Incentivizing Distributive Fairness for Crowdsourcing Workers

Chenxi Qiu
Department of Computer Science
Rowan University
Glassboro, New Jersey
qiu@rowan.edu

Anna Squicciarini and Benjamin Hanrahan
College of Information Science and Technology
The Pennsylvania State University
University Park, Pennsylvania
{asquicciarini,bvh10}@ist.psu.edu

ABSTRACT

In a crowd market such as Amazon Mechanical Turk, the remuneration of Human Intelligence Tasks is determined by the requester, for which they are not given many cues to ascertain how to "fairly" pay their workers. Furthermore, the current methods for setting a price are mostly binary – in that, the worker either gets paid or not – as opposed to paying workers a "fair" wage based on the quality and utility of work completed. Instead, the price should better reflect the historical performance of the market and the requirements of the task. In this paper, we introduce a game theoretical model that takes into account a more balanced set of market parameters, and propose a pricing policy and a rating policy to incentivize requesters to offer "fair" compensation for crowdsourcing workers. We present our findings from applying and developing this model on real data gathered from workers on Amazon Mechanical Turk and simulations that we ran to validate our assumptions. Our simulation results also demonstrate that our policies motivate requesters to pay their workers more "fairly" compared with the payment set by the current market.

CCS CONCEPTS

• **Social and professional topics**; • **Theory of computation** → *Mathematical optimization*; • **Computing methodologies** → *Modeling and simulation*;

KEYWORDS

Crowdsourcing; human intelligence task; fair compensation

ACM Reference Format:
Chenxi Qiu and Anna Squicciarini and Benjamin Hanrahan. 2019. Incentivizing Distributive Fairness for Crowdsourcing Workers. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019,* IFAAMAS, 9 pages.

1 INTRODUCTION

In a crowd market such as Amazon Mechanical Turk (AMT), requesters are solely responsible for setting the compensation for their workers. However, there are virtually no cues provided to requesters for how to determine the price for their tasks, which can be difficult [5]. It is even more difficult, perhaps impossible, for requesters to set a "fair" compensation for their workers that takes into consideration market trends, available workers, and the market's historical performance on tasks with similar characteristics. In this work we define fairness in the distributive sense [10], where our goal is to motivate requesters to minimize the disparity among workers and not to favor one worker over the other. Distributive fairness [10] focuses on the equity principle, and can be defined as consistency of all workers' input/output ratios, where in this case input is labor and output is compensation.

However, this is made somewhat difficult as the quality and effort of individual workers (or *contributors*) may vary considerably in a truly collective task, as contributors have various degrees of experience, expertise, and interest in completing tasks. The current approach of many platforms (e.g. AMT) to combat this variance in performance is either to ignore it (using a first-come-first-serve model) or to penalize poor performers by blocking future access to the labor market or create higher paying bands of tasks, the combination of which creates difficult onboarding paths and an increased sense of inequity and unfairness among the workers [28, 30].

This idea of creating a more "fair" environment is important, as researchers have found that workers are more likely to engage with crowdsourcing tasks if they feel that they are being treated fairly, and their remuneration (e.g. payment) is commensurate with their contributions [9–11]. Given that the success of an online crowdsourced market is, in large part, the byproduct of the individual workers' efforts and the environment within which they exert these efforts, it follows that promoting fair payment and treatment of workers is of the utmost importantace for more effective, capable crowdsouring system. That is, if workers believe that the environment that they are working within is fair, we can expect not only an improvement in the quality of contributors' work [6, 7, 23], but also an increase in retaining and recruiting additional workers [10, 11].

In this paper, we address this problem by introducing a game theoretical model that takes into account a more balanced set of market parameters. Particularly, to achieve high distributive fairness, we suggest a pricing policy for requesters that takes into account workers' quality, including their historical performance, interests, session details, etc. On the other hand, considering that some requesters may purposely decrease workers' compensation, e.g., by rejecting workers' submitted tasks, we allow platforms to rate each requester based on the percentage of tasks the requester approve from workers, where the rating is visible to workers. Then, we consider two types of interactions, both of which can be modeled as a Stackelberg game, a strategic game where a leader makes decision first and followers move sequentially:

1) *Requesters vs. workers.* Requesters (as leaders) first determine their pricing policy and the average approval ratio to workers, and then workers (as followers) decide the effort level to take to complete tasks.

2) *Crowdsourcing platform vs. requesters.* The platform (as the leader) publishes their rating policy to requesters, and then requesters (as followers) specify their pricing policy as well as the approval ratio.

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada.

Our framework enables not only determining conditions for *distributive fairness* to hold, but also helps define *how* requester should determine the compensation paid for their tasks according to worker performance. According to distributive fairness [10], payment accounts for individual performance, but maintains low disparity among individuals' pays. Here, the intuition is that higher variance means higher disparity, and hence decreased fairness. Note that in extreme cases, being treated equally (i.e. lower variance among workers) may denote group unfairness, i.e. every worker is treated poorly. Our rating policy can help avoid this since requesters with low payment on average will have low rating and hence are less competitive in the market.

According to both workers' and requesters' performance, our objective is to find 1) a *pricing policy* such that workers' qualities are accounted for in the payment scheme and 2) an *optimal rating policy* to motivate requesters to compensate workers so that maximum distributive fairness among workers is enabled.

According to the two types of interactions in the game theoretical framework, the problem of finding the optimal rating policy can be also formulated as a *three level optimization problem*, which generally is NP hard. Hence, we propose a time efficient solution by resorting to primal decomposition [34] and approximation. For theoretical interests, we also derive a theoretical bound of workers' payment variance to check how close our solution is to the optimal.

We note that our approach is novel in many respects. Previous work into fairness for workers in the digital workplace has been focused on the impact of working *conditions* on individual workers, and has not yet been adopted *en masse* by platform maintainers [21, 24, 28], whereas our approach is proposing a pricing policy and a rating policy where fairness is both defined and enforced (in a manner of speaking) by the market and its participants.

We present our findings from applying and developing this model on real data gathered from workers on AMT and simulations that we ran to validate our assumptions. As the prices driven by our rating policy reflects the prices set by the market (which have been found to be problematic [17, 29]), we discuss specific findings about fairness as it exists in the current market and how our system might affect fairness overtime.

2 RELATED WORK

Fairness in Crowdwork. Promoting fair treatment is of the utmost importance for more effective, capable crowdsourcing systems. If contributors believe that the environment that they are working within is fair, we can expect not only an improvement in the quality of contributors' work [6, 7, 23] (which is beneficial for the requester), but also an increase in retaining and recruiting additional contributors [10, 11] (which is beneficial for the platform owner). The notion of fairness has been applied widely, notably in social choice theory, game theory, economics, and law [6–12, 20, 26, 39]. For example, Xu et al. [42] focused on an application of "online deliberation" and they proposed to crowd-source moderation work to contributors themselves to increase perceived procedural fairness. However, as Sauermann and Cohen [37] suggested, although the intensity of effort has a strong effect on contributors' ideas, the character of effort seems to be more important in generating ideas. In context of crowdsourcing, prior studies focus mainly on the quantity of effort in online communities [42] and largely ignore the multiple types of effort, e.g. time consumption, creativity, expertise in multiple areas. On the other hand, contributors' productivity is likely to depend upon the nature of the task, for instance, whether it requires expertise or time, or even demands greater creativity [15, 37]. Therefore, rather than motivating contributors to simply devote more time, a better strategy is to motivate contributors to bring more of themselves and their expertise to the work.

Fairness has been an oft discussed topic in the field of crowdsourcing, and each field brings with it its own perspective in defining what fairness means. Franke et al. [11] first highlighted the importance of fairness in the context of crowdsourcing where they demonstrate that users' expectations on fairness have a strong impact on their decision to participate in task solving. Faullant et al. [10] followed up by further exploring how different types of fairness, i.e. distributive fairness (a fair amount and distribution of the offered reward) and procedural fairness (fair procedures/process to determine the winners), affect contributors' intentions to participate in future crowdsourcing tasks as well as their loyalty towards the platform. Recently, numerous works have also started to focus on designing strategies to improve contributors' perceived fairness [13, 42]. Complementing this perspective, when fairness is discussed in CSCW literature, they focus primarily on the working conditions for the workers of crowdsourcing [22, 24, 28]. However, there has been a few efforts to help requesters estimate the time required for their task [5] – which of course impacts price.

Task Matching or Pricing for Fairness. In the recent years, several studies related to matching and pricing in Crowdsourcing have been proposed. These studies aim to address workers' (or contributors') quality issues [1, 4, 14, 19, 25, 27, 33, 38, 40, 41, 43–47]. For example, in [44], Yin et al. proposed an approach to dynamically control whether and when to place a bonus in a crowdsourcing working session to improve requesters' utility, where their policy provides more bonus opportunities for workers with lower accuracy. In [19], Hu et al. proposed an optimal posted-price mechanism for microtask crowdsourcing that requires fewer inputs compared with many existing approaches. Considering that requesters always want all their tasks to be completed even though some of their tasks are less interesting to contributors, Kobren et al. [25] presented a task allocation approach by balancing the task value with the likelihood of a contributor dropping out after being assigned with the task.

Notably, most studies to date consider the task matching or pricing problem from the perspective of requester, i.e., aim to improve of contributors' quality of answers, but neglect the benefits of contributors. Some recent work has looked into solving problems from the perspective of contributors [1, 14, 25, 38, 45, 46]. These works either focus on improving contributors' benefits (including utilities [1, 14, 38], efficiency [14, 45, 46], and privacy [14]), or target on balancing the benefits between requesters and contributors [25]. For example, Schnitzer et al. [38] suggested to develop a new task recommendation system based on contributors' own preferred types of benefits. Gong et al. [14] targeted a single contributor and pointed out fundamental trade-offs among utility, privacy, and efficiency of the contributor. Authors also proposed a flexible optimization framework that can be adjusted to any desired trade-off point. Despite some similarities, none of these works aim to improve contributors'

perceived group fairness. As a result, the actual algorithms and methods are fundamentally different than the ones proposed in this paper: fairness trascends individual gains and utilities and we maximize it as a collective metric. Yet, it plays an important role in encouraging contributors' long-term engagement and improving contributors' performance.

To our knowledge, one of the few efforts looking into encouraging group fairness is from Mengash and Brodsky [31], [32], and [27]. In all these approaches, the proposed approach aims at developing recommendation algorithms- outside the specifics of the crowdsourcing domain. Albeit similar in looking at group criteria, these works tackle a problem that is fundamentally different from ours as they aim to recommend a service to a group of users, while our problem needs to address fairness by matching multi-tasks to multi-contributors.

3 PROBLEM FORMULATION

In this section, we first introduce the model, including notations and assumptions that will be used throughout the paper (Section 3.1 and Section 3.2). Based on the model, we then formally formulate the problem (Section 3.3).

3.1 System Model

Requesters vs. workers. We consider a scenario composed of M requesters $\{1, 2, ..., M\}$ and N workers $\{1, 2, ..., N\}$. Each individual worker in the system is defined by several dimensions, including their historical performance and skills/qualification.

In order to estimate a pricing policy for a given task, the first dimension that we evaluate is worker performance. Specifically, we use worker's *approval ratio* as a proxy of the worker's performance, which is the number of acceptable tasks submitted by the worker that the requester deems worthy of payment. This is denoted as $p_{i,j}$ (for a worker i and requester j) and is defined as

$$p_{i,j} = \frac{\text{number of tasks completed by worker } i \text{ and approved by the requester } j}{\text{number of tasks assigned by the requester } j}. \quad (1)$$

Then, we describe the pricing policy from each requester j by a function $f_j(p_{i,j})$, which is a map from workers' approval ratio to their compensation.

Here, f_j is assumed as a concave function, upper bounded by the *full payment* z, which is the maximum amount a user can earn for the tasks from requester j. In addition, f_j defines:

1) The lowest acceptable approval ratio w. The worker earns no payment if his approval ratio is lower than w, i.e., $f(p_{i,j}) = 0$ when $p_{i,j} \in [0, w)$;
2) A required approval ratio u and the worker can earn the full payment 1 if the worker approval ratio achieves u, i.e., $p_{i,j} \in [u, 1]$.

Also, we assume f_j is a quadratic function in interval $[w, u)$ and is continuous in the whole region $[0, 1]$. According to the above assumptions, we derive the pricing function f_j as:

$$f_j(p_{i,j}) = \begin{cases} 0 & p_{i,j} \in [0, w) \\ \frac{u^2}{u^2-w^2} - \frac{(p_{i,j}-u-w)^2}{u^2-w^2} & p_{i,j} \in [w, u) \\ 1 & p_{i,j} \in [u, 1] \end{cases} \quad (2)$$

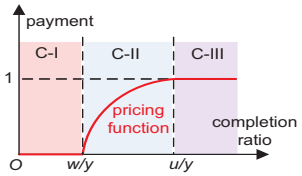


Figure 1: Pricing function.

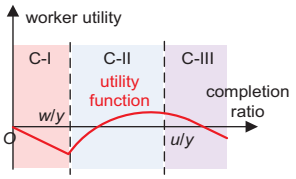


Figure 2: Utility function.

We assume that each worker i's approval ratio is affected by their effort level, i.e. not all submitted tasks may be approved, based on their quality. For each requester j, we use the term *completion ratio* $x_{i,j}$ as the ratio of the number of tasks completed by worker i over the number of tasks originally started or entered by worker i (e.g. worker starts 10 hits but only completes 8, completion ratio would be 80.0%, and if 5 gets approved, 62.5% approval rate). We assume that the approval ratio $p_{i,j}$ increases linearly with the completion ratio $x_{i,j}$, which can be described mathematically by:

$$p_{i,j} = y_j x_{i,j} \ (y_j \geq 0). \quad (3)$$

Here, we call y_j the *approval coefficient*, a decision variable made by requester j. y_j indicates the requester's willingness to give higher approval ratio to workers. For example, some requesters may be strict, and grant low approval ratio to minimize costs. Consequently, we rewrite the pricing function defined by Equation (2), where the curve of the function is depicted in Figure 1:

$$f_j(x_{i,j}; y_j) = \begin{cases} 0 & y_j x_{i,j} \in [0, w) \\ \frac{u^2}{u^2-w^2} - \frac{(y_j x_{i,j}-u-w)^2}{u^2-w^2} & y_j x_{i,j} \in [w, u) \\ 1 & y_j x_{i,j} \in [u, 1] \end{cases} \quad (4)$$

Platform vs. requester. Each requester j can determine tasks' payment by choosing his approval coefficient y_j. However, requesters may purposely decrease the approval ratio for workers to reduce their cost. For example, some requesters take advantage of asymmetrical information access in crowdsourcing platforms [22, 28] by rejecting work and keeping the results. Workers are concerned about this and spend a lot of time finding 'fair' requesters [16].

Accordingly, to ensure the fairness between workers and requesters, the platform is allowed to rate each requester "reputation" according to their approval coefficient and display each requester's reputation to workers. Then, the workers are able to select requesters that are willing to provide higher approval ratio and hence higher compensation for their work. We describe the platform rating policy by a concave function $h(y_j)$, which is a map from approval coefficient to reputation rating. Based on the definition of approval coefficient (Equation (3)), the platform can obtain the approval coefficient of each requester j by resorting to linear regression [36], where the training data includes 1) the requester's historical approval ratio y_j and 2) workers' historical completion ratio for this requester, $x_{1,j}, ..., x_{N,j}$.

We assume that h is a quadratic function. We normalize the rating of all requesters to the interval $[0, 1]$, and let $h(0) = 0$ and $h(1) = 1$. According to the above assumptions, we can derive that $h(y_j) = zy_j^2 + (1 - z)y_j$, where z, called the *rating policy parameter*, is a decision variable that can be adjusted by the platform. Note that higher z decreases the parabola's turning point (and hence y), leading to higher disparity among workers' payment.

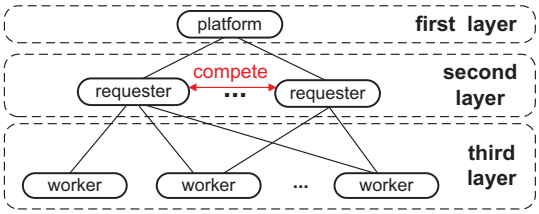


Figure 3: Three layers of the system.

In addition, we let $\overline{h}(\mathbf{y}, z)$ represent the average rating of all requesters:

$$\overline{h}(\mathbf{y}, z) = \frac{\sum_{j=1}^{M} h(y_j, z)}{M}, \tag{5}$$

where $\mathbf{y} = [y_1, ..., y_M]$, and we let $\Delta h_j(\mathbf{y}, z) = h(y_j, z) / \overline{h}(\mathbf{y}, z)$ to reflect how much higher j's rating is with respect to the average. As the requester with higher rating is more likely to attract workers with qualified skills, each requester j tends to improve their rating by maintaining their approval coefficient in a reasonably high level.

3.2 The Objectives of Different Roles

As Figure 3 shows, we can decompose different roles in the system into three layers:

Workers (layer 3). Each worker i puts a certain effort level $x_{i,j}$ (measured by the completion ratio as discussed in the prior section). The worker's objective is to maximize his/her possible payment $f(x_{i,j}; y_j)$ and to minimize the effort level:

$$\max \quad f_j(x_{i,j}; y_j) - \beta_i x_{i,j} \tag{6}$$

$$\text{s.t.} \quad 0 \le x_{i,j} \le 1, \forall i, j \tag{7}$$

where β_i, called *laziness coefficient*, is the weight assigned to user i's effort level. β_i indicates worker i's willingness to put effort in submitting correct tasks, i.e., higher β_i implies less effort to be taken by the worker.

Requester (layer 2). First, each requester j aims to maximize the overall performance from workers, which can be represented by the sum of completion ratios of all workers: $\sum_{i=1}^{N} x_{i,j}$.

On the other hand, as workers tend to choose completing tasks from requesters with higher rating, each requester j also aims to improve his/her rating to compete with other requesters. Here, we assume each requester tries to exceed the average rating of all requesters $\overline{h}(\mathbf{y}, z)$ as much as possible. Then, the objective function of requester can be written as

$$\max \sum_{i=1}^{N} x_{i,j} + \lambda_j \Delta h_j(\mathbf{y}, z) \tag{8}$$

where λ_j is a weight assigned to $\Delta h_j(\mathbf{y}, z)$, indicating the willingness of requester j to improve his/her rating on the platform.

In addition, we assume each requester j has a budget limit Γ_j such that the total payment to all the workers from requester cannot exceed Γ_j:

$$\sum_{i} f_j(x_{i,j}; y_j) \le \Gamma_j \tag{9}$$

and we let Ω_j represent the feasible region of $[y_j, \mathbf{x}_j]$:

$$\Omega_j = \left\{ [y_j, \mathbf{x}_j] \in \mathbb{R}^{N+1} \;\middle|\; \begin{array}{l} \sum_i f_j(x_{i,j}; y_j) \le \Gamma_j \\ 0 \le x_{i,j} \le 1, 0 \le y_j \le 1, \forall i, j \end{array} \right\}. \tag{10}$$

Platform (layer 1). As demonstrated in [6, 7, 10, 11, 23], if workers believe that the environment that they are working within is fair, it is expected that not only an improvement in the quality of their work, but also an increase in retaining and recruiting additional workers. As the success of a crowdsourcing platform is highly due to a high number of available workers and the quality of their work, it is of great importance to promote fair payment and treatment of workers on the platform. Accordingly, we set the objective of the crowdsourcing platform as to maximize *fairness* among the workers, which is specifically defined as minimizing the variance of workers' payments:

$$\min \sum_{j=1}^{M} \sum_{i=1}^{N} \left[f(x_{i,j}; y_j) - \overline{f}_j \right]^2 \tag{11}$$

where $\overline{f}_j = \frac{\sum_{j=1}^{N} f_j(x_{i,j}; y_j)}{N}$ is the average compensation paid to all the workers from the requester j.

3.3 Problem formulation

According to the above assumptions, we have two types of interactions across the three layers: the interaction between workers and requesters, and the interaction between the platform and requesters. We can model both types of interactions as a Stackelberg game [35], a strategic game where the leader makes the decision first, and the followers observe the leader's action and moves in a way that is personally optimal.

More precisely, for the interaction between each requester j and their workers, we model the requester as a leader, who first determines the pricing policy and the approval coefficient y_j for his/her workers. Then, the worker determines how much effort $x_{i,j}$ needs to take to reach or approximate the full payment.

For the interaction between the platform and requesters, we model the platform as a leader to determine the rating policy h for all requesters, and then each requester determines the pricing policy as well as the approval coefficient.

Finally, our *payment fairness maximization (PFM)* problem, can be written as a 3-level programming problem:

$$\begin{array}{ll} \min & \sum_{j=1}^{M} \sum_{i=1}^{N} \left[f(x_{i,j}; y_j) - \overline{f}_j \right]^2 \text{ (Layer 1)} \\ \text{s.t.} & z \in [0, 1] \\ & \quad \begin{array}{ll} \max & \sum_{i=1}^{N} x_{i,j} + \lambda_j \Delta h_j(\mathbf{y}, z) \text{ (Layer 2)} \\ \text{s.t.} & [y_j, \mathbf{x}_j] \in \Omega_j \end{array} \\ & \quad \boxed{\begin{array}{ll} \max & f_j(x_{i,j}; y_j) - \beta_i x_{i,j} \text{ (Layer 3)} \\ \text{s.t.} & x_{i,j} \in [0, 1] \end{array}} \\ & \quad i = 1, ..., N, \\ & j = 1, ..., M \end{array}$$

The decision variables of the optimization problem in layer 1, 2, and 3 are z, y_j, and $x_{i,j}$ ($i = 1, ..., N$ and $j = 1, ..., M$), respectively. The above hierarchical relationship results from the fact that the optimization related to the workers' behavior is taken as a constraint when the requester makes the decision, and similarly, the optimization related to the requesters' behavior is taken as a constraint for the platform's decision. The objective of PFM is to find out the optimal rating policy parameter z for the platform to minimize the distributive fairness among workers.

4 ALGORITHM DESIGN

To find the optimal rating policy parameter z (in layer 1), we need to analyze how requesters determine their approval coefficient given

each possible z (layer 2), which in turn requires us to derive workers' response given requesters' approval coefficient (layer 3). Hence, in this section, we start solving PFM by analyzing the best response of each worker given the requester's approval coefficient (in Section 4.1), where the main result is given in *Theorem 4.1*. We take the result of Theorem 4.1 as a constraint of workers' and requesters' behavior, which can replace the 3rd layer optimization in PFM and hence reduce PFM to a Bi-Level Programming (BiP) problem.

BiP in general is NP-hard [2]. As a solution, we propose a time-efficient and scalable approach by resorting to primal decomposition and approximation in Section 4.2. For theoretical interests, we also derive a lower bound of workers' payment variance to check how close our solution can achieve the optimal.

4.1 Best Response Analysis of Workers

According to the pricing function described by Equation (2), there are different formulas when $y_j x_{i,j} \in [0, w]$, $y_j x_{i,j} \in [w, u]$, and $y_j x_{i,j} \in [u, 1]$. Hence, we discuss the best response of each worker i in the following three cases (as shown in Figure 2):

C-I. When $x_{i,j} \in \left[0, \frac{w}{y_j}\right)$, $f\left(x_{i,j}; y_j\right) - \beta_i x_{i,j} = -\beta_i x_{i,j}$, and hence the worker's utility is maximized when $x^*_{i,j}(I) = 0$;

C-II. When $x_{i,j} \in \left[\frac{w}{y_j}, \frac{u}{y_j}\right)$, as Figure 4 shows, the best response of worker i depends on whether the *turning point* $x'_{i,j}$ of the parabola $f\left(x_{i,j}; y_j\right) - \beta_i x_{i,j}$ is in $\left(-\infty, \frac{w}{u}\right)$, $\left[\frac{w}{u}, \frac{u}{y_j}\right)$, or $\left[\frac{u}{y_j}, \infty\right)$. Specifically,

$$x^*_{i,j}(II) = \begin{cases} \frac{w}{y_j} & x'_i \in \left(-\infty, \frac{w}{u}\right) \text{ or } y_j \in \left[0, \frac{\beta_i(u^2-w^2)}{2u}\right) \\ \frac{u+w}{y_j} - \frac{\beta_i(u^2-w^2)}{2y_j^2 z} & x'_i \in \left[\frac{w}{u}, \frac{u}{y_j}\right) \\ & \text{or } y_j \in \left[\frac{\beta_i(u^2-w^2)}{2u}, \frac{\beta_i(u^2-w^2)}{2w}\right) \\ \frac{u}{y_j} & x'_i \in \left[\frac{u}{y_j}, \infty\right) \text{ or } y_j \in \left[\frac{\beta_i(u^2-w^2)}{2w}, \infty\right) \end{cases} \tag{12}$$

Note that when $y_j \in \left[0, \frac{\beta_i(u^2-w^2)}{2u}\right)$, $f\left(x_{i,j}; y_j\right) - \beta_i x_{i,j} = -\beta_i x_{i,j} < 0$, which is dominated by the best response in C-I.

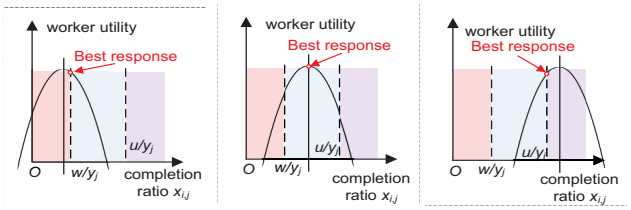
C-III. When $x_{i,j} \in \left[\frac{u}{y_j}, \frac{1}{y_j}\right]$, $f\left(x_{i,j}; y_j\right) - \beta_i x_{i,j} = z - \beta_i x_{i,j}$, which monotonically decreases with the increase of $x_{i,j}$. Therefore, the best response of worker i should be when $x_{i,j}$ is minimized, i.e., $x^*_i(III) = \frac{u}{y_j}$, and the corresponding utility of worker i is $f\left(x_{i,j}; y_j\right) - \beta_i x_{i,j} = 1 - \frac{\beta_i u}{y_j}$.

After analyzing worker i's best response in the above cases, we can eventually derive that

$$x^*_{i,j} = \arg \max_{l=I,II,III} \left\{ f\left(x^*_{i,j}(l); y_j\right) - \beta_i x^*_{i,j}(l) \right\} \tag{13}$$

Here, we note that if part of workers have their best response equal to 0, other workers have to take more effort to satisfy the *overall approval ratio constraint*, leading to a higher disparity among workers' payment. Hence, we try to avoid $x^*_{i,j} = 0$ for each worker i in our pricing policy.

Particularly, we set constraints such that the best response in C-I $x^*_{i,j}(I)$ are dominated by the best responses in the other two cases, $x^*_{i,j}(II)$ and $x^*_{i,j}(III)$. To ensure that C-II dominates C-I, A1-A3 have to be satisfied:



(a) $x'_i \in \left(-\infty, \frac{w}{u}\right)$ (b) $x'_i \in \left[\frac{w}{u}, \frac{u}{y_j}\right)$ (c) $x'_i \in \left[\frac{u}{y_j}, \infty\right)$

Figure 4: The three subcases of the best response in Case II.

A1: $y_j \notin \left[0, \frac{\beta_i(u^2-w^2)}{2u}\right)$, otherwise C-II has worse best response than that of C-I. To satisfy A1, we have

$$y_j > \max_{i,j} \left\{ \frac{\beta_i \left(u^2 - w^2\right)}{2u} \right\}. \tag{14}$$

A2. When $y_j \in \left[\frac{\beta_i(u^2-w^2)}{2u}, \frac{\beta_i(u^2-w^2)}{2w}\right)$, worker i's utility $f\left(x^*_{i,j}(II); y_j, z\right) - \beta_i x^*_{i,j}(II)$ should be higher than 0, from which we can derive that

$$y_j < \frac{\beta_i \left(u + w\right)\left(u^2 - w^2\right)}{u^2}. \tag{15}$$

A3. When $y_j \in \left[\frac{\beta_i(u^2-w^2)}{2w}, \infty\right)$, the worker's best response is $x^*_{i,j}(II) = u/y_j$. To ensure the utility of worker i to be higher than 0, we have

$$f\left(x^*_{i,j}(II); y_j, z\right) - \beta_i x^*_{i,j}(II) = 1 - \frac{\beta_i u}{y_j} > 0 \Rightarrow y_j > \beta_i u. \tag{16}$$

To ensure that C-III dominates C-I, Equation (16) still needs to be satisfied, as in this case the worker's best response is still $x^*_{i,j}(III) = u/y_j$.

Eventually, we can summarize that to exclude C-I for worker i:
1) A1 has to be satisfied,
2) Either A2 (when worker i gets partial payment) or A3 (when worker i gets full payment) should be satisfied. By combining C-II and C-III, we eventually obtain Theorem 4.1.

Theorem 4.1. *Given the approval coefficient y_j provided by requester j, the best response a worker i can be represented by*

$$x^*_{i,j} = g\left(y_j; \beta_i\right) = \begin{cases} \frac{u+w}{y_j} - \frac{\beta_i(u^2-w^2)}{2y_j^2} & y_j \in \left[0, \frac{\beta_i(u^2-w^2)}{2w}\right) \\ \frac{u}{y_j} \text{ (full payment)} & y_j \in \left[\frac{\beta_i(u^2-w^2)}{2w}, 1\right] \end{cases} \tag{17}$$

In what follows, we let $\Omega_j(y_j)$ be a projection of Ω_j onto requester j's response region: $\Omega_j(y) = \left\{y | [y, g(y; \beta_1), ..., g(y; \beta_N)] \in \Omega_j\right\}$.

4.2 Approximation Algorithm for BiP

We approximate the BiP problem by only considering a set of discrete points in z's feasible region, $z^0, z^1, ..., z^L$, where each $z^l = \frac{l}{L}$ $(l = 0, 1, ..., L)$. Then, in the first step, given $z = z^l$, we derive the corresponding optimal approval coefficients for all requesters y^l_1, y^l_2, ..., y^l_M by solving the 2nd layer optimization in PFM, and then obtain the corresponding payment variance u^l_{var} from workers (by Equation (11)).

In *Step 2*, after collecting all u^1_{var}, ..., u^L_{var}, we find the index $l_{\min}$ that has the minimum utility variance:

$$l_{\min} = \arg \min_l u^l_{\text{var}}, \tag{18}$$

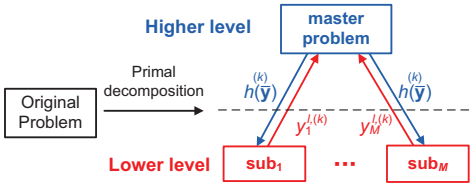


Figure 5: Primal Decomposition.

where the corresponding $z^{l_{\min}}$ is the final result. In what follows, we introduce the details how we derive each y_j^l.

Problem Decomposition. We note that, in the objective functions of the *2nd layer optimization in PFM* (or *PFM-2* for short), the decision variables $y_1^l, ..., y_M^l$ are coupled, i.e., requesters need to compete with each other. However, if we calculate the optimal values of y_1^l, ..., y_M^l together, it will generates an extremely high computation cost, especially for large scale platforms with millions of users.

To improve the scalability of our solution, we apply *primal decomposition (PD)* to decompose PFM, where PD is a classical method in combinatorial optimization and has been widely applied to distributed and parallel computation [34]. The basic idea of PD is to decompose the original large problem into distributively solvable subproblems which are then coordinated by a high-level master problem [34]. A PD is appropriate when the problem has coupling variables such that, when fixed to some value, the rest of the optimization problem decouples into a set of independent subproblems. Particularly, in PFM-2, we find that if $\overline{h}(\mathbf{y}, z)$ is fixed, the problem will be decomposed to a set of subproblems: sub_1, ..., sub_M, where sub_j ($j = 1, 2, ..., M$) is for each requester j and is defined as:

$$\max \quad \sum_{i=1}^{N} x_{i,j} + h_j\left(y_j^l, z^l\right) / \overline{h}(\mathbf{y}, z) \tag{19}$$

$$\text{s.t.} \quad [y_j^l, \mathbf{x}_j] \in \Omega_j \text{ and Equation (17) is satisfied.} \tag{20}$$

Therefore, we can separate PFM-2 into two levels, as Figure 5 shows. In the lower level, we have all the subproblems sub_j ($j = 1, 2, ..., M$). In the higher level, we let the master problem iteratively update the coupling variables $\overline{h}(\mathbf{y}, z)$ by collecting y_j from each sub_j. More precisely, we initialize $\overline{h}^{(0)}(\mathbf{y})$ by a constant h_0. In each iteration k, each sub_j first calculates $y_j^{l,(k)}$ based on $\overline{h}^{(k-1)}(\mathbf{y})$ in iteration $k-1$, and send $y_j^{l,(k)}$ back to the master problem. The master problem then derives $\overline{h}^{(k)}(\mathbf{y})$ given $y_j^{l,(k)}$ ($j = 1, ..., M$). This process is repeated until $\left|\overline{h}^{(k)}(\mathbf{y}) - \overline{h}^{(k-1)}(\mathbf{y})\right| < \epsilon$, where $\epsilon > 0$ is a threshold to check whether $\overline{h}^{(k)}(\mathbf{y})$ has converged.

Algorithm for sub_j. According to Theorem 4.1, y_j^l will fall in different intervals $\left[0, \frac{\beta_i(u^2-w^2)}{2w}\right)$ and $\left[\frac{\beta_i(u^2-w^2)}{2w}, 1\right]$ when worker i gets partial payment and full payment, respectively. Hence, the constraints for each y_j^l will be different when different workers get full payment. Therefore, we discuss the solution of each sub_j by considering the cases when different workers get full payment.

According to Equation (17), for any pair of workers i and k with $\beta_i < \beta_k$, if worker l gets full payment (implying that $y_j^l \geq \frac{\beta_k(u^2-w^2)}{2w}$), then worker i also gets the full payment, since

$$y_j^l \geq \frac{\beta_k\left(u^2-w^2\right)}{2w} \geq \frac{\beta_i\left(u^2-w^2\right)}{2w}. \tag{21}$$

This observation motivates us to first rank workers by increasing β values, and then discuss the solution by considering when the first m workers get full payment respectively. Let $y_j^{l*}(m)$ be the optimal solution when exactly m workers get the full payment. Then, after deriving each $y_j^{l*}(m)$ ($m = 0, 1, ..., N$), we select $y_j^{l*}(m)$ that minimizes the variance of all workers' payment as the output of the algorithm.

Without loss of generality, we assume that $\beta_1 < \beta_2 < ... < \beta_N$. We divide the region of y into the following $N+1$ intervals: $\left[0, \frac{\beta_1(u^2-w^2)}{2w}\right]$, $\left(\frac{\beta_1(u^2-w^2)}{2w}, \frac{\beta_2(u^2-w^2)}{2w}\right]$, ..., $\left(\frac{\beta_{N-1}(u^2-w^2)}{2w}, \frac{\beta_N(u^2-w^2)}{2w}\right]$, $\left(\frac{\beta_N(u^2-w^2)}{2w}, \infty\right)$. According to Equation (17), if y_j^l is in the $m+1$th interval, i.e.,

$$y_j^l \in \left(\frac{\beta_m\left(u^2-w^2\right)}{2w}, \frac{\beta_{m+1}\left(u^2-w^2\right)}{2w}\right] \tag{22}$$

then the first m workers get the full payment and the last $N-m$ workers get partial payment. Also, for the first m workers, Equation (16) has to be satisfied

$$z^l y_j^l > \max_{i=1,\ldots,m} \beta_i u. \tag{23}$$

For the last $N-m$ workers, Equation (15) has to be satisfied

$$y_j^l < \min_{i=m+1,\ldots,N} \left\{\frac{\beta_i\left(u+w\right)\left(u^2-w^2\right)}{u^2}\right\} \tag{24}$$

We then reformulate each sub_j by adding the constraints of Equation (14) (to avoid 0 payment) and Equation (22)-(24) (to ensure m workers get full payment):

$$\max \quad \sum_{i=1}^{N} g\left(y_j; \beta_i\right) + \Delta h\left(y; z^{(l)}\right)$$

$$\text{s.t.} \quad \text{Constraints in Equation (14)-(16), (22)-(24)}$$

which can be solved using subgradient [18].

Lower bound analysis. For theoretical interests, we also derive a lower bound $f_{\text{var}}^{\text{lower}}$ of the workers' payment variance (defined in Equation (11)) to check how close our approximation solution can achieve the optimal. In what follows, we let z^* denote the optimal z provided by the platform and let y_j^* denote the corresponding optimal approval coefficient from requester j ($i = 1, ..., N$).

The basic idea to derive the lower bound is to partition z's feasible region $[0, 1]$ into L intervals $[0, z^1)$, $[z^1, z^2)$, ..., $[z^{L-1}, z^L]$, and then check the lower bound of the payment variance given each $z \in [z^{l-1}, z^l)$ ($l = 1, ..., L$), denoted by $f_{\text{var},l}^{\text{lower}}$. After that, $f_{\text{var}}^{\text{lower}}$ can be obtained by picking up the minimum value from $f_{\text{var},1}^{\text{lower}}, ..., f_{\text{var},L}^{\text{lower}}$: $f_{\text{var}}^{\text{lower}} = \min\left\{f_{\text{var},1}^{\text{lower}}, ..., f_{\text{var},L}^{\text{lower}}\right\}$.

To derive each $f_{\text{var},l}^{\text{lower}}$, given $z \in [z^{l-1}, z^l)$, we calculate the maximum and the minimum value of each y_j such that the constraint in Theorem 4.1 is satisfied, denoted by $y_j^{l,\max}$ and $y_j^{l,\min}$, which can

be obtained by solving the following two optimization problems:

$$\max \arg \max_{y \in \Omega_j(y)} \left\{ \sum_{i=1}^{N} g(y;\beta_i) + h(y;z) \right\} \text{ (to derive } y_j^{l,\max}) \quad (25)$$

$$\min \arg \max_{y \in \Omega_j(y)} \left\{ \sum_{i=1}^{N} g(y;\beta_i) + h(y;z) \right\} \text{ (to derive } y_j^{l,\min}) \quad (26)$$

both of which have the constraint $z \in [z^{l-1}, z^l)$.

When $z \in \left[z^{l-1}, z^l\right)$, given the maximum and the minimum values of y_j, we can then derive the lower bound $f_{\text{var},l}^{\text{lower}}$ by relaxing PFM to the following quadratic programming problem:

$$\min \sum_{j=1}^{M} \sum_{i=1}^{N} \left[f(x_{i,j};y_j) - \overline{f}_j \right]^2 \text{ s.t. } y_j \in \left[y_j^{l,\max}, y_j^{l,\min} \right) \ \forall j.$$

Finally, suppose that z^* is located in the interval $\left[z^{l^*-1}, z^{l^*}\right)$, and hence $y_j^* \in [y_j^{l^*,\min}, y_j^{l^*,\max})$. We can demonstrate that $f_{\text{var}}^{\text{lower}}$ offers a lower bound of workers' payment variance since
1) the payment variance offered by y_i^* is lower bounded by $f_{\text{var},l^*}^{\text{lower}}$
2) $f_{\text{var},l^*}^{\text{lower}}$ is lower bounded by $f_{\text{var}}^{\text{lower}}$.

Figure 6 compares the payment fairness achieved by our solution and the theoretical lower bound, where we change the number of intervals partitioned in z's region from 30 to 42. Here we set both β and λ by 1. From the figure, we find that the payment variance got from our method get closer to the lower bound as the number of intervals in z's region increases. Note that the optimal utility must be within the gap between our calculated utility and its upper bound. Hence, we can conclude the payment variance converges to the optimal as the partition in z's region goes denser.

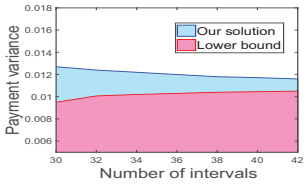


Figure 6: Comparison with the lower bound.

5 PERFORMANCE EVALUATION

We carry out an extensive evaluation of the key parameters of our proposed model using a large dataset (over 3 million records of tasks and over 2,500 workers) extracted from Mechanical Turk collected from September 2014 to January 2017 [17], using the Crowd Workers Chrome plugin [3]. This data includes HIT and HIT group identifiers, unique worker identifiers, and partial records (29.6% of HITs) of when a worker progresses through the various stages of completing a HIT (e.g. *Accept, Submit, Abandon*), and a partial record of subsequent requester actions (e.g. *Accept, Reject*). In our experiments, we remove any workers that did not complete any tasks as well as any requesters that did not approve any tasks, as we consider that these data points represent people that did not actually participate to the crowdsourcing tasks.

We mainly test the following two metrics: (a) *Cost*, defined as the total compensation a requester pay to all his/her workers, i.e., $\sum_i f(x_{i,j};y,z)$; (b) *Fairness*, defined as the *payment variance* of workers, i.e., $\sum_i \left(f(x_{i,j};y) - \overline{f} \right)^2$.

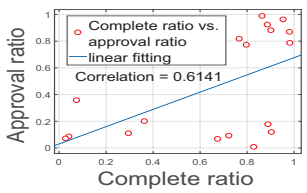


Figure 7: Corr of approval ratio and completion ratio.

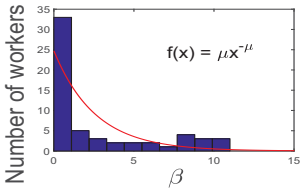


Figure 8: Workers' β, i.e. workers' laziness coefficient.

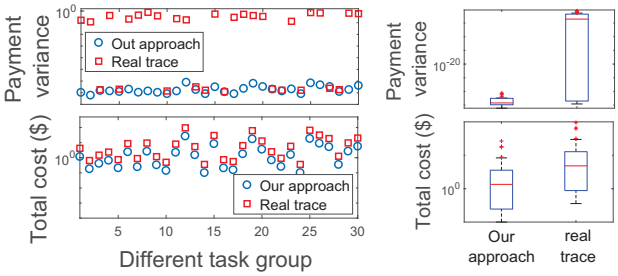
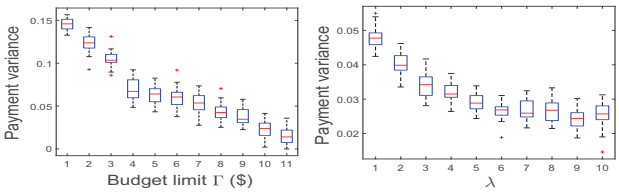


Figure 9: Comparison of workers' payment fairness between the real trace and our method.

In addition, we set $\lambda = 1$ (weight of rating in requesters' utility), $\epsilon = 0.02$ (convergence threshold), and $\Gamma = \$8$ (budget limit) by default.

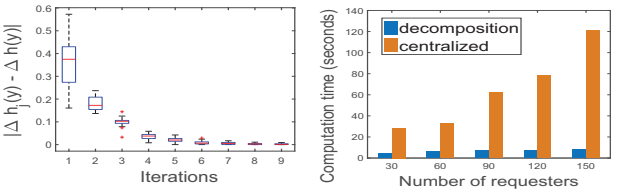
For our first experiments, we select a representative sample of 19 workers (each worker has at least 10 submitted HITs, with 12,188 submitted HITs from 19 workers in total) and calculate each worker's *completion ratio* (i.e., the total number of tasks submitted over the total number of tasks accepted by this worker) and *average approval ratio* (i.e., the total number of tasks approved by all the requesters over the total number of tasks submitted by the worker). Figure 7 shows the correlation between the two values of the 19 workers, which validates our hypothesis that the approval ratio from requesters increases with the increase of workers' completion ratio. As we assume a linear relationship between workers' completion ratio and requesters' approval ratio (Equation (3)), by linear regression, we obtain the estimated approval coefficient $\hat{\lambda} = 0.6141$. Accordingly, we take $\hat{\lambda} = 0.6141$ as the approximated average approval coefficient in the market.

Estimation of β_i. We estimate the *laziness coefficient* β_i (defined in Equation (6)) of each worker i, which reflects the willingness of worker i to put effort in submitting correct tasks, i.e., higher β_i indicates less effort to be taken by the worker. Because of the heterogeneous nature of the data - and in particular of the tasks in the dataset, we identify 100 different groups of tasks according to the *qualifications required* by the tasks (e.g., "{Categorization Masters}", "{Experiences with testing}"). For each task group k ($k = 1, \ldots, 100$), we assume the full payment $u^{(k)}$ as the maximum pay to single HIT in the group and calculate the completion of each worker i, $\hat{x}_i^{(k)}$. Then, by embedding each $\hat{u}^{(k)}$, $\hat{x}_i^{(k)}$, $\hat{y} = \hat{\lambda}$ ($k = 1, \ldots, 100$) into Equation (17), we can obtain the estimated $\hat{\beta}_i$ by linear regression. Figure 8 shows the histogram of workers' β values, which shows that around 55.2% workers have their β values lower than 1. In addition, we use an exponential function $f(x) = \mu x^{-\mu}$ to fit the histogram and $\mu = 2.54$, and we will use this distribution to generate data for synthetic simulation later.



(a) Different budget limit (b) Different λ

Figure 10: Payment variance with different Γ and λ.



(a) Convergence (b) Computation time

Figure 11: Algorithm convergence and computation time.

Our approach vs. trace. Figure 9 compares the fairness and the total cost of our rating policy (per our operational notion of items (a) and (b) at the beginning of this Section 5) with the workers' payment using data from our AMT dataset from 30 different task groups. We first calculate requesters' approval ratio of each group k ($k = 1, ..., 30$) according to their workers' estimated $\hat{\beta}$'s distribution and overall cost constraint $C^{(k)}$. To ensure that the fairness among workers' payment can be improved without increasing the total cost, we let $C^{(k)}$ be equal to the total payment for all the workers in group k. Hence, under our strategy, the overall cost will not be higher than the original cost extracted from the real trace for the same group of tasks. After deriving requesters' approval ratio in each group, we use the pricing policy defined in Equation (4) to derive each worker's payment based on their completion ratio. From the figure, we observe that our pricing policy achieves higher fairness even with slightly lower total cost compared with the real trace.

We next evaluate the performance of our rating policy with different weight λ and different budget limit Γ in Figure 10(a)(b). In Figure 10(a), we increase the budget limit from \$1 to \$11 and test the change of the variance of workers' payment in 30 task groups. The experimental result demonstrates that the higher Γ leads to a lower payment variance (or higher distributive fairness). When Γ is higher, the requester can improve the payment fairness among workers by decreasing the workers' overall approval ratio, by which workers with lower performance will have closer payment to those who are awarded with higher payment. Of course however, higher approval ratio generates higher total payment. In Figure 10(b), we depict how the payment variance evolves with the increase of λ (from 1 to 10), which is defined as the weight assigned to rating in requester's utility. Not surprisingly, the payment variance increases with the increase of λ, as higher λ indicates more willingness from requester to balance workers' payment.

Finally, we test the convergence of our decomposition algorithm as well as its computational time in the 30 task groups. More precisely, we depict the difference between $h(\mathbf{y})^{(k-1)}$ and $h(\mathbf{y})^{(k)}$ derived by the master problem in each iteration k, and check how such difference evolves from iteration 1 to iteration 9 in Figure 11(a). As we mentioned in Section 4.2, we consider the decomposition algorithm has converged to a certain value when $|h(\mathbf{y})^{(k-1)} - h(\mathbf{y})^{(k)}| < \epsilon$ ($\epsilon = 0.02$). From the figure, we find that $|h(\mathbf{y})^{(k-1)} - h(\mathbf{y})^{(k)}|$ is smaller than ϵ after the 5th iteration, demonstrating a fast convergence of the algorithm. Figure 11(b) compares the computation time of our solution with a centralized algorithm (subgradient) when the number of requesters varies from 30 to 150. As shown in the figure, we find that our algorithm has much lower computation time compared with the centralized method. Furthermore, with the increased of the number of requesters, the computation time of our solution is maintained within a certain level, i.e., less than 15 seconds, while the computation time of the centralized increases significantly, i.e., from 27 to 121 seconds.

6 CONCLUSION AND FUTURE WORK

In this work we have presented our fair pricing model, which takes into account the performance and behaviors of both requesters and workers. The model is based on the concept of *distributive fairness*, where we try to reduce the variance between the outcomes for workers completing the same set of tasks. We consider performance of workers, acceptance rate of requester and overall current trend of the crowdsourcing platform where the compensation is calculated and applied. We apply a game theoretical model to describe the interactions between the platform, requesters, and workers and formulate the problem of maximizing distributive fairness among workers as a 3-level programming problem. Considering the hardness of the problem, we propose a time-efficient approach by resorting to problem approximation and primal decomposition.

Our model shows promise in several regards. First, the results of applying it to our data sample make intuitive sense. That is, if the budget for a set of tasks increases, while the amount and quality of work that a requester needs stays the same, then the compensation policy creates a *more* fair distribution of pay. If the budget for a set of tasks stays the same, while the quality of work that a requester needs increases, then our model highlights that this creates a *less* fair distribution of pay. While this is intuitively true, our model can *recommend* to requesters how much to increase the overall budget if the requester needs to increase the quality needed. Our rating policy motivates requesters to compensate their workers in a way such the disparity among workers' payments is minimized and also each worker's compensation reasonably reflects his/her quality of work. This effect can provide incentives to the requesters to maintain a fair environment in which crowd workers do their work, while adjusting aspects of how their work needs to be completed according to their needs.

We see a number of promising directions for this research work. For example, currently all tasks are considered of the same importance for the requesters who asked for completion. This is often not the case, as requesters may have urgent tasks where rate of completion is one of the variables affecting price. Accordingly, extensions to the model could be applied to better reflect the urgency with which a requester might need a task completed and the subsequent demand (and price) that this puts on workers.

7 ACKNOWLEDGEMENTS

This work was supported in part by AFOSR grant FA9550-15-1-0149 and ICS Seed Grant. We thank Dr. Kotaro Hara and Dr. Chris Callison-Burch for providing us with the data.

REFERENCES

[1] Vamshi Ambati, Stephan Vogel, and Jaime Carbonell. 2011. Towards Task Recommendation in Micro-Task Markets.. In *Proc. of AAAI Conference on Human Computation*.

[2] J. F. Bard. 1991. Some properties of the bilevel programming problem. *Journal of Optimization Theory and Applications* 68, 2 (01 Feb 1991), 371–378. https://doi.org/10.1007/BF00941574

[3] Chris Callison-Burch. 2014. Crowd-Workers: Aggregating Information Across Turkers To Help Them Find Higher Paying Work. In *HCOMP-2014*.

[4] C. Cao, J. She, Y. Tong, and L. Chen. 2012. Whom to ask? jury selection for decision making tasks on micro-blog services.. In *Proc. of VLDB*.

[5] Justin Cheng, Jaime Teevan, and Michael S. Bernstein. 2015. Measuring Crowdsourcing Effort with Error-Time Curves. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 1365–1374.

[6] Yochi Cohen-Charash and Paul E Spector. 2001. The role of justice in organizations: A meta-analysis. *Organizational behavior and human decision processes* 86, 2 (2001), 278–321.

[7] JA. Colquitt, DE. Conlon, MJ. Wesson, and CO. Porter. 2001. Justice at the millennium: A metaanalytic review of 25 years of organizational justice research. *Journal of Applied Psychology* (2001).

[8] JA. Colquitt, BA. Scott, JB. Rodell, and DM. Long. 2013. Justice at the millennium, a decade later: A meta-analytic test of social exchange and affect-based perspectives. *Journal of Applied Psychology* (2013).

[9] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel. 2012. Fairness Through Awareness.. In *Proc. of Innovations in Theoretical Computer Science Conference*.

[10] R. Faullant, J. Füller, and K. Hutter. 2013. Fair play: perceived fairness in crowdsourcing communities and its behavioral consequences.. In *Proc. of Academy of Management Annual Meeting Proceedings*.

[11] N. Franke, P. Keinz, and K. Klausberger. 2012. Does This Sound Like a Fair Deal?" Antecedents and Consequences of Fairness Expectations in the Individual's Decision to Participate in Firm Innovation. *Organization Science* (2012).

[12] N. Franke and K. Klausberger. 2009. The role of perceived fairness in company-centred crowdsourcing communities. *Vienna University of Economics and Business Administration* (2009).

[13] X. Gan, Y. Li, W. Wang, L. Fu, and X. Wang. 2017. Social Crowdsourcing to Friend: An Incentive Mechanism for Multi-Resource Sharing. *IEEE JSAC* (2017).

[14] Yanmin Gong, Yuguang Fang, and Yuanxiong Guo. 2014. Optimal Task Recommendation for Mobile Crowdsourcing With Privacy Control. *IEEE Internet of Things Journal* (2014).

[15] Victor M Gonzalez, Leonardo Galicia, and Jesus Favela. 2008. Understanding and supporting personal activity management by IT service workers. In *Proceedings of the 2nd ACM Symposium on Computer Human Interaction for Management of Information Technology (CHiMiT '08)*. ACM, New York, NY, USA, 2:1–2:10. https://doi.org/10.1145/1477973.1477976

[16] Mary L Gray, Siddharth Suri, Syed Shoaib Ali, and Deepti Kulkarni. 2016. The Crowd is a Collaborative Network. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing (CSCW '16)*. ACM, New York, NY, USA, 134–147. https://doi.org/10.1145/2818048.2819942

[17] Kotaro Hara, Abigail Adams, Kristy Milland, Saiph Savage, Chris Callison-Burch, and Jeffrey P. Bigham. 2018. A Data-Driven Analysis of Workers' Earnings on Amazon Mechanical Turk. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, 449:1–449:14.

[18] Frederick S. Hillier. 2008. *Linear and Nonlinear Programming*. Stanford University.

[19] Zehong Hu and Jie Zhang. 2017. Optimal posted-price mechanism in microtask crowdsourcing.. In *Proc. of AAAI*.

[20] P. Hyman. 2013. Software aims to ensure fairness in crowdsourcing projects. *Commun. ACM* (2013).

[21] Lilly Irani. 2015. Difference and Dependence among Digital Workers: The Case of Amazon Mechanical Turk. *South Atlantic Quarterly* 114, 1 (2015), 225–234.

[22] Lilly C Irani and M Silberman. 2013. Turkopticon: Interrupting worker invisibility in amazon mechanical turk. In *CHI 2013*. ACM, 611–620.

[23] K. James. 1993. The social context of organizational justice: Cultural, intergroup, and structural effects on justice behaviors and perceptions. *In: Cropanzano R (ed.) Justice in the Workplace: Approaching fairness in human resource management.* (1993).

[24] Aniket Kittur, Jeffrey V Nickerson, Michael Bernstein, Elizabeth Gerber, Aaron Shaw, John Zimmerman, Matt Lease, and John Horton. 2013. The future of crowd work. In *CSCW 2013*. ACM, 1301–1318.

[25] A. Kobren, C. H. Tan, P. Ipeirotis, and E. Gabrilovich. 2015. Getting more for less: Optimized crowdsourcing with dynamic tasks and goals.. In *Proc. of WWW*.

[26] H. Li, JB. Bingham, and EE. Umphress. 2007. Fairness from the top: Perceived procedural justice and collaborative problem solving in new product development. *Organization Science* (2007).

[27] Qing Liu, Talel Abdessalem, Huayu Wu, Zihong Yuan, and Stephane Bressan. 2016. Cost minimization and social fairness for spatial crowdsourcing tasks.. In *Proc. of International Conference on Database Systems for Advanced Applications*.

[28] David Martin, Benjamin V. Hanrahan, Jacki O'Neill, and Neha Gupta. 2014. Being A Turker. *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing* (2 2014), 224–235. https://doi.org/10.1145/2531602.2531663

[29] David Martin, Benjamin V Hanrahan, Jacki O'Neill, and Neha Gupta. 2014. Being a turker. In *CSCW 2014*. ACM, 224–235.

[30] David Martin, Jacki O'Neill, Neha Gupta, and Benjamin V. Hanrahan. 2016. Turking in a Global Labour Market. *Computer Supported Cooperative Work: CSCW: An International Journal* 25, 1 (2 2016), 39–77.

[31] Hanan Mengash and Alexander Brodsky. 2015. A group package recommender based on learning group preferences, multi-criteria decision analysis, and voting. *EURO Journal on Decision Processes* (2015).

[32] Hanan Mengash and Alexander Brodsky. 2016. Tailoring Group Package Recommendations to Large Heterogeneous Groups Based on Multi-Criteria Optimization. In *System Sciences (HICSS), 2016 49th Hawaii International Conference on*. IEEE, 1537–1546.

[33] Victor Naroditskiy, Nicholas R Jennings, Pascal Van Hentenryck, and Manuel Cebrian. 2014. Crowdsourcing contest dilemma. *Journal of the Royal Society, Interface / the Royal Society* 11 (10 2014). https://doi.org/10.1098/rsif.2014.0532

[34] D. P. Palomar and Mung Chiang. 2006. A Tutorial on Decomposition Methods for Network Utility Maximization. *IEEE J.Sel. A. Commun.* 24, 8 (Aug. 2006), 1439–1451. https://doi.org/10.1109/JSAC.2006.879350

[35] Chenxi Qiu, Anna Cinzia Squicciarini, Sarah Michele Rajtmajer, and James Caverlee. 2017. Dynamic Contract Design for Heterogenous Workers in Crowdsourcing for Quality Control. In *ICDCS*. IEEE, 1168–1177.

[36] S. M. Ross. 2003. *Introduction to Probability Models, 8th Edition*. Amsterdam: Academic Press.

[37] H. Sauermann and WM. Cohen. 2010. What makes them tick? Employee motives and firm innovation. *Management Science* (2010).

[38] Steffen Schnitzer, Christoph Rensing, Sebastian Schmidt, Kathrin Borchert, Matthias Hirth, and Phuoc Tran-Gia. 2015. Demands on task recommendation in crowdsourcing platforms - the worker's perspective.. In *Proc. of CrowdRec*.

[39] CE. Shalley and LL. Gilson. 2004. What leaders need to know: A review of social and contextual factors that can foster or hinder creativity. *The Leadership Quarterly* (2004).

[40] Yaron Singer and Manas Mittal. 2013. Pricing Mechanisms for Crowdsourcing Markets.. In *Proc. of ACM WWW*.

[41] H. Xie, J. C. S. Lui, and W. Jiang. 2014. Mathematical Modeling of Crowdsourcing Systems: Incentive Mechanism and Rating System Design. In *2014 IEEE 22nd International Symposium on Modelling, Analysis Simulation of Computer and Telecommunication Systems*. 181–186. https://doi.org/10.1109/MASCOTS.2014.31

[42] Chencan Xu, Yawen Li, Weiyu Zhang, Mark Klein, and Shengdong Zhao. 2016. Towards Greater Perceived Fairness: Crowdsourcing Moderation Work to Online Deliberation Participants.. In *Proc. of CHI - Workshop: Crowd Dynamics: Exploring Conflicts and Contradictions in Crowdsourcing*.

[43] Ming Yin and Yiling Chen. 2014. Increased efficiency through pricing in online labor markets. *Journal of Electronic Commerce Research*.

[44] Ming Yin and Yiling Chen. 2015. Bonus or Not? Learn to Reward in Crowdsourcing.. In *Proc. of IJCAI*.

[45] Man-Ching Yuen, Irwin King, and Kwong-Sak Leung. 2011. Task Matching in Crowdsourcing.. In *Proc. of IEEE International Conference on Cyber, Physical and Social Computing*.

[46] Man-Ching Yuen, Irwin King, and Kwong-Sak Leung. 2012. Task Recommendation in Crowdsourcing Systems.. In *Proc. of Workshop on Crowdsourcing and Data Mining*.

[47] Z. Zhao, F. Wei, M. Zhou, W. Chen, and W. Ng. 2015. Crowd-selection query processing in crowdsourcing databases: A task-driven approach.. In *Proc. of EDBT*.