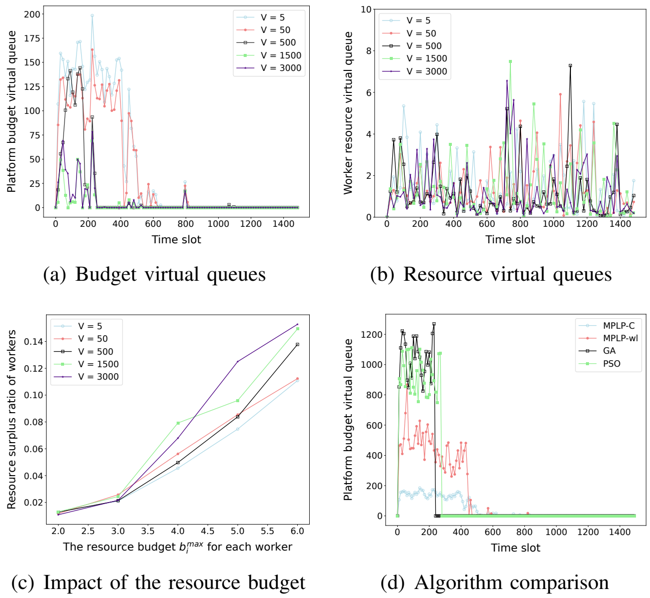
A. Experimental Settings

In the simulation experiment, we establish a total of $T = 1500$ slots and assume the presence of 25 crowdsourcing workers. The long-term resource constraint for each worker was randomly assigned within the range of [3, 7] per time slot, with a unit price of $\tau = 1$. The time-average budget for the platform to purchase worker resources is set to eighty percent of the total resources of the workers. The number of tasks received by the platform in each time slot t follows a Poisson distribution with $\lambda = 4|\mathcal{W}|$. For task a_j, the minimum resource $r_{j,t}^{min}$ required to execute is a random number on [0.5, 1.5]. For the parameter in the profit function of task a_j, α_j and β_j are randomly distributed in [2, 4] and [6, 8] respectively. b_i^{max} is a multiple of time-average resource, where the multiple is in the range of [2, 6]. We run the algorithm 100 times under each given setting, and the data points in our figures are the average results of 100 runs.

We compare MPLP-C with three algorithms: (1) the MPLP framework without b_i^{max} (MPLP-wl); (2) Particle swarm optimization (PSO) algorithm; (3) Genetic algorithm (GA).

B. Preformance Evaluation

Impact of control parameter V: From Fig. 1, we can see that the time-average platform utility increases as the control parameter V increases. The result matches the conclusion of Theorem 2, that is, the gap of the time-average platform utility between our algorithm and the optimal solution is limited by



(a) Budget virtual queues

(b) Resource virtual queues

(c) Impact of the resource budget

(d) Algorithm comparison

Fig. 3. Algorithm performance analysis

$\mathcal{O}(1/V)$. In addition, overall, the time-average virtual queue size of the platform and workers also increases as V increases, and the result matches the conclusion of Theorem 3, that is, the size of the time-average virtual queue is limited by $\mathcal{O}(V)$.

Role of virtual queues: As shown in Fig. 2(a), when the platform consumes a lot of budget to purchase worker resources in time slot t, the value of the budget virtual queue in time slot $t+1$ will increase. Due to the queue stability control of our algorithm, the budget consumption in time slot $t+1$ will be reduced accordingly. A similar result can be seen in Fig. 2(b), but we can see that the size of the workers' virtual queues does not change as dramatically over time as the platform's. The reason is that the workers also have a short-term resource budget constraint b_i^{max} in each time slot, which imposes a strict upper limit on the workers' resource consumption in each time slot.

The queue stability: As shown in Fig. 3(a), the platform budget virtual queue eventually converges to 0, although it fluctuates at t=800, the fluctuations are much smaller than those between t=0 and t=200, which is normal in the simulation. This indicates that long-term platform budget constraints can be effectively guaranteed. However, in Fig. 3(b), we observe that the worker resource virtual queue does not converge but exhibits quasi-periodicity, rapidly increasing and then quickly converging to 0, repeating the cycle. Fig. 3(c) shows that an increasing resource budget b_i^{max} of workers in a single time slot results in larger remaining worker resources.

Algorithm comparison: Stability is crucial in spatiotemporal crowdsourcing systems. Therefore, we compare the virtual queue variations across different algorithms. As shown in Fig. 3(d), the MPLP-C algorithm has a better virtual queue control effect than the MPLP-wl algorithm, which indicates that the short-term constraint plays a positive role in virtual queue control. Also, we can see that our algorithm is much better than algorithms PSO and GA in virtual queue control. The results suggest that our designs are more suitable for the long-term stable operation of online MCS systems.

VI. Conclusion

This paper investigates the online worker scheduling problem for spatiotemporal crowdsourcing systems, the objective is to maximize the platform's long-term utility with the long-term constraints of workers and the platform. To address the problem, we employ Lyapunov optimization techniques to decouple the long-term constraints and design a centralized algorithm based on Markov approximations to find solutions for each time slot. Extensive computer simulations have validated the efficacy and reliability of our designs.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (with grants No. 62202016 and No. 62202055).

References

[1] M. Lyu, X. Li, and C.-H. Chen, "Achieving knowledge-as-a-service in iiot-driven smart manufacturing: A crowdsourcing-based continuous enrichment method for industrial knowledge graph," *Advanced Engineering Informatics*, vol. 51, p. 101494, 2022.

[2] Q. Tao, Y. Tong, Z. Zhou, Y. Shi, L. Chen, and K. Xu, "Differentially private online task assignment in spatial crowdsourcing: A tree-based approach," in *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pp. 517–528, IEEE, 2020.

[3] 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, 2023.

[4] M. Wang, Y. Wang, A. Sai, *et al.*, "Task assignment for hybrid scenarios in spatial crowdsourcing: A q-learning-based approach," *Applied Soft Computing*, vol. 131, p. 109749, 2022.

[5] J. Zhang, T. Jiang, X. Gao, *et al.*, "An online fairness-aware task planning approach for spatial crowdsourcing," *IEEE Transactions on Mobile Computing*, vol. 23, no. 1, pp. 150–163, 2024.

[6] B. Zhao, H. Dong, Y. Wang, *et al.*, "Ppo-ta: Adaptive task allocation via proximal policy optimization for spatio-temporal crowdsourcing," *Knowledge-Based Systems*, vol. 264, p. 110330, 2023.

[7] W. Huang, P. Li, B. Li, *et al.*, "Towards stable task assignment with preference lists and ties in spatial crowdsourcing," *Information Sciences*, vol. 620, pp. 16–30, 2023.

[8] Q. Zhang, Y. Wang, G. Yin, *et al.*, "Two-stage bilateral online priority assignment in spatio-temporal crowdsourcing," *IEEE Transactions on Services Computing*, 2022.

[9] Z. Tang, F. Mou, J. Lou, W. Jia, Y. Wu, and W. Zhao, "Multi-user layer-aware online container migration in edge-assisted vehicular networks," *IEEE/ACM Transactions on Networking*, vol. 32, no. 2, pp. 1807–1822, 2024.

[10] B. Guo, Y. Liu, L. Wang, *et al.*, "Task allocation in spatial crowdsourcing: Current state and future directions," *IEEE Internet of Things Journal*, vol. 5, pp. 1749–1764, June 2018.

[11] M. Neely, *Stochastic network optimization with application to communication and queueing systems*. Springer Nature, 2022.

[12] C. Chekuri and S. Khanna, "A polynomial time approximation scheme for the multiple knapsack problem," *SIAM Journal on Computing*, vol. 35, no. 3, pp. 713–728, 2005.

[13] T. Ouyang, Z. Zhou, and X. Chen, "Follow me at the edge: Mobility-aware dynamic service placement for mobile edge computing," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 10, pp. 2333–2345, 2018.

[14] M. Chen, "Markov approximation for combinatorial network optimization," *IEEE Transactions on Information Theory*, vol. 59, no. 10, pp. 6301–6327, 2013.

[15] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.