Optimization Algorithm Balancing Output and Fairness in Crowdsourcing

Xiao Chen
Department of Computer Science
Texas State University
San Marcos, TX 78666
xc10@txstate.edu

***Abstract*—Crowdsourcing has become increasingly popular in recent years as it enables requesters to find a group of workers to work on small tasks that an individual or organization cannot easily do. One of the main challenges in crowdsourcing is to ensure worker participation. Many papers propose incentive mechanisms among which a few are long-term incentives. However, none of them have combined incentives with workers' output. In this paper, we address not only the long-term incentive issue but also the workers' time-average output maximization issue by formulating a stochastic optimization problem. In our problem, while maximizing workers' output is an explicit objective, the long-term incentive is realized through the requester's fairness towards workers. We solve the problem using the Lyapunov technique and turn the solution into interactive but independent optimization decisions on the side of the workers and the requester in each time slot. To evaluate the performance of our solution, we conduct theoretical analysis first and then simulations to compare our solution with theoretical values and two other variations. Analysis and simulation results show that our solution can maximize workers' time-average output while ensuring fairness to retain the workers in the long run.**

***Index Terms*—crowdsourcing, incentive, Lyapunov, optimization, stochasticity**

I. Introduction

Crowdsourcing [1] has gained popularity in recent years because it allows requesters to find a group of workers online to work on small tasks that an individual or organization cannot easily do. There are three basic components in crowdsourcing: requesters who publish tasks on a platform, workers who carry out the tasks, and a platform such as Amazon Mechanical Turk [2] that matches requesters and workers.

One of the main problems in crowdsourcing is to ensure worker participation. There are two classes of incentive mechanisms: the short-term and the long-term. Many of the existing incentive mechanisms [3]–[5] are short-term, compensating workers directly and not considering how to attract them to participate for a longer period of time. In the long-term category, [6]–[8] exploited reverse auction to encourage workers in the long run. In [9], a game-based incentive mechanism was developed to encourage workers to participate over the long term and provide high-quality data. In [10], a dynamic long-term incentive mechanism was explored based on reputation and contract theory. A few other works took stochasticity and unpredictability into account and optimized incentives using stochastic optimization techniques like Lyapunov optimation [11]–[13].

In this paper, we address not only the long-term incentive problem but also the problem of maximizing workers' time-average output by formulating a stochastic optimization problem. To the best of our knowledge, this is the first paper that has ever combined these two issues in crowdsourcing. In our defined problem, while maximizing workers' output is an explicit objective, the requester ensures long-term incentives by treating workers fairly, which involves guaranteeing queue stability by removing and paying for tasks from their queues. Our key contributions are manifold: (1) We define and solve a stochastic optimization problem using the Lyapunov technique and turn the solution into interactive but independent optimization decisions on the side of the workers and the requester in each time slot. On the workers' side, they decide the optimal number of tasks to accept and accomplish, i.e. *Optimal Task Output* (OTO). On the requester's side, he uses an *Optimal Task Selection* (OTS) algorithm to select tasks from workers' queues to ensure that each worker has a chance to be paid. This serves as an incentive for the workers to output more tasks in the long run. In addition, the solution can balance the stochastic maximization of time-average workers' output and fairness. (2) We demonstrate that the proposed solution deviates by at most $O(1/V)$ from optimality and has queue backlogs bounded by $O(V)$, where V is a non-negative control parameter. (3) We conduct simulations to compare the proposed algorithm with theoretical results and with two other variations by adopting *round robin* (RR) and the *longest queue first* (LQF) algorithms, respectively in the task selection process of the requester. The simulation results indicate that our proposed algorithm can maximize workers' output while ensuring fairness to retain them.

The rest of the paper is organized as follows: Section II references the related work. Section III defines the problem. Section IV provides the solution to the problem. Section V describes the simulations we conducted, and the conclusion is in Section VI.

II. Related Work

In crowdsourcing, ensuring continual participation from workers is a very important issue. There are two classes of incentive mechanisms: the short-term and the long-term. Most of the existing incentive mechanisms are short-term incentives [3]–[5] that directly compensate workers and do not consider drawing workers to participate in crowd tasks for a long period of time.

The long-term incentives use reverse auctions, game theory, contract theory, and the Lyapunov method to attract workers in the long run. Papers [6], [7], [8] all use reverse auction methods to guarantee workers' long-term participation. Chi et al. [9] developed a multistrategy repeated game-based incentive mechanism to guide participants to provide long-term participation and high-quality data. Combining reputation and contract theory, Zhao et al. [10] explored a dynamic long-term incentive mechanism to attract mobile users to participate in crowdsourcing networks. Gao et al. [11] investigated a Lyapunov-based VCG auction policy for online worker selection while considering long-term user participation incentives. Sun et al. [12] presented a semi-online frugal incentive mechanism by introducing a Lyapunov method and then extended it to satisfy the long-term participation constraint and approximate optimality. And Wang et al. [13] focused on a reputation framework to attract and retain workers in a competitive market using Lyapunov optimization.

In this paper, we will discuss not only the incentive problem but also the workers' time-average output problem using the Lyapunov model, which has not been explored in the existing literature.

III. Problem Definition

In this section, we define the problem that we want to solve.

A. The Workers

Let $A_i(t)$ denote the arrival of tasks at worker i at time slot t. $A_i(t)$ is an i.i.d. random process with the maximum value of A_i^{max}. Due to the willingness and capability of worker i, he may only perform a few, denoted by $a_i(t)$, $0 \le a_i(t) \le A_i(t)$, of the arrived tasks. Therefore, a queue may build up with time. At time slot t, a requester may select worker i and purchase his $b_i(t)$ performed tasks. Once a task is chosen, it will be removed from the worker's queue. We use $Q_i(t)$ to represent the backlog of worker i's task queue. It is updated in each time slot as follows:

$$Q_i(t+1) = [Q_i(t) - b_i(t)] + a_i(t) \tag{1}$$

B. The Requester

At time slot t, a requester will receive information (details provided later in Algorithm 1) from the workers and decide from which workers to purchase the finished tasks and how many. If the requester picks $b_i(t)$ finished tasks from worker i, he will pay for them with a price of $p_i(t) = u_i b_i(t)$, where u_i is the unit price of a task charged by worker i. We assume that at time slot t, the total amount of money that a requester has is $P(t)$. It is a stochastic variable with a maximum value of P^{max}. If the requester cannot afford all the tasks received at time slot t using $P(t)$, a queue of money, denoted by $M(t)$, will accumulate. At each time slot, it is updated as follows:

$$M(t+1) = \max[M(t) - P(t), 0] + \sum_{i=1}^{N} p_i(t) \tag{2}$$

C. Problem Formulation

We first select the utility function $\phi(\overline{a})$ that satisfies our goal to embed both workers' output and fairness.

$$\phi(\overline{a}) = \sum_{i=1}^{N} \log(1 + \overline{a_i}) \tag{3}$$

Here, $\overline{a} = [\overline{a_1}, \overline{a_2}, \cdots, \overline{a_N}]$ is a vector that contains the time-average of tasks completed by N workers. If $Y(\tau)$ is a stochastic process, the time-average of the process is defined as: $\overline{Y} = \lim_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} E[Y(\tau)]$. We chose $\phi(\overline{a})$ to be a logarithm function because it is concave and has a diminishing return property with each increase of $\overline{a}$. Let us use two workers as an example to explain this. If we want to maximize $\log(1+\overline{a_1}) + \log(1+\overline{a_2})$ and if $\overline{a_1} < \overline{a_2}$, the sum of the two logarithm functions will increase more if we increase $\overline{a_1}$ than $\overline{a_2}$. This means that the requester should remove more tasks from worker 1 to boost his output. That is to say that the utility function implies fairness, which is used as an incentive because if a worker is rarely selected and paid, he may lose interest in further participation.

By adding the constraints, we formulate problem $\mathcal{P}1$.

$$\mathcal{P}1: \underset{a(t),b(t)}{\text{maximize}} \quad \phi(\overline{a}) \tag{4a}$$

$$\text{subject to} \quad 0 \le a_i(t) \le A_i(t), \forall i \in N \tag{4b}$$

$$\lim_{t\to\infty} \frac{E\{|Q_i|\}}{t} = 0, \forall i \in N \tag{4c}$$

$$\lim_{t\to\infty} \frac{E\{|M|\}}{t} = 0 \tag{4d}$$

In problem $\mathcal{P}1$, line (4a) is the objective. Constraint (4b) means that the number of performed tasks cannot be larger than the number of arrived tasks. Constraints (4c) and (4d) ensure that the queues Q_i and M are mean rate stable, which is essential for maintaining system stability over time [14].

IV. Solution

The objective of problem $\mathcal{P}1$ is a function of time averages. It is not easy to deal with. We transform it into an equivalent problem $\mathcal{P}2$.

$$\mathcal{P}2: \underset{a(t),b(t),\gamma(t)}{\text{maximize}} \quad \overline{\phi(\gamma)} \tag{5a}$$

$$\text{subject to} \quad \text{Constraints (4b), (4c), (4d)} \tag{5b}$$

$$\overline{\gamma_i} \le \overline{a_i} \tag{5c}$$

$$0 \le \gamma_i(t) \le A_i^{max} \tag{5d}$$

Vector $\gamma(t) = [\gamma_1(t), \cdots, \gamma_N(t)]$ contains a list of auxiliary variables in each time slot t.

Problem $\mathcal{P}2$ is related to $\mathcal{P}1$ as follows: $\mathcal{P}2$ has two more constraints (5c) (5d) than $\mathcal{P}1$. We need to prove that the maximum for $\mathcal{P}2$ is at least as good as that for $\mathcal{P}1$ under these two constraints. Let $\overline{\phi(\gamma)}$ be the maximum for $\mathcal{P}2$, and $\overline{a^*}$ the corresponding time-average task output. We have $\phi(\overline{a^*}) \ge \phi(\overline{\gamma}) \ge \overline{\phi(\gamma)}$. The first inequality is due to constraint (5c) and the monotonic increasing property of function ϕ, and the second is Jensen's inequality [15]. We denote ϕ^{opt} as the maximum for $\mathcal{P}1$ and $a^{opt}(t)$ the corresponding task output at time slot t. We can construct a policy that satisfies all constraints in $\mathcal{P}2$ and by setting $\gamma(t) = \overline{a^{opt}}$ for all t, we get $\overline{\phi(\gamma)} = \phi^{opt}$. Attaching to the previous inequalities, we have $\phi(\overline{a^*}) \geq \phi^{opt}$. Thus, solving $\mathcal{P}2$ ensures all constraints in $\mathcal{P}1$ are satisfied while producing a maximum that is at least as good as that for $\mathcal{P}1$.

In $\mathcal{P}2$, to satisfy the constraint (5c), we replace it with the mean rate stable condition of a worker-specific virtual queue, denoted by $G_i(t)$, which is updated as follows:

$$G_i(t+1) = \max\{G_i(t) + \gamma_i(t) - a_i(t), 0\} \tag{6}$$

Constraint (5c) is satisfied if and only if $G_i(t)$ is mean rate stable [14]. Therefore, problem $\mathcal{P}2$ becomes problem $\mathcal{P}3$:

$$\begin{aligned} \mathcal{P}3: \quad & \underset{a(t),b(t),\gamma(t)}{\text{maximize}} \quad \overline{\phi(\gamma)} && \text{(7a)} \\ & \text{subject to} \quad \text{Constraints (4b), (4c), (4d), (5d)} && \text{(7b)} \\ & \lim_{t\to\infty} \frac{E\{|G_i|\}}{t} = 0 && \text{(7c)} \end{aligned}$$

After the transformation, problem $\mathcal{P}3$ only involves time averages, rather than a function of time averages, it can be solved using the Lyapunov drift-plus-penalty framework [14].

Let $\Theta(t) = [Q_i(t), M(t), G_i(t)]$ be a concatenated vector of all actual and virtual queues, with update equations (1), (2), and (6), respectively. We define the Lyapunov function:

$$L(\Theta(t)) = \frac{1}{2}\{M(t)^2 + \sum_{i=1}^{N}[Q_i(t)^2 + G_i(t)^2]\} \tag{8}$$

The drift-plus-penalty function is:

$$\Delta(\Theta(t)) - VE\{\sum_{i=1}^{N}\log(1+\gamma_i(t))|\Theta(t)\}, \tag{9}$$

where $\Delta(\Theta(t)) = E\{L(\Theta(t+1)) - L(\Theta(t))|\Theta(t)\}$ represents the conditional Lyapunov drift for slot t, that is, the conditional expectation of the change in the Lyapunov function from one slot to the next, and $V \geq 0$ is a parameter that represents a weight on how much we emphasize system utility and queue stability.

According to [14], minimizing an upper bound of (9) can maximize the time-average conditional expectation of system utility while achieving queue stability. The upper-bound of (9) is:

$$\begin{aligned} &\Delta(\Theta(t)) - VE\sum_{i=1}^{N}\log(1+\gamma_i(t))|\Theta(t) \leq B + \\ &M(t)E[\sum_{i=1}^{N}p_i(t) - P(t)|\Theta(t)] + \sum_{i=1}^{N}Q_i(t)E[a_i(t) - b_i(t)|\Theta(t)] \\ &+ \sum_{i=1}^{N}G_i(t)E[\gamma_i(t) - a_i(t)|\Theta(t)] - VE[\sum_{i=1}^{N}\log(1+\gamma_i(t))|\Theta(t)] \end{aligned} \tag{10}$$

Theorem 1: The drift-plus-penalty function is upper-bounded by the right-hand side of (10).

Proof. We first look at $\Delta(\Theta(t)) = E\{L(\Theta(t+1)) - L(\Theta(t))|\Theta(t)\}$. From (8), $\Delta(\Theta(t)) = \frac{1}{2}E\{M(t+1)^2 + \sum_{i=1}^{N}[Q_i(t+1)^2 + G_i(t+1)^2] - M(t)^2 - \sum_{i=1}^{N}[Q_i(t)^2 + G_i(t)^2]|\Theta(t)\}$. Expanding it using (1), (2), (6) and utilizing formula $(\max[x-y,0]+z)^2 \leq x^2+y^2+z^2 = 2x(z-y)$ for any $x, y, z \geq 0$, we get

$$\begin{aligned} &\Delta(\Theta(t)) \leq \frac{1}{2}E\{M(t)^2 + P(t)^2 + (\sum_{i}^{N}p_i(t))^2 + 2M(t)\sum_{i=1}^{N}p_i(t) \\ &\quad - 2M(t)P(t) - M(t)^2 + \sum_{i=1}^{N}[(Q_i(t) - b_i(t) + a_i(t))^2 - Q_i(t)^2] \\ &\quad + \sum_{i=1}^{N}[(G_i(t) - ((a_i(t) - \gamma_i(t)))^2 - G_i(t)^2]|\Theta(t)\} \\ &\leq \frac{1}{2}E\{P(t)^2|\Theta(t)\} + \frac{1}{2}E\{\sum_{i=1}^{N}p_i(t)^2|\Theta(t)\} + \frac{1}{2}E\{\sum_{i=1}^{N}b_i(t)^2|\Theta(t)\} \\ &\quad + E\{\sum_{i=1}^{N}a_i(t)^2|\Theta(t)\} + \frac{1}{2}E\{\sum_{i=1}^{N}\gamma_i(t)^2|\Theta(t)\} \\ &\quad + M(t)E[\sum_{i=1}^{N}p_i(t) - P(t)|\Theta(t)] + \sum_{i=1}^{N}Q_i(t)E[a_i(t) - b_i(t)|\Theta(t)] \\ &\quad + \sum_{i=1}^{N}G_i(t)E[\gamma_i(t) - a_i(t)|\Theta(t)] \\ &\leq (P^{max})^2 + 2(A_i^{max})^2 + M(t)E[\sum_{i=1}^{N}p_i(t) - P(t)|\Theta(t)] \\ &\quad + \sum_{i=1}^{N}Q_i(t)E[a_i(t) - b_i(t)|\Theta(t)] \quad + \sum_{i=1}^{N}G_i(t)E[\gamma_i(t) - a_i(t)|\Theta(t)] \end{aligned}$$

Let $B = (P^{max})^2 + 2(A_i^{max})^2$. It is a constant and obtained by maximizing the expected conditional value of each squared item. After finding the upper-bound of $\Delta(\Theta(t))$, we subtract $VE[\sum_{i=1}^{N}\log(1+\gamma_i(t))|\Theta(t)]$ from both sides. This theorem is proved. □

Now solving problem $\mathcal{P}3$ becomes minimizing the right-hand side of (10), subject to the constraints in $\mathcal{P}3$. Since B is a constant and $P(t)$ is irrelevant to the variables $a(t), b(t), \gamma(t)$, we can drop them and rearrange the remaining parts based on the variables. Then minimizing the right-hand side becomes minimizing $f_1(\gamma(t)) + f_2(a(t)) + f3((b(t))$, where

$$\begin{aligned} f_1(\gamma(t)) &= \sum_{i=1}^{N}[-V\log(1+\gamma_i(t)) + G_i(t)\gamma_i(t)] \\ f_2(a(t)) &= \sum_{i=1}^{N}[Q_i(t) - G_i(t)]a_i(t) \\ f_3(b(t)) &= \sum_{i=1}^{N}[-Q_i(t) + u_iM(t)]b_i(t) \end{aligned} \tag{11}$$

Since $a(t)$, $b(t)$, and $\gamma(t)$ are decoupled in both the objectives and constraints, the three functions can be minimized separately as follows.

a). Minimize $f_1(\gamma(t))$

$$\begin{aligned} &\min\ f_1(\gamma(t)) = \sum_{i=1}^{N}[-V\log(1+\gamma_i(t)) + G_i(t)\gamma_i(t)] \\ &\text{subject to } 0 \leq \gamma_i(t) \leq A_i^{max}, \forall i \in N \end{aligned} \tag{12}$$

Since each worker is independent, we can let each worker

$$\begin{aligned} &\min_{\gamma_i(t)} \; -V\log(1+\gamma_i(t)) + G_i(t)\gamma_i(t) \\ &\text{subject to } 0 \le \gamma_i(t) \le A_i^{max} \end{aligned} \tag{13}$$

We take the first-order partial derivative of the objective in (13) with respect to $\gamma_i(t)$ and make it equal to 0. We get $\gamma_i(t) = \frac{V}{G_i(t)\ln 2} - 1$. Considering the boundary values of $\gamma_i(t)$ and the case when $G_i(t) = 0$, the optimal

$$\gamma_i(t) = \begin{cases} A_i^{max}, & \text{if } G_i(t) = 0 \\ 0, & \text{if } G_i(t) \ge \frac{V}{\ln 2} \\ \min\{\frac{V}{G_i(t)\ln 2} - 1, A_i^{max}\}, & \text{if } G_i(t) < \frac{V}{\ln 2} \end{cases} \tag{14}$$

b). Minimize $f_2(a(t))$

$$\begin{aligned} &\min \; f_2(a(t)) = \sum_{i=1}^{N} [Q_i(t) - G_i(t)] a_i(t) \\ &\text{subject to } 0 \le a_i(t) \le A_i(t), \forall i \in N \end{aligned} \tag{15}$$

The optimal

$$a_i(t) = \begin{cases} 0, & \text{if } Q_i(t) \ge G_i(t) \\ A_i(t), & \text{if } Q_i(t) < G_i(t) \end{cases} \tag{16}$$

c). Minimize $f_3(b(t))$

Minimizing this function is equivalent to maximizing its negative, which is:

$$\begin{aligned} &\max \; f_3(b(t)) = \sum_{i=1}^{N} [Q_i(t) - u_i M(t)] b_i(t) \\ &\text{subject to } u_i b_i(t) \le P(t) \end{aligned} \tag{17}$$

This is a knapsack problem [16]. Here $b(t)$ is a set of items, each having a value $Q_i(t) - u_i M(t)$. The given limit is the total amount of money $P(t)$ that the requester can pay at time slot t. To solve the problem, we can use the greedy method to order the workers in the descending order of $\frac{Q_i(t) - u_i M(t)}{u_i}$ and choose the workers one by one until the requester's budget runs out. Assume that the last worker chosen is L, then the optimal

$$b_i(t) = \begin{cases} a_i(t), & \text{if } i < L \\ P(t) - \sum_{i=1}^{L-1} u_i a_i(t), & \text{if } i = L \\ 0, & \text{if } i > L \end{cases} \tag{18}$$

Based on the solution, we propose an optimal solution in Algorithm 1 (A1) to solve $\mathcal{P}3$, and therefore $\mathcal{P}1$. The algorithm describes the interaction between the requester and the workers over time to maximize workers' output and ensure fairness through queue stability.

The following two theorems present the optimality and queue stability of our solution.

Theorem 2: Define ϕ^{opt} and $\phi(\bar{a})$ as the maximum utilities of problems $\mathcal{P}1$ and $\mathcal{P}3$, respectively. Then $\phi^{opt} - \phi(\bar{a}) \le B/V$ holds true. That is, the solution provided by Algorithm 1 deviates from the optimal solution to $\mathcal{P}1$ by $O(1/V)$.

Algorithm 1: Optimal Solution Balancing Workers' Output and Fairness

Initialize all queues to empty;
V is given; each worker i decides u_i;
/*In each time slot t, the workers and the requester interact as follows:*/

For each worker i: /*Optimal Task Output*/

Acquire $Q_i(t), G_i(t), A_i(t)$;
Find optimal $\gamma_i(t)$ using (14) and $a_i(t)$ using (16);
Send $\gamma_i(t)$, $a_i(t)$, $Q_i(t)$, and u_i to the crowdsourcing platform;
Receive $b_i(t)$ from the platform;
Update $Q_i(t+1)$ using (1);
Update $G_i(t+1)$ using (6);

For the requester: /*Optimal Task Selection*/

Aquire $P(t)$;
Collect $\gamma_i(t)$, $a_i(t)$, $Q_i(t)$, and u_i from worker i;
Find optimal $b_i(t)$ using (18);
Send $b_i(t)$ to worker i;
Update $M(t+1)$ using (2);

Proof. According to [14], for any $\delta > 0$, there exists a randomized i.i.d control policy π over time slots that makes the resulting $a_i(t)$, $b_i(t)$, and $\gamma(t)$ independent of $\Theta(t)$, and

$$\begin{aligned} -\phi(\gamma(t)|\pi) \le -\phi^{opt} + \delta, \quad E[\sum_{i=1}^{N} p_i(t) - P(t)|\pi] \le \delta \\ E[a_i(t) - b_i(t)|\pi] \le \delta, \quad E[\gamma_i(t) - a_i(t)|\pi] \le \delta \end{aligned} \tag{19}$$

Plugging (19) into the right side of (10), making $\delta \to 0$, and expanding $\Delta(\Theta(t))$, we get

$$E\{L(\Theta(t+1)) - L(\Theta(t))|\Theta(t)\} - VE[\phi(\gamma(t))] \le B - V\phi^{opt}$$

Using the law of telescoping sums [17] over time slots $\{0, 1, \cdots, T-1\}$, we have

$$E[L(T)] - E[L(0)] - V\sum_{t=0}^{T-1} E[\phi(\gamma(t))] \le BT - VT\phi^{opt}$$

All the queues are initially empty. So $E[L(0)] = 0$. Dividing both sides by VT and taking the limit $T \to \infty$ yield

$$\phi^{opt} - \frac{B}{V} \le \lim_{T\to\infty} \frac{1}{T} E[\phi(\gamma(t))] = \overline{\phi(\gamma)} \le \phi(\overline{\gamma}) \le \phi(\bar{a})$$

The latter part is attributed to Jensen's inequality [15], (5c), and the monotonic increasing property of function ϕ. Thus, $\phi^{opt} - \phi(\bar{a}) \le B/V$, i.e., $O(1/V)$, holds true. □

Theorem 3: The solution given by Algorithm 1 makes all the queues bounded by $O(V)$. More specifically,

$$\overline{G(t)} \le \frac{V}{\ln 2} + \frac{1}{N}\sum_{i=1}^{N} A_i^{max} \tag{20}$$

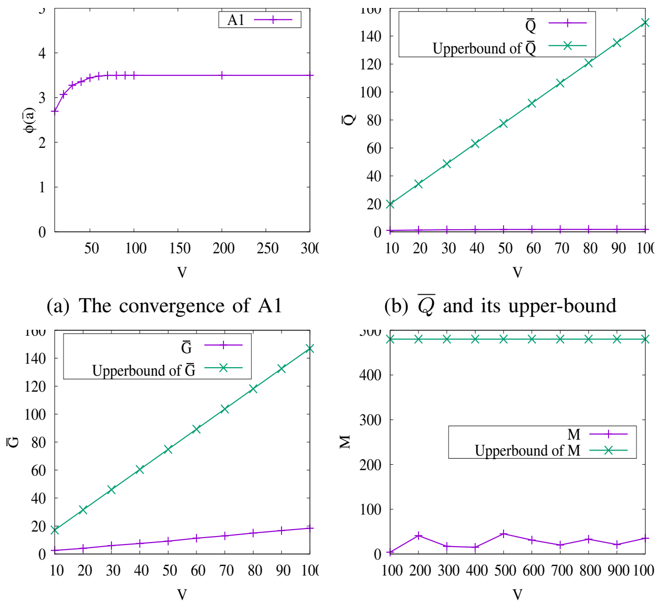
$$\overline{Q(t)} \leq \frac{V}{\ln 2} + \frac{2}{N}\sum_{i=1}^{N} A_i^{max} \tag{21}$$

$$M(t) \leq \max\{\frac{V/\ln 2 + 2A_i^{max}}{u_i}\} + \sum_{i=1}^{N} u_i(\frac{V}{\ln 2} + 2A_i^{max}) \tag{22}$$

Proof. To prove (20), we first prove the bound of $G_i(t) \leq \frac{V}{\ln 2} + A_i^{max}$. We use induction. When $t = 0$, $G_i(0) = 0$. The bound is true. Now we assume that the bound is true at t. Next, we prove that the bound holds true at $t+1$. $G_i(t+1)$ is updated by (6). Whenever the max operation returns a zero, the bound is true. Now let us look at cases when the max operation does not return a zero. There are three cases in (14). Case 1. If $G_i(t) = 0$, $\gamma_i(t) = A_i^{max}$ and $G_i(t+1) = A_i^{max} - a_i(t) \leq \frac{V}{\ln 2} + A_i^{max}$. The bound is true at $t+1$. Case 2. If $G_i(t) \geq \frac{V}{\ln 2}$, $\gamma_i(t) = 0$ and $G_i(t+1) = G_i(t) - a_i(t) \leq G_i(t) \leq \frac{V}{\ln 2} + A_i^{max}$. The last inequality uses the assumption in the induction. So the bound also holds at $t+1$. And case 3. If $G_i(t) < \frac{V}{\ln 2}$, $\gamma_i(t) = \min\{\frac{V}{G_i(t)\ln 2} - 1, A_i^{max}\}$. No matter what the min is, $G_i(t+1) = G_i(t) + \gamma_i(t) - a_i(t) \leq G_i(t) + A_i^{max} - a_i(t)$. Since $G_i(t) < \frac{V}{\ln 2}$, $G_i(t+1) < \frac{V}{\ln 2} + A_i^{max} - a_i(t) < \frac{V}{\ln 2} + A_i^{max}$. The bound of $G_i(t)$ holds true. Finally $\overline{G(t)} = \frac{1}{N}\sum_{i=1}^{N} G_i(t) \leq \frac{V}{\ln 2} + \frac{1}{N}\sum_{i=1}^{N} A_i^{max}$.

To prove (21), we first prove the bound of $Q_i(t) \leq \frac{V}{\ln 2} + 2A_i^{max}$. We use induction. When $t = 0$, $Q_i(0) = 0$. The bound is true. Now we assume that the bound is true at t. Next, we prove that the bound holds true at $t+1$. $Q_i(t+1)$ is updated by (1). According to (16), if $Q_i(t) \geq G_i(t)$, $a_i(t) = 0$. So $Q_i(t+1) = Q_i(t) - b_i(t) \leq Q_i(t) \leq \frac{V}{\ln 2} + 2A_i^{max}$ holds true. The last inequality uses the induction assumption at t. If $Q_i(t) < G_i(t)$, $a_i(t) = A_i(t)$. Then $Q_i(t+1) = Q_i(t) - b_i(t) + A_i(t) < G_i(t) - b_i(t) + A_i(t)$. Since $G_i(t) \leq \frac{V}{\ln 2} + A_i^{max}$ holds from the above proof and $A_i(t) \leq A_i^{max}$, $Q_i(t+1) < \frac{V}{\ln 2} + 2A_i^{max}$ holds true. Then $\overline{Q(t)} = \frac{1}{N}\sum_{i=1}^{N} Q_i(t) \leq \frac{V}{\ln 2} + \frac{2}{N}\sum_{i=1}^{N} A_i^{max}$.

Now let us prove (22). Queue $M(t)$ is updated by (2). Whenever the max part returns a zero, $M(t+1)$ is zero and the bound of $M(t)$ holds true. Next, we just need to look at the case when the max part does not return a zero. Again, we use induction. When $t = 0$, $M(0) = 0$. The bound holds. We assume the bound holds at t and then prove that the bound still holds at $t+1$. According to (18), if $P(t) < \max\{\frac{Q_i(t) - u_i M(t)}{u_i}\}$, the requester will not select any worker, i.e., $b_i(t) = 0$. Then $M(t+1) = M(t) - P(t) \leq \max\{\frac{V/\ln 2 + 2A_i^{max}}{u_i}\} + \sum_{i=1}^{N} u_i(\frac{V}{\ln 2} + 2A_i^{max})$ holds true. The last inequality is due to the induction assumption at t. If $P(t) \geq \max\{\frac{Q_i(t) - u_i M(t)}{u_i}\}$, the requester will select workers. If $M(t) > P(t)$, $M(t+1) = M(t) - P(t) + \sum_{i=1} u_i b_i \leq M(t)$ because $-P(t) + \sum_{i=1} u_i b_i \leq 0$ due to the fact that $P(t)$ is all the money the requester has at t to pay for tasks $\sum_{i=1} u_i b_i$ selected from the workers. So the bound still holds at $t+1$ by the induction assumption. If $M(t) \leq P(t)$, then $M(t+1) = \sum_{i=1} u_i b_i$. We know that $b_i(t) \leq Q_i(t)$ as you cannot select more than you have in the queue. Using the bound of $Q_i(t)$ from the above, $M(t+1) \leq \sum_{i=1}^{N} u_i(\frac{V}{\ln 2} + 2A_i^{max}) \leq \max\{\frac{V/\ln 2 + 2A_i^{max}}{u_i}\} + \sum_{i=1}^{N} u_i(\frac{V}{\ln 2} + 2A_i^{max})$. After considering all the cases, we have proved that the bound of $M(t)$ holds true. □



(a) The convergence of A1 (b) $\overline{Q}$ and its upper-bound

(c) $\overline{G}$ and its upper-bound (d) M and its upper-bound

Fig. 1. Evaluation of Algorithm 1 by its upper-bounds

V. Simulations

In this section, we evaluate Algorithm 1 by first comparing its performance with theoretical values and then comparing it with two other variations. The simulations were conducted using Matlab.

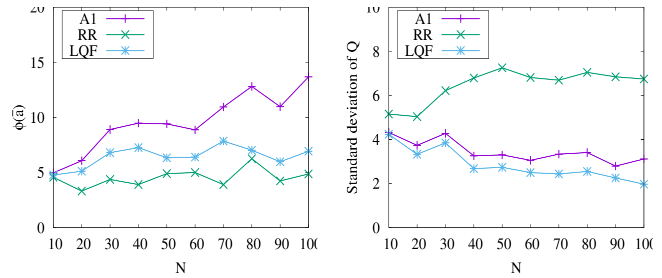
A. Setting

In all the simulations, the maximum time slots $T_{max} = 1000$; A_i^{max} is randomly generated in the range of $[0, 5]$ for each worker; u_i is randomly generated in the range of $[1, 5]$; and P_t is randomly generated in the range of $[0, 100]$. All other parameters are set in each specific simulation below.

B. Comparing with Theoretical Values

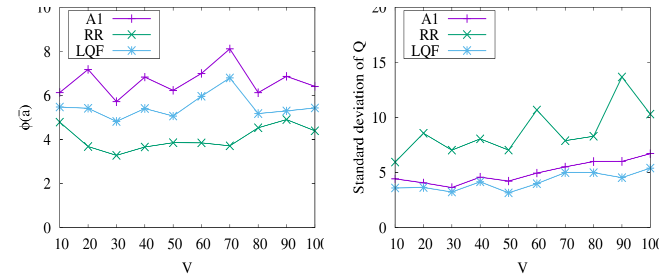
From Theorem (2) we know that, with the increase of V, the proposed algorithm will get closer to the optimal solution to problem $\mathcal{P}1$. In other words, the optimal value produced by Algorithm 1 should converge. In Figure 1(a), we set the number of workers $N = 10$ and varied the value of V from 10 to 100 with a gap of 10 first, and then jumped the V value to 200 and then 300. We can see from the figure that the result really starts to converge after $V = 70$.

According to Theorem (3), there exist upper bounds for all physical and virtual queues. These are demonstrated by the $\overline{Q}$, $\overline{G}$, and M values and their upper bounds in Figs. 1(b)-(d).



(a) $\phi(\overline{a})$ (b) Standard deviation of Q

Fig. 2. Comparison of the algorithms with different Ns



(a) $\phi(\overline{a})$ (b) Standard deviation of Q

Fig. 3. Comparison of the algorithms with different Vs

C. Comparing with other Algorithms

In this simulation, we compared Algorithm 1 (A1) with two variations. We kept the workers' part the same and replaced the requester's optimal task selection (OTS) with round robin (RR) and the longest queue first (LQF), respectively. To be simple, we just call the whole algorithms RR and LQF. In RR, we let the requester pick tasks from workers in a round-robin fashion. In LQF, the requester will choose tasks from workers based on the length of the physical queue Q. The worker with the longest queue will be considered first. The purpose of LQF is to balance the physical queue of the workers.

In the first experiment, we set the value of V to 10 and varied the value of N from 10 to 100 with a gap of 10. We evaluated the optimal value produced by the three algorithms and the standard deviation of their physical queue Q. Given that virtual queues are only utilized for problem-solving, we solely compare the physical queues of the algorithms in this simulation. Our goal is to determine which algorithm can generate the best optimization value for problem $\mathcal{P}1$, and which algorithm has the most balanced queue. We ran the algorithms 1000 times and averaged the results. The simulation results are shown in Figs. 2(a) and (b).

From the figures, we can see that A1 obtains the best optimization value for problem $\mathcal{P}1$, LQF is the second, and RR is the last. In queue balance, LQF is the best because its main focus is the queue length, RR is the worst, and A1 is close to LQF. The results indicate that A1 not only produces the best solution but also balances its queue well.

In the second experiment, we set the value of N to 20 and varied the value of V from 10 to 100 with a gap of 10. We compared the optimal value produced by the three algorithms and the standard deviation of their physical queue Q. We ran the algorithms 1000 times and averaged the results. The simulation results are similar to the above and are depicted in Figs. 3(a) and (b).

VI. Conclusion

In this paper, we formulated a stochastic optimization problem addressing both the long-term incentive issue and the workers' time-average output maximization issue. We solved the problem using the Lyapunov technique and turned the solution into interactive but independent optimization decisions on the side of the workers and the requester in each time slot. To evaluate the performance of our solution, we conducted a theoretical analysis first and then simulations to compare our solution with theoretical values and two other variations. Analysis and simulation results have shown that our solution can maximize workers' time-average output while ensuring fairness to retain them over time. In the future, we will expand our model to include factors like task quality, worker capabilities, competition among requesters, and more.

References

[1] Crowdsourcing. https://en.wikipedia.org/wiki/Crowdsourcing.

[2] Amazon mechanical turk. http://mturk.com.

[3] H. Zhou, X. Chen, S. He, J. Chen, and J. Wu. Draim: a novel delay-constraint and reverse auction-based incentive mechanism for wifi offloading. *IEEE Journal on Selected Areas in Communications*, 38(4):711—-722, 2020.

[4] Y. Wang, Y. Gao, Y. Li, and X. Tong. A worker-selection incentive mechanism for optimizing platform-centric mobile crowdsourcing systems. *Computer Networks*, 171:107144, 2020.

[5] Z. Cai, Z. Duan, and W. Li. Exploiting multi-dimensional task diversity in distributed auctions for mobile crowdsensing. *IEEE Transactions on Mobile Computing*, 2020.

[6] J. S. Lee and B. Hoh. Sell your experiences: a market mechanism based incentive for participatory sensing. In *IEEE PerCom*, pages 60–68, 2010.

[7] L. G. Jaimes, I. Vergara-Laurens, and A. L. Miguel. A location-based incentive mechanism for participatory sensing systems with budget constraints. In *IEEE PerCom*, pages 103–108, 2012.

[8] H. W. Wang, S. Guo, J. N. Cao, and M. Y. Guo. MELODY: A Long-term Dynamic Quality-aware Incentive Mechanism for Crowdsourcing. *IEEE Transactions on Parallel and Distributed Systems*, 29(4), 2018.

[9] C. X. Chi, Y. J. Wang, Y. S. Li, and X. R. Tong. Multistrategy repeated game-based mobile crowdsourcing incentive mechanism for mobile edge computing in internet of things. *Wireless Communications and Mobile Computing*, 2021:6695696:1–6695696:18, 2021.

[10] N. Zhao, Q. X. Wan, J. L. Chen, and M. H. Wu. Dynamic incentive mechanism in mobile crowdsourcing networks by combining reputation and contract theory. *International Journal of Distributed Sensor Networks*, 18(6), 2022.

[11] L. Gao, F. Hou, and J. W. Huang. Providing long-term participation incentive in participatory sensing. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 2803–2811, 2015.

[12] J. J. Sun and N. Z. Liu. Frugal incentive mechanism in periodic mobile crowdsensing. *International Journal of Communication Systems*, 2018.

[13] S. P. Wang, Q. Z. Li, L. Z. Cui, Y. L. Jiang, Z. Q. Shen, and H. Yu. Csp-rm: Reputation management decision support for crowdsourcing service providers. *IEEE International Conference on Web Services (ICWS)*, pages 161–169, 2023.

[14] M. J. Neely. *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Synthesis Lectures on Communication Networks. Morgan & Claypool Publishers, 2010.

[15] J. L. W. V. Jensen. Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta Mathematica*, 30:175–193, 1906.

[16] Knapsack problem. https://en.wikipedia.org/wiki/Knapsack_problem.

[17] B. S. Thomson and A. M. Bruckner. *Elementary Real Analysis*. CreateSpace, 2008.