IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 23, 2021, accepted February 1, 2021, date of publication February 11, 2021, date of current version February 19, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3058789

Profit Maximization in Mobile Crowdsourcing: A Competitive Analysis

QI LI[1], LIJUN CAI[1], HUANLE XU[2], AND TAO MENG[3]
[1]College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China
[2]Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong
[3]College of Computer and Information Engineering, Central South University of Forestry and Technology, Changsha 410004, China

Corresponding author: Lijun Cai (ljcai@hnu.edu.cn)

ABSTRACT Mobile Crowdsourcing, which uses a crowd of workers to complete computer-complexity tasks, has become more and more popular. How to maximize the total profit, which is the difference between total task utility and total worker costs, is a key design goal in such crowdsourcing system. However, this problem is extremely hard because even in the offline situation, this problem has been proved to be NP-hard. In the online situation, we need to consider not only future unknown task arrivals, but also the heterogeneity of both tasks and workers. In this paper, we address this challenging problem from following aspects. To solve this problem, we first build an optimization framework to formulate this problem by explicitly modelling task utility and worker costs. Then, under this framework, we design efficient online assignment algorithms to assign tasks to workers. To analyze the efficiency of our proposed algorithms, we develop a dual-fitting method to analyze both primal objective and dual objective. We prove that our designed algorithms can achieve a constant competitive ratio of 2. Finally, we conduct extensive trace-driven simulations to demonstrate that the online algorithms can outperform baseline algorithms by nearly 20% in terms of the total profit achieved.

INDEX TERMS Task utility, worker costs, online task assignments, dual-fitting method, competitive performance.

I. INTRODUCTION

Mobile crowdsourcing systems are becoming effective platforms for workers to perform tasks that are extremely difficult for computers. In Amazon Mechanical Turk [1], typical tasks include translating some paragraphs from Chinese into English, labelling contents of images, or collecting school addresses in a specific area. In mobile crowdsourcing, workers are assigned tasks and get paid for completing tasks. The payment of a task is also called *task utility*, which can be formulated by a utility function. In a utility function, the input is completed tasks and the output is utility. The task utility function can be considered as a concave function [2]. The cost of a worker can be formulated by a cost function. In a cost function, the input is also completed tasks and the output is a cost. We can model a cost function as a convex function, which has been adopted in [3]. The profit of a task is the difference between the utility achieved by completing the task and the cost incurred during processing the task, which is similar to the models built in existing works [2], [10], [30].

In this paper, we aim to address the problem of maximizing the total profit in an online manner, which is a key design goal in a crowdsourcing system. This is a very hard problem. The hardness stems from following four aspects. First, the workers and the tasks in the crowdsourcing are heterogeneous. Different tasks have different utility, which results the heterogeneity of tasks [4]. Different workers in the same system may have different prices [5]. Second, existing mechanisms mainly consider the scenario where all tasks belong to a monopolistic system. In this system, a worker cannot refuse task assignments. However, a worker can refuse task assignments to produce fail-assignments when the profit of a task cannot satisfy his requirement. Third, a task assignment decision needs to be made in an online manner, which turns out to be an integer programming problem [6]. With the whole knowledge of all task arrivals, it is NP-hard of assigning a task to a proper worker to maximize the total profit. Without knowing future arriving tasks, it is more difficult to assign a task to a proper worker to achieve this aim. Fourth, every task

The associate editor coordinating the review of this manuscript and approving it for publication was Daniel Grosu.

assignment will affect remaining task assignments, which is intractable to analyze the competitive performance of online algorithms [7].

A practical task assignment scheme for a crowdsourcing system to maximize its total profit should satisfy the following four requirements: scheduling tasks in an online manner, meeting the heterogeneity requirement of both tasks and workers, utility functions being concave, and cost functions being convex. At the best of our knowledge, no prior scheme meets all these requirements. The scheme proposed in [8] meets the requirement of heterogeneity of both tasks and workers. But it is an offline scheme. The scheme proposed in [9] is an online scheme. It also meets the requirements of the heterogeneity of both tasks and workers. But it uses a linear utility function. The scheme proposed in [2] meets the requirements of scheduling tasks in an online manner, considering the heterogeneity of both tasks and workers, utility functions being concave. But it does not consider convex cost functions. Note that a linear function is a special case of both its corresponding concave function and convex function. The scheme proposed in [10] meets the requirements of scheduling tasks in an online manner, the heterogeneity of workers. But this scheme does not consider the heterogeneity of tasks.

In this paper, we propose a task assignment scheme to maximize its total profit that meets the above four requirements. To achieve an online manner, we propose an online optimization framework to model the problem in slotted time. To meet the task heterogeneity requirement, we let each task having a unique utility function. To meet the worker heterogeneity requirement, we use different parameter to model the heterogeneity of each worker. We utilize the primal-dual approach to solve this optimization framework, which contains a concave utility function and a convex cost function. In this optimization problem, we rely on the previous assignments to design online algorithms. To solve this NP-hard problem, we transform the integer programming problem to a fraction programming problem, which uses available tasks and workers to construct a pseudo-tree. In the online algorithms, we design an objective value to search an appropriate task in the pseudo-tree. To analyze the competitive performance, we develop a dual-fitting approach to analyze online algorithms by splitting one optimization problem into two sub-problems.

In this paper, we have made the following contributions:

- We propose a novel optimization framework which meets the heterogeneity requirement of both tasks and workers for profit maximization in an online manner. Moreover, our optimization framework can handle a concave utility function and a convex cost function, which is processed by the primal-dual approach.
- We design two efficient online algorithms TAOA (Task Assignment Online Algorithm) which transform the integer programming problem into a fraction programming problem to solve the NP-hard problem. In these proposed algorithms, we use available tasks and workers to construct a pseudo-tree for appropriate task assignments.
- We analyze the competitive performance of online algorithms by a dual-fitting method, which can achieve a constant ratio 2. In addition, we examine the performance of online algorithms by extensive trace-driven simulations and these algorithms outperform baseline algorithms nearly 20% by real dataset.

II. RELATED WORK

In recent years, task assignments have been extensively studied in mobile crowdsourcing markets. For example, in [13]–[15], the utility of spatial crowdsourcing system is defined as the number of finished tasks, while in [16] it is defined as the reliability and diversity of finished tasks. In order to maximize the utility of all tasks, Tong and She [11] propose a Hungarian-based method called TGOA with competitive ratio $\frac{1}{4}$ in random order model. They also utilize a greedy-based method called TGOA-Greedy to achieve a competitive ratio of $\frac{1}{8}$ when tackling the same model [17]. Both [11] and [17] use a threshold-based method to maximize utility in bipartite matching [11] and trichromatic matching [17]. A threshold of utility is sampled beforehand and this method makes an assignment when task utility is above the threshold. Goel *et al.* [18] design an incentive-compatible mechanism to get an optimal utility under matching constraints.

Many studies have been done on task assignments about cost minimization in mobile crowdsourcing systems. Singer and Mittal [19] put forward a constant-competitive mechanism that improves assignment number under a budget and finds out a minimum total price. Cheng and Varma [20] put forward a new price plan and show the change of task remuneration, which has an effect on the number of workers willing to complete tasks. Bernstein *et al.* [21] use the queuing theory to analyze retainer model for optimizing total costs and show the performance of the real-time crowdsourcing platform. In particular, Mao and Humphrey [22] allocate resource to virtual machines within deadlines and minimum financial costs. Be consistent with others, [23] considers the problem which contains a fixed time-limit and a resource cost budget in a cloud environment.

In addition, there are many models to study the profit in mobile crowdsourcing systems. Xia [10] first establish a task reward pricing model with task temporal constraints (i.e., expected completion time and deadline). Xia *et al.* [24] develop a workload allocation policy that makes a reasonable tradeoff between worker utility and platform profit. Liu [8] design four incentive mechanisms to select workers to form a valid team and determine each individual worker's payment. In [30], Zhang et al model social welfare (profit) maximization problem using a primal-dual optimization framework. This framework provides richer structures for convex programs that guide the design of online auctions.

The primal-dual approach [25], [26] has been used to design online algorithms and auctions for various problems, such as ski rental problem and metrical job system

problem. As a classic tool for the design of approximation algorithms, the primal-dual approach has been successfully applied to online optimization problem with linear objectives [31]. The original primal-dual approach focuses on linear program and does not naturally model convex cost function. Recently, there are many studies about extending the primal-dual approach to solve the convex program problems, such as online scheduling on speed-scalable machines [27] and online combinatorial auctions with production costs [28]. The primal-dual approach with convex programs is also used for an online concave matching problem to get optimal competitive ratios [29].

There is a significant difference between a dual-fitting method and the primal-dual approach. A dual-fitting method is only used as a tool to analyze a given algorithm, while the primal-dual approach guides the design of optimization algorithm. The dual-fitting method is first developed by [32] and is widely used for the analysis of online algorithms [34]. In particular, [33], [34] address linear objectives and use the dual-fitting method to derive competitive bounds for traditional scheduling algorithms. In [35], the dual-fitting method is based on a non-convex program and Lagrangian duality to solve optimization problems. Anand *et al.* [37] give a more direct and interesting approach for analyzing online scheduling algorithms with resource augmentation. Reference [36] considers the general online covering and packing problems with convex objectives and shows online algorithms with competitive ratios. Agrawal and Devanur [38] uses the dual-fitting method to get near-optimal regret guarantees for online stochastic convex programming problems.

III. ONE TASK FOR ONE WORKER MODEL

As we know, a ridesharing platform (DiDi and Uber) is considered as a typical mobile crowdsourcing system. In this ridesharing platform, the available taxis in one area do not change often. The customer requests a taxi service by mobile devices and the system assigns this request to a taxi driver. At last, the customer will pay some money to this driver and this diver need consumes a certain amount of resource such as the gasoline consumption. In this model, we consider that workers are chosen from an available set [39] and arrive a mobile crowdsourcing system randomly [10]. We consider that time is slotted and task j enters the system at time a_j. Each task has a deadline d_j [11], which specifies the finished time. Each worker can only serve one task at any time slot and all workers are indexed by $[1, 2, \cdots, M]$. Upon arrival, multiple tasks join a global queue managed by the system and wait to be assigned to workers. Thus, we can make online task assignments in mobile crowdsourcing systems. After task j is processed by a worker and this worker will receive a utility. Furthermore, worker i charges for a cost when this worker is serving a task at time t. The cost is additive by time t such as the gasoline consumption. In this paper, our goal is to maximize the total profit, which is the difference between total utility and total costs. Our system

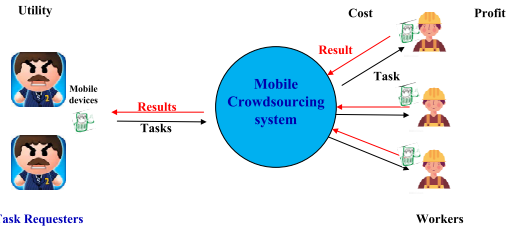


FIGURE 1. The mobile crowdsourcing system diagram.

TABLE 1. The notations of scheduling parameters.

Notation	Corresponding meaning
a_j	arrival time of task j
d_j	leaving time of task j
f_j	utility of task j
g_{it}	cost of worker i at time t
M	number of workers
N	number of tasks
c_j	constant value serves to heterogeneous tasks
r_i	constant value serves to heterogeneous workers
TAOAO	scheduling algorithm in the first model
TAOAM	scheduling algorithm in the second model
$x_j^i(t)$	a scheduling decision variable
X_i	capacity of worker i
α_j	dual variable for task utility in TAOAO
β_i	dual variable for worker cost in TAOAO
λ_j	dual variable for task utility in TAOAM
γ_i	dual variable for worker cost in TAOAM
P	objective value of primal problem at the end
D	objective value of dual problem at the end
P^*	objective value of optimal solutions
c	constant value to denote competitive ratio

diagram is shown in Fig.1 and all scheduling variables are summarized in Table 1.

A. UTILITY MAXIMIZATION

In this section, we consider each task has a utility function. In addition, task j has a concave utility function $f_j(\cdot)$ [2], which is inferred from the completed task by deadline d_j. $x_j^i(t)$ is a scheduling decision variable, which represents whether task j is served by worker i at time t. If worker i accept task j, $x_j^i(t)$ will keep 1 until task finished. c_j is served as a parameter for heterogeneous tasks. In this paper, we adopt a same type of the utility function in [40] and the utility maximization problem can be formulated as follows:

$$\max_{\{\boldsymbol{x}(t)\}} \quad \sum_{j=1}^{N} f_j\Big(\sum_{t=a_j}^{d_j}\sum_{i=1}^{M} x_j^i(t)\cdot c_j\Big) \tag{1a}$$

$$\text{s.t.} \quad \sum_{i=1}^{M} x_j^i(t) \le 1 \quad \forall j, t, \tag{1b}$$

$$a_j \le t \le d_j, \tag{1c}$$

$$x_j^i(t) \in \{0, 1\}, \quad \forall j, i, t. \tag{1d}$$

Here, the first constraint (1b) denotes that each task can only be assigned to one worker at any time slot. The second

constraint (1c) denotes that task j is not available beyond its availability window $[a_j, d_j]$.

Assumption 1: For all $j \in [1, 2, \cdots, N]$, c_j is known when task j enters to the crowdsourcing system.

B. COST MINIMIZATION

In this section, we consider workers are heterogeneous and each worker will incur a certain amount of cost. In addition, the cost of each worker is additive by time t [3]. r_i serves as a parameter to model the resource consumption on worker i. As such, minimizing overall costs yield the following optimization problem:

$$\min_{\{\boldsymbol{x}(t)\}} \quad \sum_{i=1}^{M}\sum_{t=0}^{\infty}\sum_{j=1}^{N} x_j^i(t)\cdot r_i \tag{2a}$$

$$s.t. \quad \sum_{j=1}^{N} x_j^i(t) \le 1, \quad \forall i, t, \tag{2b}$$

$$x_j^i(t) \in \{0, 1\}, \quad \forall j, i, t, \tag{2c}$$

$$t \in [0, \infty). \tag{2d}$$

The first constraint (2b) denotes that each worker can only process one task at any time slot. The second constraint (2d) states that worker i is available all the time until leaving the system.

Assumption 2: For all $i \in [1, 2, \cdots, M]$, r_i is known to the crowdsourcing system.

C. OPTIMIZATION PROBLEM FOR PROFIT MAXIMIZATION

In this section, we focus on maximizing the total profit, which is defined as the difference between total utility and total costs. Define $\boldsymbol{x}_j(t) = \{x_j^1(t), x_j^2(t), \cdots, x_j^M(t)\}$ and $\boldsymbol{x}(t) = \{\boldsymbol{x}_1(t), \boldsymbol{x}_2(t), \cdots, \boldsymbol{x}_N(t)\}$ as the vector of scheduling decision variables. In our model, one worker will be assigned one task, which cannot exceed worker's capacity. Moreover, the time constrain of utility function is inconsistent with the time constrain of cost function. Thus, we use $x_j^i(t) = 0$ and $t \notin [a_j, d_j]$ to replace $a_j \le t \le d_j$ in the utility function. In this paper, we aim to maximize the total profit by determining $\boldsymbol{x}(t)$. We formulate the profit optimization problem P1 as follows:

$$\max_{\{\boldsymbol{x}(t)\}} \sum_{j=1}^{N} f_j\Big(\sum_{t=0}^{\infty}\sum_{i=1}^{M} x_j^i(t)\cdot c_j\Big) - \sum_{i=1}^{M}\sum_{t=0}^{\infty}\sum_{j=1}^{N} x_j^i(t)\cdot r_i$$

$$\text{s.t. } x_j^i(t) = 0, t \notin [a_j, d_j], \forall j, i, t,$$

$$\text{Eq. (1b), (1d), (2b) are satisfied.} \tag{P1}$$

D. COMPETITIVE PERFORMANCE METRICS

At the beginning of each time-slot, the system needs to decide which tasks can be scheduled without knowing future task arrivals. This scheduling problem turns out to be an integer programming problem, which is NP-hard even for a linear utility function [30]. As such, we use an approximation online algorithm to solve this integer programming problem in P1. The following definition characterizes the competitive performance of an algorithm by an approximation ratio c.

Definition 1: An algorithm is c-competitive if the algorithm's objective P is within a factor of c to the optimal solution's objective P^*, i.e., $P^* \le c \cdot P$.

In our model, the smaller approximation ratio is, the better solution is. For the reason that this approximation solution is closer to the optimal solution.

IV. ONLINE DUAL-FITTING OPTIMIZATION FRAMEWORK

In the primal problem P1, the profit reveals only when a task leaves a worker. Moreover, the online decision for the task assignment should be made before the task is executed. As such, it is intractable to solve P1 directly in an online manner. To deal with this issue, we adopt the primal-dual approach to design approximation online algorithms. Observe that the two constraints in P1 are linear. We let $\boldsymbol{\alpha} = (\alpha_j | j = 1, 2, \cdots, N)$ and $\boldsymbol{\beta} = (\beta_i | i = 1, 2, \cdots, M)$ denote Lagrangian dual variables corresponding to the first and second constraints respectively. Thus, the dual problem of P1 can be formulated as:

$$\min_{\boldsymbol{\alpha}\ge\mathbf{0},\boldsymbol{\beta}\ge\mathbf{0}} \max_{\boldsymbol{p}} \quad \Phi(\boldsymbol{p}, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

$$\text{s.t. } x_j^i(t) \in \{0, 1\}, \tag{D1}$$

where $\Phi(\boldsymbol{p}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ is given by:

$$\begin{aligned}
&\Phi(\boldsymbol{p},\boldsymbol{\alpha},\boldsymbol{\beta}) \\
&= \sum_{j=1}^{N} f_j\Big(\sum_{t=0}^{\infty}\sum_{i=1}^{M} x_j^i(t)\cdot c_j\Big) - \sum_{i=1}^{M}\sum_{t=0}^{\infty}\sum_{j=1}^{N} x_j^i(t)\cdot r_i \\
&\quad - \sum_{j=1}^{N}\sum_{t=0}^{\infty} \alpha_j(t)\cdot\Big(1 - \sum_{i=1}^{M} x_j^i(t)\Big) \\
&\quad + \sum_{i=1}^{M}\sum_{t=0}^{\infty} \beta_i(t)\cdot\Big(1 - \sum_{j=1}^{N} x_j^i(t)\Big).
\end{aligned} \tag{3}$$

We split the dual problem D1 into optimization problem T and optimization problem M. T is used to yield an approximation algorithm and M is used to update dual variables from the task assignment. Thus, we get the following optimization problems:

$$\min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \quad T + M$$

$$where \quad T = \max\Big\{\sum_{j=1}^{N}\sum_{i=1}^{M}\sum_{t=0}^{\infty} x_j^i(t)\cdot(\alpha_j(t) - \beta_i(t))\Big\},$$

$$s.t. \quad \alpha_j(t) \ge 0, \ \forall j, t, \qquad \beta_i(t) \ge 0, \ \forall i, t,$$

$$T \ge 0, \ \forall i, j, t,$$

$$x_j^i(t) = 0, t \notin [a_j, d_j], \forall j, i, t,$$

$$\text{Eq. (1b), (1d), (2b) are satisfied.} \tag{D2}$$

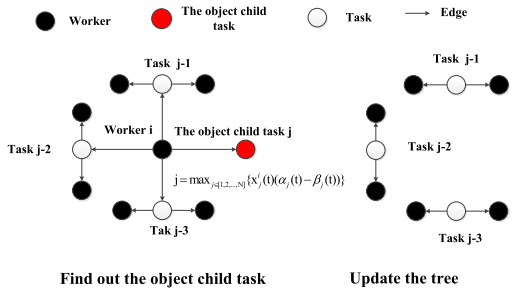


FIGURE 2. The pseudo-tree for One task for one worker.

M is shown as:

$$M = \max_{\boldsymbol{\alpha}\geq 0,\boldsymbol{\beta}\geq 0} \Big\{ \sum_{j=1}^{N} f_j\Big(\sum_{t=0}^{\infty}\sum_{i=1}^{M} x_j^i(t)\cdot c_j\Big) - \sum_{i=1}^{M}\sum_{t=0}^{\infty}\sum_{j=1}^{N} x_j^i(t)\cdot r_i - \Big(\sum_{j=1}^{N}\sum_{t=0}^{\infty}\alpha_j(t) - \sum_{i=1}^{M}\sum_{t=0}^{\infty}\beta_i(t)\Big)\Big\} \tag{4}$$

A. DESIGN APPROXIMATION ALGORITHM

It turns out that, the task utility function is more than a sum of multiple timely-dependent functions. We illustrate this property via the following lemma.

Lemma 1: There exist a function sequence $v_j(t)$ *such that* $f_j(\sum_{t=a_j}^{d_j}\sum_{i=1}^{M} x_j^i(t)\cdot c_j) \geq \sum_{t=a_j}^{d_j} v_j(t)\cdot \sum_{i=1}^{M} x_j^i(t)\cdot c_j$.

where $v_j(t) = \dfrac{df_j\Big(\sum_{i=1}^{M}\sum_{\tau=a_j}^{t} x_j^i(\tau)\cdot c_j\Big)}{d\Big(\sum_{i=1}^{M}\sum_{\tau=a_j}^{t} x_j^i(\tau)\cdot c_j\Big)}$.

Readers may refer to Appendix VII-A for the detailed proof of Lemma 1. By Lemma 1, we can solve this optimization problem in a time-slot.

The optimization problem T can be solved in each individual time slot, by doing this, we can determine whether to allocate a task to a worker in an online manner. The scheduling decision variables need to satisfy $\sum_{i=1}^{M} x_j^i(t) \leq 1$ and $\sum_{j=1}^{N} x_j^i(t) \leq 1$, which yields the following optimization problem:

$$\begin{aligned} \max_{\{\boldsymbol{x}(t)\}} \ & \sum_{j=1}^{N}\sum_{i=1}^{M} x_j^i(t)\cdot\Big(\alpha_j(t)-\beta_i(t)\Big) \\ s.t. \quad & \alpha_j(t)\geq 0, \quad \forall j,t, \qquad \beta_i(t)\geq 0, \quad \forall i,t, \\ & \alpha_j(t) > \beta_i(t) \quad \forall j,i,t, \\ & \text{Eq. (1b), (1d), (2b) are satisfied.} \end{aligned} \tag{D3}$$

To solve the integer programming problem in D3, a common method is to relax $x_j^i(t)$ as many continuous variables, i.e., $x_j^i(t) \in [0,1]$. Moreover, the sum of these continuous variables is 1. After this, we round the fractional value to an integer. Similar to the method developed in existing work [12], the system uses tasks and workers to construct a pseudo-tree in Fig.2. The worker is considered as a root node and tasks are regarded as leaf nodes. If $x_j^i(t) > 0$, there is an edge between worker i and task j. Then, the system finds out a worker's child task following $j = \mathbf{max}_{j\in[1,2,\cdots,N]}\Big\{x_j^i(t)\cdot(\alpha_j(t)-\beta_i(t))\Big\}$ and checks whether the constraint $\alpha_j(t) > \beta_i(t)$ is satisfied. If the condition is satisfied, the system sets $x_j^i(t) = 1$ and assigns task j to worker i. Then, task j and worker i will be removed from the constructed tree. If this constraint is violated, the system rejects to assign task j to worker i and removes the edge between them. Moreover, the failed-assignment task will be allocated to another worker by constructing a new edge at time t.

It remains to update the dual variables in each time slot, to achieve this, we adopt the primal-dual approach. After solving D3 and making scheduling decisions in time slot $t-1$, we can get $v_{j-1}(t-1)\cdot c_{j-1}$ and r_i. We substitute $\alpha_j(t) = v_{j-1}(t-1)\cdot c_{j-1}$ and $\beta_i(t) = r_i$ into D3. Then, we can make the task assignment in time t and calculate $v_j(t)$ using $x_j^i(t)$. We update next dual variables by applying $\alpha_j(t+1) = v_j(t)\cdot c_j$ and $\beta_i(t+1) = r_i$. By these updated dual variables, the system can determine the task assignment in time $t+1$ on worker i. The optimal solution of D3 can be derived efficiently with complexity of at most $\mathcal{O}(N)$ by solving the corresponding KKT equations [43]. The corresponding pseudo-code of TAOAO is shown in Algorithm 1.

Algorithm 1 Task Assignment Online Algorithm to One Task for One Worker Model

Input: c_{j-1}, c_j, r_i, $v_{j-1}(t-1)$;
Output: $x_j^i(t)$, $v_j(t)$;
Initialize : $x_j^i(t) = 1$ and caclulate $v_j(t)$;
When tasks and workers arrive at the system, the system utilizes tasks and workers to construct a pseudo-tree;
Update dual variables$\alpha_j(t) = v_{j-1}(t-1)\cdot c_{j-1}$ and$\beta_i(t) = r_i$;
while tasks are available **do**
    Find a child task j for worker i which is satisfied $j = \mathbf{max}_{j\in[1,2,\cdots,N]}\Big\{x_j^i(t)\cdot(\alpha_j(t)-\beta_i(t))\Big\}$;
    if $\alpha_j(t) > \beta_i(t)$ **then**
        Assign task j to worker i;
        $x_j^i(t) = 1$;
        Calculate $v_j(t)$, utility and cost;
        Remove task j and worker i from the constructed tree;
    else
        $x_j^i(t) = 0$, remove the edge between worker i and task j;
    end if
end while

B. COMPETITIVE PERFORMANCE ANALYSIS

In this section, we analyze the competitive performance of TAOAO by a dual-fitting method. Let P and D denote the

primal and dual objective values respectively. Based on the primal problem P1, the primal objective value P can be denoted by:

$$P = \sum_{j=1}^{N} f_j\Big(\sum_{t=0}^{\infty}\sum_{i=1}^{M} x_j^i(t)\cdot c_j\Big) - \sum_{i=1}^{M}\sum_{t=0}^{\infty}\sum_{j=1}^{N} x_j^i(t)\cdot r_i. \quad (5)$$

Applying weak duality theory, we have, the dual objective value D is more than the optimal solution's objective value P^*, i.e., $P^* \le D$ [26]. Thus, we can use the dual objective value to set up performance analysis. By the dual problem D2, D can be expressed by:

$$\begin{aligned} D &= \sum_{j=1}^{N}\sum_{i=1}^{M}\sum_{t=0}^{\infty} x_j^i(t)\Big(\alpha_j(t) - \beta_i(t)\Big) \\ &\quad + \Big(\sum_{j=1}^{N} f_j\Big(\sum_{t=0}^{\infty}\sum_{i=1}^{M} x_j^i(t)\cdot c_j\Big) - \sum_{i=1}^{M}\sum_{t=0}^{\infty}\sum_{j=1}^{N} x_j^i(t)\cdot r_i\Big) \\ &\quad - \sum_{j=1}^{N}\sum_{t=0}^{\infty}\alpha_j(t) + \sum_{i=1}^{M}\sum_{t=0}^{\infty}\beta_i(t). \end{aligned} \quad (6)$$

Following the primal-dual approach, we update $\alpha_j(t) = v_{j-1}(t-1)\cdot c_{j-1}$ and $\beta_i(t) = r_i$. At the end, we can get $\sum_{j=1}^{N}\sum_{i=1}^{M}\sum_{t=0}^{\infty}\alpha_j(t) = \sum_{j=1}^{N}\sum_{i=1}^{M}\sum_{t=0}^{\infty} v_j(t)\cdot c_j$ by ignoring the last one. Thus, the value of T can be given by:

$$\begin{aligned} T &= \sum_{j=1}^{N}\sum_{i=1}^{M}\sum_{t=0}^{\infty} x_j^i(t)\Big(\alpha_j(t) - \beta_i(t)\Big) \\ &= \sum_{j=1}^{N}\sum_{i=1}^{M}\sum_{t=0}^{\infty} x_j^i(t)\Big(v_j(t)\cdot c_j - r_i\Big) \end{aligned} \quad (7)$$

Considering lemma (1), we have $f_j(\sum_{t=0}^{\infty}\sum_{i=1}^{M} x_j^i(t)\cdot c_j) \ge \sum_{t=0}^{\infty} v_j(t)\cdot\sum_{i=1}^{M} x_j^i(t)\cdot c_j$. Thus, we can get:

$$\begin{aligned} T &= \sum_{j=1}^{N}\sum_{i=1}^{M}\sum_{t=0}^{\infty} x_j^i(t)\Big(v_j(t)\cdot c_j - r_i\Big) \\ &\le \sum_{j=1}^{N} f_j\Big(\sum_{t=0}^{\infty}\sum_{i=1}^{M} x_j^i(t)\cdot c_j\Big) - \sum_{j=1}^{N}\sum_{i=1}^{M}\sum_{t=0}^{\infty} x_j^i(t)\cdot r_i \end{aligned} \quad (8)$$

Therefore, we have $T \le P$.

By the constraint $\alpha_j(t) > \beta_i(t)$ for every assignment, we can get $\sum_{j=1}^{N}\sum_{t=0}^{\infty}\alpha_j(t) - \sum_{i=1}^{M}\sum_{t=0}^{\infty}\beta_i(t) > 0$. Thus, the value of Q is guaranteed by,

$$\begin{aligned} Q &= \sum_{j=1}^{N} f_j\Big(\sum_{t=0}^{\infty}\sum_{i=1}^{M} x_j^i(t)\cdot c_j\Big) - \sum_{i=1}^{M}\sum_{t=0}^{\infty}\sum_{j=1}^{N} x_j^i(t)\cdot r_i \\ &\quad - \sum_{j=1}^{N}\sum_{t=0}^{\infty}\alpha_j(t) + \sum_{i=1}^{M}\sum_{t=0}^{\infty}\beta_i(t) \\ &< \sum_{j=1}^{N} f_j\Big(\sum_{t=0}^{\infty}\sum_{i=1}^{M} x_j^i(t)\cdot c_j\Big) - \sum_{i=1}^{M}\sum_{t=0}^{\infty}\sum_{j=1}^{N} x_j^i(t)\cdot r_i \\ &= P. \end{aligned} \quad (9)$$

Thus, we can get $Q < P$.

Combine Eq(8) and Eq(9), the primal objective value P and the dual objective value D satisfy:

$$D = T + Q < 2P \quad (10)$$

We can conclude that $P \le P^* \le D < 2\cdot P$ and $c = 2$ where P^* is the objective value of optimal solutions. TAOAO is a 2-competitive algorithm with respect to the total profit.

V. MULTIPLE TASKS FOR ONE WORKER MODEL

In crowdsourcing system, multiple passengers can carpool a car at the same time by Shun Feng Che. In this section, we consider a worker can serve multiple tasks simultaneously [41]. Upon arrival, multiple tasks join a global queue managed by the system and wait to be assigned to a worker at the beginning of each time slot. If a task is satisfied with the worker's capacity constraint, it will be assigned to the worker. Then, the system will use this capacity constraint to find out another task which can also be assigned to the worker. When one task is completed, this worker proceeds to process unfinished tasks. In this model, our goal is also to maximize the total profit.

A. COST MINIMIZATION

In this model, r_i is a constant that serves as a parameter to characterize heterogeneous task consumption on worker i. For all $i \in [1, 2, \cdots, M]$, r_i is known to the crowdsourcing system. Moreover, we assume that multiple tasks can be run in parallel on worker i. The sum of r_i cannot exceed the worker's capacity X_i. The cost function of each worker is additive by time t. We formulate cost function as a convex function similar to [3]. As such, minimizing overall costs yield the following optimization problem:

$$\min_{\{\boldsymbol{x}(t)\}} \quad \sum_{i=1}^{M}\sum_{t=0}^{\infty} g_{it}\Big(\sum_{j=1}^{N} x_j^i(t)\cdot r_i\Big) \quad (11a)$$

$$s.t. \quad \sum_{j=1}^{N} x_j^i(t)\cdot r_i \le X_i, \quad \forall i, \quad (11b)$$

$$x_j^i(t) \in \{0, 1\}, \quad \forall j, i, t. \quad (11c)$$

B. OPTIMIZATION PROBLEM FOR PROFIT MAXIMIZATION

In this section, we focus on maximizing the total profit, which is defined as the difference between total utility and total costs. Similar to section III-D, our profit maximization problem is an integer optimization programming problem, which is NP-hard. Paralleling Section III-A, we utilize Eq(1) to denote the utility function. In this paper, we aim to maximize the total profit by determining $\boldsymbol{x}(t)$, which yields the

optimization problem P2, as shown below:

$$\begin{aligned}
\max_{(\boldsymbol{x}(t))} \ & \sum_{j=1}^{N} f_j\Big(\sum_{t=0}^{\infty}\sum_{i=1}^{M} x_j^i(t)\cdot c_j\Big) - \sum_{i=1}^{M}\sum_{t=0}^{\infty} g_{it}\Big(\sum_{j=1}^{N} x_j^i(t)\cdot r_i\Big) \\
\text{s.t.} \ & \sum_{i=1}^{M} x_j^i(t) \le 1, \quad \forall j, t, \\
& \sum_{j=1}^{N} x_j^i(t)\cdot r_i \le X_i, \quad \forall i, \\
& x_j^i(t) = 0, t \notin [a_j, d_j], \\
& \text{Eq. (1d) are satisfied.}
\end{aligned} \tag{P2}$$

C. ONLINE DUAL-FITTING OPTIMIZATION FRAMEWORK

Based on the dual problem, we can design approximation algorithms and conduct performance analysis. We adopt the primal-dual approach to solve this dual problem. Observe that the two constraints in P2 are linear. We let $\boldsymbol{\lambda} = (\lambda_j | j = 1, 2, \cdots, N)$,$\boldsymbol{\gamma} = (\gamma_i | i = 1, 2, \cdots, M)$ denote the Lagrangian dual variables corresponding to the first and second constraints respectively. The dual problem of P2 can be formulated as:

$$\begin{aligned}
\min_{\boldsymbol{\lambda}\ge\mathbf{0},\boldsymbol{\gamma}\ge\mathbf{0}} \max_{\boldsymbol{p}} \quad & \Phi(\boldsymbol{p}, \boldsymbol{\lambda}, \boldsymbol{\gamma}) \\
\text{s.t. } & x_j^i(t) \in \{0, 1\},
\end{aligned} \tag{D4}$$

where $\Phi(\boldsymbol{p}, \boldsymbol{\lambda}, \boldsymbol{\gamma})$ is given by:

$$\begin{aligned}
\Phi(\boldsymbol{p}, \boldsymbol{\lambda}, \boldsymbol{\gamma}) = & \sum_{j=1}^{N} f_j\Big(\sum_{t=0}^{\infty}\sum_{i=1}^{M} x_j^i(t)\cdot c_j\Big) \\
& - \sum_{i=1}^{M}\sum_{t=0}^{\infty} g_{it}\Big(\sum_{j=1}^{N} x_j^i(t)\cdot r_i\Big) \\
& - \sum_{j=1}^{N}\sum_{t=0}^{\infty} \lambda_j(t)\Big(1 - \sum_{i=1}^{M} x_j^i(t)\Big) \\
& + \sum_{i=1}^{M}\sum_{t=0}^{\infty} \gamma_i(t)\Big(X_i - \sum_{j=1}^{N} x_j^i(t)\cdot r_i\Big).
\end{aligned} \tag{12}$$

We split the dual problem D4 into optimization problem A and optimization problem B. A is utilized to yield an approximation algorithm and B is used to update the dual variables from the task assignment. Thus, we can get the following optimization problems:

$$\begin{aligned}
\min_{\boldsymbol{\lambda},\boldsymbol{\gamma}} \quad & A + B \\
\textit{where} \quad & A = \max\Big\{\sum_{t=0}^{\infty}\sum_{j=1}^{N}\sum_{i=1}^{M} x_j^i(t)(\lambda_j(t) - \gamma_i(t)\cdot r_i)\Big\}, \\
\text{s.t.} \quad & \lambda_j(t) \ge 0, \ \forall j, t, \qquad \gamma_i(t) \ge 0, \ \forall i, t, \\
& \lambda_j(t) > \sum_{j=1}^{N} \gamma_i(t)\cdot r_i, \ \forall i, j, t,
\end{aligned}$$

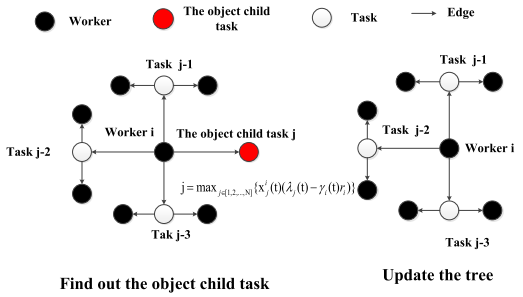


FIGURE 3. The pseudo-tree for multiple tasks for one worker model.

$$\begin{aligned}
& x_j^i(t) = 0, t \notin [a_j, d_j], \\
& \text{Eq. (1b), (1d), (11b) are satisfied.}
\end{aligned} \tag{D5}$$

B is shown as:

$$\begin{aligned}
B = \max_{\boldsymbol{\lambda}\ge\mathbf{0},\boldsymbol{\gamma}\ge\mathbf{0}} \Big\{ & \sum_{j=1}^{N} f_j\Big(\sum_{t=0}^{\infty}\sum_{i=1}^{M} x_j^i(t)\cdot c_j\Big) \\
& - \sum_{i=1}^{M}\sum_{t=0}^{\infty} g_{it}\Big(\sum_{j=1}^{N} x_j^i(t)\cdot r_i\Big) \\
& - \sum_{t=0}^{\infty}\Big(\sum_{j=1}^{N}\lambda_j(t) - \sum_{i=1}^{M}\gamma_i(t)\cdot X_i\Big)\Big\}
\end{aligned} \tag{13}$$

D. DESIGN APPROXIMATION ALGORITHM

In this section, we design approximation algorithms based on the primal-dual approach. It turns out that, the worker cost function is less than a sum of multiple functions. We illustrate this property as follows.

Lemma 2: There exist a function sequence $o_i(t)$ such that $g_{it}(\sum_{j=1}^{N} x_j^i(t)\cdot r_i) \le \sum_{j=1}^{N} o_i(t)\cdot x_j^i(t)\cdot r_i$.

where $o_i(t) = \dfrac{dg_{it}\Big(\sum_{\kappa=1}^{j} x_\kappa^i(t)\cdot r_i\Big)}{d\Big(\sum_{\kappa=1}^{j} x_\kappa^i(t)\cdot r_i\Big)}$.

Readers may refer to Appendix VII-B for the proof of Lemma 2. By Lemma 1 and Lemma 2, we can solve this optimization problem in a time-slot.

Based on the solution of A, we can determine whether to allocate a task to a worker in each time slot. In our model, scheduling decision variables need to satisfy $\sum_{i=1}^{M} x_j^i(t) \le 1$ and $\sum_{j=1}^{N} x_j^i(t)\cdot r_i \le X_i$, which yields the following optimization problem:

$$\begin{aligned}
\max_{\{\boldsymbol{x}(t)\}} \ & \sum_{j=1}^{N}\sum_{i=1}^{M} x_j^i(t)\Big(\lambda_j(t) - \gamma_i(t)\cdot r_i\Big) \\
s.t. \ & \sum_{j=1}^{N} \gamma_i(t)\cdot r_i < \lambda_j(t), \quad \forall j, i, t, \\
& \sum_{j=1}^{N} x_j^i(t)\cdot r_i \le X_i, \ \forall i,
\end{aligned}$$

$$x_j^i(t) = 0, t \notin [a_j, d_j],$$
$$\text{Eq. (1b), (1d), (11b) are satisfied.} \quad \text{(D6)}$$

To solve the integer programming problem in D6, a common method is to relax $x_j^i(t)$ as many continuous variables, i.e., $x_j^i(t) \in [0, 1]$. Moreover, the sum of these continuous variables is 1. After this, we round the fractional value to an integer. Similar to the method developed in existing work [12], the system utilizes tasks and workers to construct a pseudo-tree in Fig.3. Then, we update the dual variables by $\lambda_j(t) = v_{j-1}(t-1) \cdot c_{j-1}$ and $\gamma_i(t) = o_i(t-1)$. The system proceeds to find out a worker's child task following $j = \mathbf{max}_{j\in[1,2,\cdots,N]} \left\{x_j^i(t) \cdot (\lambda_j(t) - \gamma_i(t) \cdot r_i)\right\}$ and checks whether $X_i \geq \sum_{j=1}^{N} x_j^i(t) \cdot r_i$ is satisfied. If this condition is satisfied, the system will assign task j to worker i and remove task j from the constructed tree. Then, the system searches another task which can also be assigned to worker i at time t. If this constraint is violated, the system rejects task j and removes the edge between them. Moreover, the failed-assignment task will be allocated to another worker by constructing a new edge at time t. At last, we calculate $v_j(t) \cdot c_j$ and $o_i(t)$ for the next assignment. The optimal solution of D6 can be derived efficiently with complexity of at most $\mathcal{O}(N)$ by solving the corresponding KKT equations [43]. The pseudo-code of TAOAM is shown in Algorithm 2.

Algorithm 2 Task Assignment Online Algorithm to Multiple Tasks for One Worker Model

Input: $c_{j-1}, c_j, r_i, v_{j-1}(t-1)$;
Output: $c_j, c_{j-1}, r_i, X_i, v_{j-1}(t-1), o_i(t-1)$;
Initialize : $x_j^i(t) = 1$ and caclulate $v_j(t), o_i(t)$;
When tasks and workers arrive at the system, the system utilizes tasks and workers to construct a pseudo-tree;
Update dual variables $\lambda_j(t) = v_{j-1}(t-1) \cdot c_{j-1}$ and $\gamma_i(t) = o_i(t-1)$;
while tasks are available **do**
    Find a child task j for worker i which is satisfied $j = \mathbf{max}_{j\in[1,2,\cdots,N]} \left\{x_j^i(t)(\lambda_j(t) - \gamma_i(t) \cdot r_i)\right\}$;
    if $\lambda_j(t) > \sum_{j=1}^{N} \gamma_i(t) \cdot r_i X_i \geq \sum_{j=1}^{N} x_j^i(t) \cdot r_i$ **then**
        Assign task j to worker i;
        $x_j^i(t) = 1$;
        Calculate $v_j(t)$, $o_i(t)$, utility and cost;
        Update dual variables $\lambda_j(t)$ and $\gamma_i(t)$;
        Remove task j from the constructed tree;
    else
        $x_j^i(t) = 0$, Reject task j for worker i
    end if
end while

E. COMPETITIVE PERFORMANCE ANALYSIS

In this section, we analyze the competitive performance of TAOAM by a dual-fitting method. Let P and D denote the primal and dual objective values at the end. We first infer from the primal problem P2, the primal objective value P can be expressed by:

$$P = \sum_{j=1}^{N} f_j\Big(\sum_{t=0}^{\infty}\sum_{i=1}^{M} x_j^i(t) \cdot c_j\Big) - \sum_{i=1}^{M}\sum_{t=0}^{\infty} g_{it}\Big(\sum_{j=1}^{N} x_j^i(t) \cdot r_i\Big). \quad (14)$$

By dual problem D5, the dual objective value D can be denoted by:

$$\begin{aligned} D = &\sum_{t=0}^{\infty}\sum_{j=1}^{N}\sum_{i=1}^{M} x_j^i(t)\Big(\lambda_j(t) - \gamma_i(t) \cdot r_i\Big) \\ &+ \sum_{j=1}^{N} f_j\Big(\sum_{t=0}^{\infty}\sum_{i=1}^{M} x_j^i(t) \cdot c_j\Big) - \sum_{i=1}^{M}\sum_{t=0}^{\infty} g_{it}\Big(\sum_{j=1}^{N} x_j^i(t) \cdot r_i\Big) \\ &- \Big(\sum_{j=1}^{N}\sum_{t=0}^{\infty} \lambda_j(t) - \sum_{i=1}^{M}\sum_{t=0}^{\infty} \gamma_i(t) \cdot X_i\Big) \end{aligned} \quad (15)$$

Following the primal-dual approach, we update $\lambda_j(t) = v_{j-1}(t-1) \cdot c_{j-1}$ and $\gamma_i(t) = o_i(t-1)$. For worker i, we can get $\sum_{j=1}^{N}\sum_{t=0}^{\infty} \lambda_j(t) = \sum_{j=1}^{N}\sum_{t=0}^{\infty} v_j(t) \cdot c_j$ and $\sum_{t=0}^{\infty} \gamma_i(t) \cdot r_i = \sum_{t=0}^{\infty} o_i(t) \cdot r_i$ by ignoring the last one. Thus, at the end, the value of A can be given by:

$$\begin{aligned} A &= \sum_{t=0}^{\infty}\sum_{j=1}^{N}\sum_{i=1}^{M} x_j^i(t)\Big(\lambda_j(t) - \gamma_i(t) \cdot r_i\Big) \\ &= \sum_{t=0}^{\infty}\sum_{j=1}^{N}\sum_{i=1}^{M} x_j^i(t)\Big(v_j(t) \cdot c_j - o_i(t) \cdot r_i\Big) \end{aligned} \quad (16)$$

Consider the lemma (1) and the lemma (2), we have $f_j(\sum_{t=0}^{\infty}\sum_{i=1}^{M} x_j^i(t) \cdot c_j) \geq \sum_{t=0}^{\infty}\sum_{i=1}^{M} v_j(t) \cdot x_j^i(t) \cdot c_j$ and $g_{it}(\sum_{j=1}^{N} x_j^i(t) \cdot r_i) \leq \sum_{j=1}^{N} o_i(t) \cdot x_j^i(t) \cdot r_i$. Thus, we can get:

$$\begin{aligned} A &= \sum_{t=0}^{\infty}\sum_{j=1}^{N}\sum_{i=1}^{M} x_j^i(t)\Big(v_j(t) \cdot c_j - o_i(t) \cdot r_i\Big) \\ &\leq \sum_{j=1}^{N} f_j\Big(\sum_{t=0}^{\infty}\sum_{i=1}^{M} x_j^i(t) \cdot c_j\Big) - \sum_{i=1}^{M}\sum_{t=0}^{\infty} g_{it}\Big(\sum_{j=1}^{N} x_j^i(t) \cdot r_i\Big) \end{aligned} \quad (17)$$

Therefore, we have $A \leq P$.

Inferred from $\lambda_j(t) > \sum_{j=1}^{N} \gamma_i(t) \cdot r_i$, we can get $\lambda_j(t) \geq \gamma_i(t) \cdot X_i$ for every task assignment. Thus,

$$\sum_{t=0}^{\infty}\Big(\sum_{j=1}^{N} \lambda_j(t) - \sum_{i=1}^{M} \gamma_i(t) \cdot X_i\Big) \geq 0 \quad (18)$$

Thus, the value of B is guaranteed by,

$$\begin{aligned} B = &\sum_{j=1}^{N} f_j\Big(\sum_{t=0}^{\infty}\sum_{i=1}^{M} x_j^i(t) \cdot c_j\Big) - \sum_{i=1}^{M}\sum_{t=0}^{\infty} g_{it}\Big(\sum_{j=1}^{N} x_j^i(t) \cdot r_i\Big) \\ &- \Big(\sum_{j=1}^{N}\sum_{t=0}^{\infty} \lambda_j(t) - \sum_{i=1}^{M}\sum_{t=0}^{\infty} \gamma_i(t) \cdot X_i\Big) \end{aligned}$$

$$\leq \sum_{j=1}^{N} f_j\Big(\sum_{t=0}^{\infty}\sum_{i=1}^{M} x_j^i(t)\cdot c_j\Big) - \sum_{i=1}^{M}\sum_{t=0}^{\infty} g_{it}\Big(\sum_{j=1}^{N} x_j^i(t)\cdot r_i\Big) \quad (19)$$

Thus, we have $B \leq P$.

Combine (17) and (19), the primal objective value P and the dual objective value D satisfy:

$$D = A + B \leq 2 \cdot P \quad (20)$$

We can conclude that $P \leq P^* \leq D \leq 2 \cdot P$ and $c = 2$ where P^* is the objective value of optimal solutions. TAOAM is also a 2-competitive algorithm with respect to the total profit.

VI. PERFORMANCE EVALUATION

In this section, we evaluate the performance of our proposed online algorithms using a real dataset, which is collected from a practical ridesharing platform. In crowdsourcing systems, the passengers submit their current locations and destinations to the platform by mobile phones. In the platform, the different car presents different unit-price and the payment of a passenger can be viewed as her utility. In addition, each car consumes a certain amount of resource such as the gasoline consumption, which can be viewed as a worker cost. Note that a worker can refuse the task assignment, if the profit of this assignment doesn't make a satisfaction. As a result, we apply the approximation algorithms to make online assignments in such a platform.

A. EXPERIMENT FOR TAOAO

Experiment Setup: We use three months' data from March to May in 2013 [42] as a training data set to evaluate our proposed algorithms. Based on the trace data, we obtain arrival time a_j and finished time d_j for each task j. Moreover, we can get the running length of each task and the unit-price of every car. Similar to [40], the utility of task j is modeled as a concave function $f_j(x_j) = z_j(x_j \cdot c_j)^s$. Here, we consider $s = 1/2$, z_j is generated from a uniform distribution in [1,5] and c_j is the running length of each task. For the cost function, r_i is generated from a uniform distribution in [1,5], which is similar to [40].

Baseline Algorithms: We use the following four schemes as baselines for comparison with our proposed algorithms:

- **OEC**: (Optimization Estimations Costs). In each time-slot, the crowdsourcing system assigns the task to a worker by cvx to minimize cost.
- **NLF**: (Nearest Length First). In each time-slot, the crowdsourcing system assigns the task to the nearest worker.
- **BUF** (Biggest Utility First). In each time-slot, the crowdsourcing system chooses the task with the biggest utility to schedule.
- **PDOAO** (Primal-Dual Online Algorithm One). In each time-slot, the system chooses the task based on [30] in one task for one worker model.

In our proposed algorithms, we find out a worker's child task with the maximum objective value by every task assignment. Therefore, the different number of workers will affect the total profit. In this section, we scale out the different number of workers in a ridesharing platform to show its impact on the total profit. Results in Fig. 4a show that when the number of workers is 200, the total profit is better than that of the other seven situations. Given economic efficiency and the total profit, we ignore that the number of workers is more than 1600 and choose the number of workers to be 200 for TAOAO.

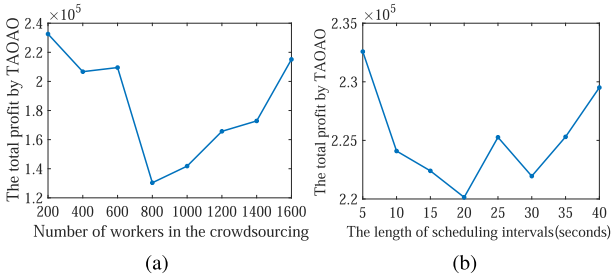


FIGURE 4. The impact of worker number and scheduling intervals in the one task for one worker model.

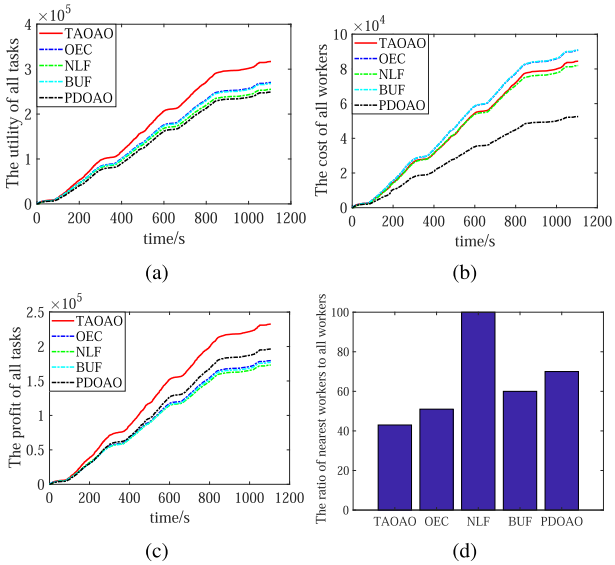


FIGURE 5. The utility, the costs, the total profit and the ratio of nearest workers.

In our model, every task assignment will affect the remaining task assignments. In the proposed algorithms, the different length of each time slot will contain different number of workers and tasks, which will make an influence in remaining task assignments. Thus, it is necessary to study the length of each time slot for total profit. In this part, we evaluate the performance by tuning the length of a scheduling interval, i.e., the length of each time slot. Fig. 4b shows the total profit of 5 seconds is more than the total profit of the other scheduling intervals. As one may expect, with shorter scheduling interval, TAOAO makes better scheduling decisions which tasks are assigned to appropriate workers accurately. Given passengers' waiting time, we choose the scheduling interval to be 5 seconds for TAOAO.

With the parameters set above, we proceed to compare the performance of TAOAO with the other four baseline schemes,

namely, OEC, NLF, BUF and PDOAO. Fig. 5a shows the utility of all tasks when different algorithms iterate over time. For TAOAO, the curve is always upper that of the other four algorithms. OEC and BUF are closely throughout the entire simulation. NLF and PDOAO are closely throughout the entire simulation. Fig. 5a also shows that TAOAO is 20% larger than the other four algorithms. In summary, TAOAO is aimed to maximize total utility. BUF is based on the scheduling priority for the task with the maximum utility. NLF and PDOAO assign a task to a worker without considering the utility.

In Fig. 5b, we depict the costs of all workers with respect to different algorithms iterate over time. For TAOAO, the curve is always under that of OEC and BUF. TAOAO is closely to NLF throughout the entire simulation. The curve of PDOAO is under that of other four algorithms. That is to say, the worker costs of PDOAO is less than that of others. Fig. 5b also shows that, TAOAO reduces total cost vector by nearly 10% comparing to OEC and BUF. In summary, TAOAO algorithm is based on the online primal-dual approach, which will minimize overall costs in each time-slot. In the NLF algorithm, the system assigns the task to the nearest length worker, which will decrease the subsection costs by minimizing empty-car length. The BUF algorithm assigns tasks to workers without considering the costs. In the PDOAO algorithm, the system assigns tasks to workers mainly considering the costs.

Fig. 5c shows the total profit when different algorithms iterate over time. For TAOAO, the curve is always above that of the other four algorithms throughout the entire simulation. The curve of NLF is always under that of the other four curves, which is the mean that the total profit of NLF is smaller than that of others. In a ridesharing platform, it is therefore justified to adopt TAOAO as the scheduling algorithm which aims to maximize the total profit. Fig. 5c shows that TAOAO improves the total actual profit by 20% comparing to the other four algorithms. Therefore, our online algorithm TAOAO shows a good performance as compared to the baseline algorithms.

Fig. 5d shows the ratio of nearest workers to all workers for the four algorithms. For TAOAO, the histogram is always lower than that of the other four algorithms. The histogram of NLF is always above of the other four histograms, which each task is assigned to the nearest worker. Fig. 5d shows that TAOAO algorithm only chooses nearly 43% nearest workers to accomplish tasks and OEC algorithm chooses nearly 51% nearest workers to accomplish tasks. Fig. 5d shows that BUF algorithm chooses nearly 60% nearest workers to accomplish tasks and PDOAO algorithm chooses nearly 70% nearest workers to accomplish tasks. Therefore, we can get that appropriate workers don't include all nearest workers in a ridesharing platform.

B. EXPERIMENT FOR TAOAM

We use three months' data from March to May in 2013 [42] as a training data set to evaluate our proposed algorithms. Based on the trace data, we obtain arrival time a_j and finished time d_j for each task j. Moreover, we can get the running length of each task and the unit-price of every car. Similar to [40], the utility of task j is modeled as a concave function $f_j(x_j) = z_j(x_j \cdot c_j)^s$. Here, we consider $s = 1/2$, z_j is generated from a uniform distribution in [1,5]. The cost function is defined as $g_i(x_j) = q_i(\sum_{j=1}^{N} x_j \cdot r_i)^m$, which is a convex function form similar to [30]. Here, we consider $m = 2$, q_i is generated from a uniform distribution in [1,5].

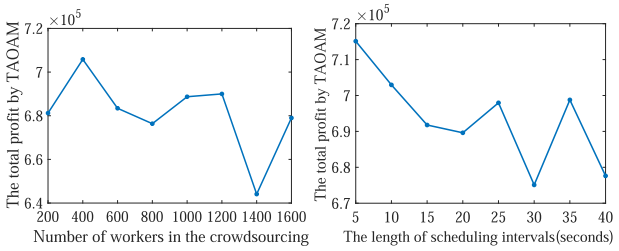


FIGURE 6. The impact of worker number and scheduling intervals in the multiple tasks for one worker model.

Baseline Algorithms: We use **OEC**, **RA**, **BUF**, **PDOAM** as baselines for comparison with our proposed algorithms:

- **RA**: (Random Assignment). In each time-slot, the crowdsourcing system assigns a task to a worker randomly.
- **PDOAM** (Primal-Dual Online Algorithm Multiple). In each time-slot, the system chooses the task based on [30] in multiple tasks for one worker model.

C. THE IMPACT OF M FOR TAOAM

In this section, we scale out the different number of workers in a ridesharing platform to show its impact on the total profit. Results in Fig. 6a shows that when the number of workers is 400, the total profit is better than that of the other seven situations. Given the total profit, we choose the number of workers to be 400 for TAOAM.

In this part, we evaluate the performance by tuning the length of a scheduling interval, i.e., the length of each time slot. Fig. 6b shows the total profit of 5 seconds is more than that of the other scheduling intervals. As one may expect, with shorter scheduling interval, TAOAM makes better scheduling decisions which tasks are assigned to appropriate workers accurately. Given passengers' waiting time, we choose the scheduling interval to be 5 seconds for TAOAM.

With 400 workers and 5 seconds, we proceed to compare the performance of TAOAM with the other four baseline schemes, namely, OEC, RA, BUF and PDOAM. Fig. 7a shows the utility of all tasks when different algorithms iterate over time. For TAOAM, the curve is always upper that of the other four algorithms. OEC, RA and BUF are closely throughout the entire simulation. The curve of PDOAM is under others, which is the mean that the utility of PDOAM is less than that of others. Fig. 7a also shows that TAOAM is 10% larger than the other four algorithms. TAOAM utilizes the online primal-dual approach to maximize total utility, which is better than others.

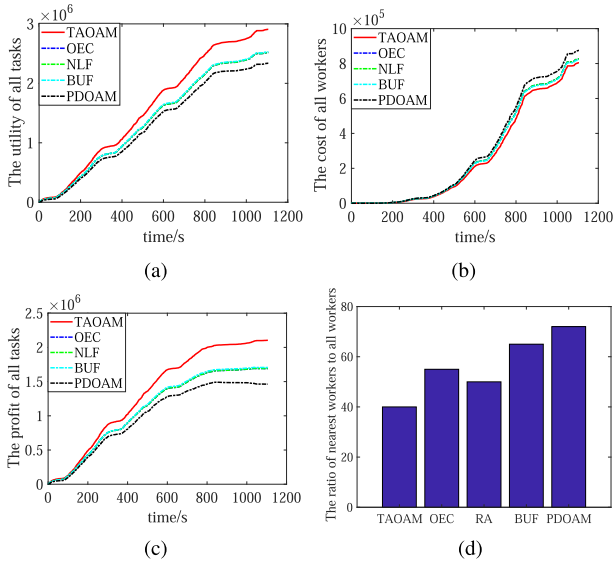


FIGURE 7. The utility, the costs, the total profit and the ratio of nearest workers.

In Fig. 7b, we depict the costs of all workers with respect to different algorithms iterate over time. For TAOAM, the curve is always under that of OEC, RA, BUF and PDOAM. That is to say, the worker costs of TAOAM is less than that of other four algorithms. Fig. 7b also shows that, TAOAM reduces the total costs by nearly 5% comparing to the other four algorithms. In summary, TAOAM algorithm is based on the online primal-dual approach, which will minimize overall costs in each time-slot.

Fig. 7c shows the total profit of all workers when different algorithms iterate over time. For TAOAM, the curve is above that of the other four algorithms. In a ridesharing platform, it is therefore justified to adopt TAOAM as the scheduling algorithm which aims to maximize the total profit. Fig. 7c shows that TAOAM improves the total profit by 20% comparing to the other four algorithms. Therefore, our online algorithm shows a good performance as compared to the baseline algorithms.

Fig. 7d shows the ratio of nearest workers to all workers for the five algorithms. For TAOAM, the histogram is lower than that of the other four algorithms. Fig. 5d shows that TAOAM algorithm only chooses nearly 40% nearest workers to accomplish tasks and OEC algorithm chooses nearly 56% nearest workers to accomplish tasks. Fig. 5d shows that RA algorithm chooses nearly 50% nearest workers to accomplish tasks and BUF algorithm chooses nearly 65% nearest workers to accomplish tasks. For PDOAM, the histogram is higher than that of the other four algorithms. Therefore, we can get that appropriate workers don't include all nearest workers in a ridesharing platform.

VII. CONCLUSION

In this paper, we make an attempt to study the online optimization problem by considering both task utility and worker costs to maximize the total profit in mobile crowdsourcing systems. Following this framework, we construct a pseudo-tree to search the maximum objective value for task assignments and study two optimization problems based on workers' capacity. In addition, we present the online algorithms for task assignments and analyze the competitive performance using a dual-fitting method. Trace driven simulations validate our designed online solutions perform much better than baseline algorithms. As future work, we plan to combine our model with deep-learning for task online scheduling in a crowdsourcing system.

APPENDIX

A. PROOF OF LEMMA 1

Proof: To prove this lemma, it suffices to show that $f_j(\cdot)$ is a concave and increasing function, for any $t \in [a_j, d_j]$, it follow that,

$$\begin{aligned}
&f_j\Big(\sum_{t=a_j}^{d_j}\sum_{i=1}^{M} x_j^i(t)\cdot c_j\Big)\\
&= f_j\Big(\sum_{i=1}^{M} x_j^i(t_1)\cdot c_j + \sum_{i=1}^{M} x_j^i(t_2)\cdot c_j + \cdots + \sum_{i=1}^{M} x_j^i(t_n)\cdot c_j\Big)\\
&\geq f_j'\Big(\sum_{i=1}^{M} x_j^i(t_1)\cdot c_j\Big)\sum_{i=1}^{M} x_j^i(t_1)\cdot c_j\\
&\quad + f_j'\Big(\sum_{i=1}^{M} x_j^i(t_1)\cdot c_j + \sum_{i=1}^{M} x_j^i(t_2)\cdot c_j\Big)\sum_{i=1}^{M} x_j^i(t_2)\cdot c_j\\
&\quad + \cdots + f_j'\Big(\sum_{i=1}^{M} x_j^i(t_1)\cdot c_j + \cdots + \sum_{i=1}^{M} x_j^i(t_n)\cdot c_j\Big)\sum_{i=1}^{M} x_j^i(t_n)\cdot c_j.
\end{aligned} \tag{21}$$

where $t_1, t_2 \ldots t_n$ are the time slots.

Let v_{jt} replace every derivative term:

$$\begin{cases}
f_j'\Big(\sum_{i=1}^{M} x_j^i(t_1)\cdot c_j\Big) = v_j(t_1),\\
f_j'\Big(\sum_{i=1}^{M} x_j^i(t_1)\cdot c_j + \sum_{i=1}^{M} x_j^i(t_2)\cdot c_j\Big) = v_j(t_2),\\
\cdots\\
f_j'\Big(\sum_{i=1}^{M} x_j^i(t_1)\cdot c_j + \cdots + \sum_{i=1}^{M} x_j^i(t_n)\cdot c_j\Big) = v_j(t_n).
\end{cases} \tag{22}$$

Inferred from E.q(22), we have:

$$\begin{aligned}
&\sum_{t=t_1}^{t_n} f_j'\Big(\sum_{t=t_1}^{t_n}\sum_{i=1}^{M} x_j^i(t)\cdot c_j\Big)\cdot\sum_{i=1}^{M} x_j^i(t)\cdot c_j\\
&\qquad = \sum_{t=t_1}^{t_n} v_j(t)\cdot\sum_{i=1}^{M} x_j^i(t)\cdot c_j.
\end{aligned} \tag{23}$$

Thus, we have:

$$f_j\Big(\sum_{t=t_1}^{t_n}\sum_{i=1}^{M} x_j^i(t)\cdot c_j\Big)$$

$$\geq \sum_{t=t_1}^{t_n} f_j'\Big(\sum_{t=t_1}^{t_n}\sum_{i=1}^{M} x_j^i(t)\cdot c_j\Big)\cdot \sum_{i=1}^{M} x_j^i(t)\cdot c_j$$
$$= \sum_{t=t_1}^{t_n}\sum_{i=1}^{M} v_j(t)\cdot x_j^i(t)\cdot c_j. \quad (24)$$

This completes the proof. □

B. PROOF OF LEMMA 2

Proof: To prove this lemma, it suffices to show that $g_{it}(\cdot)$ is a convex and increasing function, for any $t \in [0, \infty]$, it follow that,

$$\begin{aligned} & g_{it}\Big(\sum_{j=1}^{N} x_j^i(t)\cdot r_i\Big) \\ &= g_{it}\Big(x_1^i(t)\cdot r_i + x_2^i(t)\cdot r_i + \cdots + x_N^i(t)\cdot r_i\Big) \\ &\leq g_{it}'\Big(x_1^i(t)\cdot r_i\Big)\cdot x_1^i(t)\cdot r_i \\ &\quad + g_{it}'\Big(x_1^i(t)\cdot r_i + x_2^i(t)\cdot r_2^i\Big)\cdot x_2^i(t)\cdot r_i + \cdots \\ &\quad + g_{it}'\Big(x_1^i(t)\cdot r_i + x_2^i(t)\cdot r_i + \cdots + x_N^i(t)\cdot r_i\Big)\cdot x_N^i(t)\cdot r_i. \end{aligned} \quad (25)$$

Thus, we have:

$$g_{it}\Big(\sum_{j=1}^{N} x_j^i(t)\cdot r_i\Big) \leq \sum_{j=1}^{N} g_{it}'\Big(\sum_{\kappa=1}^{j} x_\kappa^i(t)\cdot r_i\Big) x_j^i(t)\cdot r_i \quad (26)$$

Let $o_i(t)$ replace the derivative term, we have:

$$g_{it}'\Big(\sum_{\kappa=1}^{j} x_\kappa^i(t)\cdot r_i\Big) x_j^i(t)\cdot r_i = o_i(t)\cdot x_j^i(t)\cdot r_i. \quad (27)$$

Thus, we have:

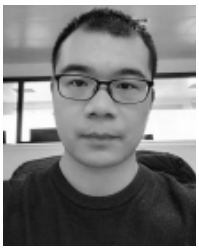
$$g_{it}\Big(\sum_{j=1}^{N} x_j^i(t)\cdot r_i\Big) \leq \sum_{j=1}^{N} o_i(t)\cdot x_j^i(t)\cdot r_i. \quad (28)$$

This completes the proof. □

REFERENCES

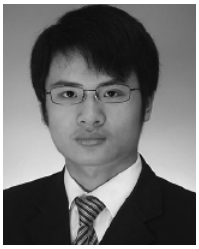
[1] [Online]. Available: http://www.mturk.com
[2] Y. Zhang, Y. Gu, M. Pan, N. H. Tran, Z. Dawy, and Z. Han, "Multi-dimensional incentive mechanism in mobile crowdsourcing with moral hazard," *IEEE Trans. Mobile Comput.*, vol. 17, no. 3, pp. 604–616, Mar. 2018.
[3] A. R. Cardoso, H. Wang, and H. Xu, "The online saddle point problem and online convex optimization with knapsacks," *Mach. Learn.*, 2020.
[4] G. Goel, A. Nikzad, and A. Singla, "Mechanism design for crowdsourcing markets with heterogeneous tasks," in *Proc. AAAI*, 2014, pp. 1–10.
[5] H. Hu, Y. Zheng, Z. Bao, G. Li, J. Feng, and R. Cheng, "Crowdsourced poi labelling: Location-aware result inference and task assignment," in *Proc. ICDE*, 2016, pp. 61–72.
[6] D. Wedelin, "An algorithm for large scale 0–1 integer programming with application to airline crew scheduling," *Ann. Oper. Res.*, vol. 57, no. 1, pp. 283–301, Dec. 1995.
[7] D. Deng, C. Shahabi, and U. Demiryurek, "Maximizing the number of worker's self-selected tasks in spatial crowdsourcing," in *Proc. 21st ACM SIGSPATIAL Int. Conf. Adv. Geograph. Inf. Syst.*, Nov. 2013, pp. 314–323.
[8] Q. Liu, T. Luo, R. Tang, and S. Bressan, "An efficient and truthful pricing mechanism for team formation in crowdsourcing markets," in *Proc. ICC*, 2015, pp. 567–572.
[9] H. Chien, A. Vaughan, and J. Wortman, "Online task assignment in crowdsourcing markets," in *Proc. AAAI*, 2012, pp. 1–10.
[10] J. F. Xia, "Profit-driven task assignment in spatial crowdsourcing," in *Proc. IJCAI*, 2019, pp. 1–7.
[11] Y. Tong, J. She, B. Ding, L. Wang, and L. Chen, "Online mobile micro-task allocation in spatial crowdsourcing," in *Proc. ICDE*, 2016, pp. 49–60.
[12] Y. Zhao, C. Tian, Z. Zhu, J. Cheng, C. Qiao, and A. X. Liu, "Minimize the make-span of batched requests for FPGA pooling in cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 11, pp. 2514–2527, Nov. 2018.
[13] L. Kazemi and C. Shahabi, "Geocrowd: Enabling query answering with spatial crowdsourcing," in *Proc. 21st SIGSPATIAL GIS*, 2012, pp. 189–198.
[14] H. To, C. Shahabi, and L. Kazemi, "A server-assigned spatial crowdsourcing framework," *ACM Trans. Spatial Algorithms Syst.*, vol. 1, no. 1, pp. 1–28, Aug. 2015.
[15] Y. Li, M. L.-Yiu, and W. Xu, "Oriented online route recommendation for spatial crowdsourcing task workers," in *Advances in Spatial and Temporal Databases*. Cham, Switzerland: Springer, 2015, pp. 137–156.
[16] P. Cheng, X. Lian, Z. Chen, R. Fu, L. Chen, and J. Han, "Reliable diversity-based spatial crowdsourcing by moving workers," *Proc. VLDB Endowment*, vol. 8, no. 10, pp. 347–360, 2015.
[17] T. Song, Y. Tong, and L. Wang, "Trichromatic online matching in real-time spatial crowdsourcing," in *Proc. ICDE*, 2017, pp. 1009–1020.
[18] G. Goel, A. Nikzad, and A. Singla, "Allocating tasks to workers with matching constraints: Truthful mechanisms for crowdsourcing markets," *J. ACM*, pp. 279–280, 2014.
[19] Y. Singer and M. Mittal, "Pricing mechanisms for crowdsourcing markets," in *Proc. WWW*, 2013, pp. 1157–1166.
[20] M. Cheng and V. Varma, "Scaling up the crowd: Micro task pricing schemes for worker retention and latency improvement," in *Proc. AAAI*, 2014, pp. 193–208.
[21] S. M. Bernstein, D. R. Karger, R. C. Miller, and J. Brandt, "Analyt IC methods for optimizing realtime crowdsourcing," *Comput. Sci.*, 2012.
[22] M. Mao and M. Humphrey, "Auto-scaling to minimize cost and meet application deadlines in cloud workflows," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal.*, 2011, pp. 1–12.
[23] Q. Zhu and G. Agrawal, "Resource provisioning with budget constraints for adaptive applications in cloud environments," in *Proc. HPDC*, 2010, pp. 304–307.
[24] S. Sarker, M. A. Razzaque, M. M. Hassan, A. Almogren, G. Fortino, and M. Zhou, "Optimal selection of crowdsourcing workers balancing their utilities and platform profit," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8602–8614, Oct. 2019.
[25] N. Buchbinder, K. Jain, and J. Naor, "Online primal-dual algorithms for maximizing ad-auctions revenue," in *Proc. 15th Annu. Eur. Symp. Algorithms*, 2007, pp. 253–264.
[26] N. Buchbinder and J. Naor, "The design of competitive online algorithms via a primal-dual approach," *Found. Trends Theor. Comput. Sci.*, pp. 263–293, 2009.
[27] N. R. Devanur and Z. Huang, "Primal dual gives optimal energy efficient online algorithms," in *Proc. SODA*, 2014, pp. 1123–1140.
[28] A. Blum, A. Gupta, Y. Mansour, and A. Sharma, "Welfare and profit maximization with production costs," in *Proc. IEEE FOCS*, Oct. 2011, pp. 77–86.
[29] N. R. Devanur and K. Jain, "Online matching with concave returns," in *Proc. STOC*, 2012, pp. 137–144.
[30] X. Zhang, Z. Huang, C. Wu, and Z. Li, "Online auctions in IaaS clouds: Welfare and profit maximization with server costs," in *Proc. SIGMETRICS*, 2015, pp. 3–15.
[31] N. Buchbinder, K. Jain, and J. Naor, "Online primal-dual algorithms for covering and packing," in *Proc. 13th Annu. Eur. Symp. Algorithms*, 2005, pp. 689–701.
[32] A. Gupta, R. Krishnaswamy, and K. Pruhs, "Online primal-dual for non-linear optimization with applications to speed scaling," in *Proc. WAOA*, 2002, pp. 173–186.
[33] S. Anand, N. Garg, and A. Kumar, "Resource augmentation for weighted flow-time explained by dual fitting gradient ascent," in *Proc. SODA*, 2012, pp. 1228–1241.
[34] S. Im, J. Kulkarni, and K. Munagala, "Competitive algorithms from competitive equilibria: Non-clairvoyant scheduling under polyhedral constraints," in *Proc. STOC*, 2014, pp. 313–322.
[35] K. T. Nguyen, "Lagrangian duality in online scheduling with resource augmentation and speed scaling," in *Proc. ESA*, 2013, pp. 755–766.

[36] Y. Azar, N. Buchbinder, T. H. Chan, S. Chen, I. R. Cohen, A. Gupta, Z. Huang, N. Kang, V. Nagarajan, J. Naor, and D. Panigrahi, "Online algorithms for covering and packing problems with convex objectives," in *Proc. IEEE FOCS*, Oct. 2016, pp. 148–157.
[37] S. Anand, N. Garg, and A. Kumar, "Resource augmentation for weighted flow-time explained by dual fitting," in *Proc. 23rd Annu. ACM-SIAM Symp. Discrete Algorithms*, Jan. 2012, pp. 1228–1241.
[38] S. Agrawal and N. R. Devanur, "Fast algorithms for online stochastic convex programming," in *Proc. SODA*, 2015, pp. 1405–1424.
[39] B. Zheng, C. Huang, C. S. Jensen, L. Chen, N. Q. V. Hung, G. Liu, G. Li, and K. Zheng, "Online trichromatic pickup and delivery scheduling in spatial crowdsourcing," in *Proc. ICDE*, 2020, pp. 973–984.
[40] Z. Zheng and N. Shroff, "Online multi-resource allocation for deadline sensitive jobs with partial values in the cloud," in *Proc. IEEE Infocom*, Apr. 2016, pp. 1–9.
[41] E. Aldhahri, V. Shandilya, and S. Shiva, "Crowdsourcing multi-objective recommendation system," in *Proc. WWW*, 2018, pp. 1371–1379.
[42] X. Zhou, H. Rong, C. Yang, Q. Zhang, A. V. Khezerlou, H. Zheng, Z. Shafiq, and A. X. Liu, "Optimizing taxi driver profit efficiency: A spatial network-based Markov decision process approach," *IEEE Trans. Big Data*, vol. 6, no. 1, pp. 145–158, Mar. 2020.
[43] H. Xu, Y. Liu, W. C. Lau, T. Zeng, J. Guo, and A. X. Liu, "Online resource allocation with machine variability: A bandit perspective," *IEEE/ACM Trans. Netw.*, vol. 28, no. 5, pp. 2243–2256, Oct. 2020.

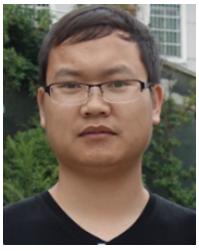


QI LI is currently pursuing the Ph.D. degree with the College of Computer Science and Electronic Engineering, Hunan University, Changsha, China. His research interests include job scheduling and resource allocation.

LIJUN CAI received the Ph.D. degree from the College of Computer Science and Electronic Engineering, Hunan University, in 2007. He is currently a Professor with Hunan University. His research interests include bioinformatics, cloud computing, and big data scheduling and management.



HUANLE XU received the B.Sc. (Eng.) degree from the Department of Information Engineering, Shanghai Jiao Tong University (SJTU), in 2012, and the Ph.D. degree from the Department of Information Engineering, The Chinese University of Hong Kong (CUHK), in 2016. He is interested in designing wonderful algorithms for real applications and practical systems using mathematical tools. His primary research interests include job scheduling and resource allocation in cloud computing, decentralized social networks, parallel graph algorithms, and machine learning.



TAO MENG is currently pursuing the Ph.D. degree with the College of Computer Science and Electronic Engineering, Hunan University, Changsha, China. His research interests include date mining and network analysis.