Design and Analysis of Incentive and Reputation Mechanisms for Online Crowdsourcing Systems

HONG XIE and JOHN C. S. LUI, Chinese University of Hong Kong
DON TOWSLEY, University of Massachusetts

Today, online crowdsourcing services like Amazon Mechanical Turk, UpWork, and Yahoo! Answers are gaining in popularity. For such online services, it is important to attract "workers" to provide high-quality solutions to the "tasks" outsourced by "requesters." The challenge is that workers have different skill sets and can provide different amounts of effort. In this article, we design a class of incentive and reputation mechanisms to solicit high-quality solutions from workers. Our incentive mechanism allows multiple workers to solve a task, splits the reward among workers based on requester evaluations of the solution quality, and guarantees that high-skilled workers provide high-quality solutions. However, our incentive mechanism suffers the potential risk that a requester will eventually collects low-quality solutions due to fundamental limitations in task assigning accuracy. Our reputation mechanism ensures that low-skilled workers do not provide low-quality solutions by tracking workers' historical contributions and penalizing those workers having poor reputations. We show that by coupling our reputation mechanism with our incentive mechanism, a requester can collect at least one high-quality solution. We present an optimization framework to select parameters for our reputation mechanism. We show that there is a trade-off between system efficiency (i.e., the number of tasks that can be solved for a given reward) and revenue (i.e., the amount of transaction fees), and we present the optimal trade-off curve between system efficiency and revenue. We demonstrate the applicability and effectiveness of our mechanisms through experiments using a real-world dataset from UpWork. We infer model parameters from this data, use them to determine proper rewards, and select the parameters of our incentive and reputation mechanisms for UpWork. Experimental results show that our incentive and reputation mechanisms achieve 98.82% of the maximum system efficiency while only sacrificing 4% of revenue.

Categories and Subject Descriptors: H.4 [**Information Systems Applications**]: World Wide Web

General Terms: Crowdsourcing, Incentive Schemes, Reputation Systems

Additional Key Words and Phrases: Bayesian game, repeated game, equilibrium

ACM Reference Format:
Hong Xie, John C. S. Lui, and Don Towsley. 2016. Design and analysis of incentive and reputation mechanisms for online crowdsourcing systems. ACM Trans. Model. Perform. Eval. Comput. Syst. 1, 3, Article 13 (May 2016), 27 pages.
DOI: http://dx.doi.org/10.1145/2897510

The work of John C. S. Lui was supported in part by GRF 14205114. An An earlier version of the article appeared in *Proceedings of the 23rd IEEE/ACM International Symposium on Quality of Service (IWQoS'15)* [Xie et al. 2015]. In this journal version, we add an optimization framework to explore tradeoffs in selecting parameters for our reputation mechanism, and add experiments on a dataset from UpWork to show the effectiveness and applicability of our incentive and reputation mechanisms.
Authors' addresses: H. Xie and J. C. S. Lui, Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, NT, Hong Kong SAR, The People's Republic of China; emails: hongx87@gmail.com, cslui@cse.cuhk.edu.hk; D. Towsley, Department of Computer Science, University of Massachusetts Amherst, MA 01003 USA; email: towsley@cs.umass.edu.

DOI: http://dx.doi.org/10.1145/2897510

1. INTRODUCTION

Today, many tasks that are easy for humans, such as image labeling and translation, still challenge the most sophisticated computer. Motivated by a demand of efficient paradigms to solve such tasks, and with the recent advancement of Internet technologies, online crowdsourcing arose [Howe 2006]. Coined by Jeff Howe and Mark Robinson in 2005, crowdsourcing has emerged as an efficient and cost-effective paradigm to obtain needed services, ideas, or content [Howe 2006, 2008; Yuen et al. 2011]. Many online crowdsourcing systems have emerged over the past decade (e.g., Amazon Mechanical Turk [2008], UpWork [Elance 1999], Yahoo! Answers [2012]). A variety of "tasks" can be outsourced to an online crowdsourcing system (e.g., image labeling [Ipeirotis 2010], question answering [Adamic et al. 2008; Yahoo! Answers 2012], product design [Threadless 2000], and human behavioral data collection [Mason and Suri 2012; Paolacci et al. 2010]). Crowdsourcing has been used as a data-gathering paradigm as well. Some well-known examples include Wikipedia and YouTube. In general, an online crowdsourcing system creates an online labor market to solve tasks by soliciting contributions from a large group of users online.

A typical online crowdsourcing system classifies users into two types: requesters and workers. Requesters outsource a large number of tasks to workers, who in turn solve the tasks and reply to requesters with solutions. To encourage workers to participate, requesters usually set appropriate rewards for tasks, and the rewards are granted to workers who contribute to solving the task. A variety of rewards have been deployed in different online crowdsourcing systems (e.g., monetary rewards [Amazon Mechanical Turk 2008; Taskcn 2010], nonmonetary rewards [Von Ahn 2006], or simply altruism [Yahoo! Answers 2012]). Workers, on the other hand, have different skill sets, ranging from low skills to high skills. High-skilled workers are capable of providing high-quality solutions, whereas solutions by low-skilled workers may have little value to requesters. Furthermore, it is common that requesters interact with workers, for which they have little prior knowledge of the workers' skills.

In general, online crowdsourcing systems face two fundamental challenges: (1) how to consistently solicit active participation of users (requesters and workers) and (2) how to solicit high-quality solutions from workers [Horton 2010; Mason and Watts 2010]. This is because an online crowdsourcing system can collapse either due to the lack of user participation or to the low quality of the solutions. This article focuses on the design of incentive and reputation mechanisms to solicit active participation and to generate high-quality solutions by the workers.

Designing incentive and reputation mechanisms for crowdsourcing systems faces two challenges. The first is how to incentivize workers to provide their maximum effort. This involves designing an effective incentive mechanism to determine the appropriate reward. If the reward is small, workers may not participate or may just exert a small amount of effort, leading to a task not being solved or a low-quality solution. Requesters, on the other hand, always want to minimize reward payments. Note that simply incentivizing workers to provide their maximum effort is not enough, as workers may not have sufficient skills. The second challenge is how to guarantee that sufficiently skilled workers can be recruited and at the same time provide their maximum effort. Traditional approaches to address this challenge are based on adaptive task assignment algorithms [Ho and Vaughan 2012]. However, such algorithms only improve the probability that a task is assigned to a high-skilled workers. The complex nature of worker skills and tasks challenges the assigning accuracy. We pose a new angle to address this challenge by deploying a reputation mechanism that provides extra incentives for workers to perform self-selection such that at least one sufficiently skilled worker is guaranteed to participate. Specifically, we reduce our problem to how

to define and update workers' reputations based on their historical contributions so that when coupled with appropriate rewards, requesters are guaranteed to collect at least one high-quality solution for each of their tasks. To the best of our knowledge, this is the first work combining incentive and reputation mechanisms to guarantee high-quality solutions. Our contributions are as follows:

—We design a mechanism to incentivize workers to provide their maximum effort. Our incentive mechanism allows multiple workers to work on a task and split the reward among workers based on requester evaluations of the solution quality. We use the Bayesian game framework to derive the minimum reward needed to ensure that workers provide their maximum effort.
—Our incentive mechanism suffers the potential risk that a requester will collect low-quality solutions due to fundamental limitations in task assigning accuracy. We design a class of reputation mechanisms and show that coupling this reputation mechanism with our incentive mechanism ensures at least one high-quality solution per task. We also show that our incentive and reputation mechanisms are robust against human factors (i.e., preferences or biases) on the part of requesters in evaluating solution quality.
—We present an optimization framework to determine parameters for our reputation mechanism. We show that there is a trade-off between system efficiency (i.e., the number of tasks that can be solved for a given reward) and revenue (i.e., the amount of transaction fees). We identify extremal design points that correspond to the maximum efficiency or the maximum revenue, as well as the optimal trade-off curve of efficiency and revenue.
—We demonstrate the applicability and effectiveness of our mechanisms through experiments using a dataset from UpWork. From the data, we infer model parameters and use them to determine the reward payment and select parameters for our incentive and reputation mechanisms for UpWork. Our incentive and reputation mechanisms are shown to achieve 98.82% of the maximum system efficiency while only sacrificing 4% revenue.

This article is organized as follows. In Section 2, we present the crowdsourcing system model. Sections 3 and 4 present the design and analysis of our incentive and reputation mechanisms. In Section 5, we demonstrate the robustness of our incentive and reputation mechanisms against human factors. Section 6 presents the design trade-offs in selecting the parameters for our reputation mechanism. In Section 7, we present experimental results on a real-world dataset. Related work is given in Section 8, and we conclude in Section 9.

2. CROWDSOURCING SYSTEM MODEL

Crowdsourcing systems usually include three components: tasks, workers, and requesters. Requesters outsource tasks to workers to perform and provide a reward for each task (the appropriate reward will be discussed later). The reward is distributed to workers who made appropriate contributions to solving the task.

Tasks are classified into different types. For example, UpWork classifies these into tasks such as "Mobile," "Data Science," and "Translation." [Elance 1999]. Our model considers $L \geq 1$ types of tasks. For a type $\ell \in \{1, \ldots, L\}$ task, a requester sets a nonnegative reward of r_ℓ to be split among contributing workers, and the requester also pays a transaction fee of $T_\ell = \tau_\ell r$, where $\tau_\ell \in [0, 1]$, to the crowdsourcing system. Hence, the total payment by the requester is $r_\ell + T_\ell$. Without loss of generality, we focus on one type of task in our analysis. Later we will see that our analysis applies to all task types. Thus, for ease of presentation, we drop the subscript ℓ.

We consider a population of heterogeneous workers (i.e., workers have different skill sets). More precisely, each worker is associated with a skill level $m \in \mathcal{M} \triangleq \{1, \ldots, M\}$. A higher value of m implies that the worker has a higher skill set. We emphasize that this categorization is task dependent (i.e., a worker may have skill level M for one task and skill level 1 for another task). We assume that skill levels are only known to workers.

2.1. Model for Task Assignment

We use a probabilistic model to characterize the task assignment process of a crowdsourcing system. Some real-world crowdsourcing systems (e.g., Amazon Mechanical Turk [2008]) allow workers to select tasks, whereas others like Clickworker [2001] require the system administrator to assign tasks to workers. We use a general probabilistic model to represent all possible cases of the task assignment process: with probability $\beta_m \in [0, 1]$, a task is assigned (or recommended) to a worker having skill level m, where $\sum_{m=1}^{M} \beta_m = 1$. Tasks are independently assigned to different workers. The task assignment process is represented by an M-dimensional vector $(\beta_1, \ldots, \beta_M)$. In practice, an optimal and deterministic task assignment strategy is usually impossible to compute, as tasks are heterogeneous and it is difficult to know the exact skill level of a worker. Furthermore, the deterministic task assignment strategy is a special case of the probabilistic task assignment strategy. For example, $(\beta_1, \ldots, \beta_M) = (0, \ldots, 0, 1)$ represents the deterministic task assignment strategy that tasks are always assigned to workers having skill level M. We assume that the values of $\beta_1, \ldots, \beta_M$ are known to all workers. This assumption is realistic. On the one hand, the system operator can monitor the task assigning accuracy and make it public to all workers. On the other hand, a worker can perceive the task assigning accuracy from his own historical task assignments.

2.2. Model for Worker Action

Our model allows $n \geq 1$ workers to solve a task denoted by $w_1, \ldots, w_n$. Let $\theta_i \in \{1, \ldots, M\}$ denote the skill level of w_i. When a task is assigned to a worker, this worker can either refuse to take on this task or exert some effort to solve this task. When a worker refuses a task, the task is then reassigned to another worker until exactly n workers commit to solving this task. Without loss of generality, we denote the probability that a worker having skill type m refuses a task by $\eta_m \in [0, 1]$. Using Bayes' rule, one can easily express the probability mass function of θ_i as

$$\Pr[\theta_i = m'] = \frac{(1 - \eta_{m'})\beta_{m'}}{\sum_{j=1}^{M}(1 - \eta_j)\beta_j}, \quad \forall m' \in \mathcal{M}. \tag{1}$$

We emphasize that $\theta_1, \ldots, \theta_n$ are independent and identically distributed random variables. We assume that $\eta_1, \ldots, \eta_M$ are known to all workers. This assumption is reasonable, because as we shall see later, our incentive and reputation mechanisms control $\eta_1, \ldots, \eta_M$, through which $\eta_1, \ldots, \eta_M$ are made public to all workers.

Worker w_i exerts effort $a_i \in \mathcal{K} \triangleq \{0, 1, \ldots, K\}$ to complete a task. The larger the a_i, the greater the effort. Moreover, there is a cost associated with each effort level—for instance, $c_{m,k}$ is the cost associated with effort level $k \in \mathcal{K}$ and skill set type $m \in \mathcal{M}$, where $c_{m,K} > \cdots > c_{m,0} = 0$, capturing the physical meaning that the more effort a worker provides, the larger the cost incurred. Without loss of generality, we assume a strictly greater order to simplify the analysis. Here, $c_{m,0} = 0$ corresponds to a type m worker choosing to be a free rider by exerting no effort. Furthermore, the cost of exerting the same effort level decreases as a worker is more skilled in solving a given task—for instance, $c_{M,k} \leq \cdots \leq c_{1,k}$, where $k \in \mathcal{K}$.

2.3. Model for Solution Quality and Benefit

The quality of a solution is jointly determined by a worker's skill level and effort level. Specifically, the quality of a solution increases as the skill level increases or as greater effort is exerted. Let $Q(m, k) \in [0, 1]$, $(m \in \mathcal{M}, k \in \mathcal{K})$, denote the quality of a solution submitted by a worker with skill level m exerting a level k effort. A larger value of $Q(m, k)$ means that the solution has higher quality. Our model assumes that $Q(m, k)$ exhibits three properties. First, a higher effort level implies higher quality: $Q(m, k) > Q(m, k'), \forall k > k'$. Second, a higher worker skill level implies higher quality: $Q(m, k) > Q(m', k), \forall m > m', k > 0$. Thirs, the quality of a zero effort solution is zero: $Q(M, 0) = \cdots = Q(1, 0) = 0$. Without loss of generality, we assume a strictly greater order to simplify the analysis.

The benefit of a solution to a requester is determined by its quality. A solution submitted by a worker having skill level $m \in \mathcal{M}$ working at effort level $k \in \mathcal{K}$ contributes a benefit of $V_{m,k} \geq 0$ to a requester. Our model assumes that $V_{m,k}$ exhibits the following two properties. First, the higher the quality of a solution, the greater benefit to a requester: $V_{m,k} > V_{m',k'}$ if and only if $Q(m, k) > Q(m', k')$. Second, two solutions bring the same benefit if and only if they have the same quality: $V_{m,k} = V_{m',k'}$ if and only if $Q(m, k) = Q(m', k')$.

A requester collects n solutions and selects the one with the highest quality. If a tie occurs (i.e., multiple solutions of the highest quality), each solution in this tie is equally likely to be selected. The overall benefit of these n solutions corresponds to the *largest* single solution benefit (i.e., $\max\{V_{\theta_1,a_1}, \ldots, V_{\theta_n,a_n}\}$). Note that we have to ensure $V_{M,K} > r + T$ to attract requesters to participate. Namely, there is at least one worker with a skill set such that if he exerts his highest effort, he will contribute a benefit that outweighs a requester's cost. For ease of presentation, we focus on the scenario where requesters have high expectation regarding solutions: they will only be satisfied if at least one solution has the highest possible quality $Q(M, K)$—that is, $V_{M,k} < r + T$ for all $k < K$, and $V_{m,K} < r + T, \forall m < M$. Our results can be easily extended to other selections of the minimum solution quality that satisfies a requester.

2.4. Discussion

Discretizing workers' skill set and effort level is practical for modeling real-world crowdsourcing systems. First, it simplifies the analysis and makes it much easier to present the main ideas and results than using a continuous model. Furthermore, as we will show later, although our incentive mechanism applies when workers have continuous skills levels, the analysis will be more complicated. Finally, the main challenge lies in reputation system design if we consider a continuous model. It will be complicated to set the quality threshold and design a feedback rating system as well.

3. INCENTIVE MECHANISM

We propose a class of mechanisms to incentivize high-quality solutions. They allow multiple workers to solve a task and split the reward based on requester evaluations of the quality of the solution. We use the Bayesian game framework to derive the *minimum* reward needed to guarantee that workers exert their greatest effort. We also identify the minimum number of workers needed so that at least one worker with skill level M participates. We derive the reward needed to guarantee that this worker provides his best effort.

3.1. Incentive Mechanism Design

Our objective is to determine the minimum reward needed to attract both requesters and workers to participate, and to incentivize workers to exert their maximum effort

(level K). This is challenging because large rewards will attract workers but may discourage requesters from participating. Furthermore, a social dilemma arises as to when to settle the payment. If a requester pays before a task begins, workers may have an incentive to free ride (i.e., take the reward while exerting no effort). However, if a requester pays after receiving solutions, he may refuse to pay the workers, thus discouraging workers from participating. Finally, the incentive mechanism design is complicated by the heterogeneity of workers with different skill sets and their goal to maximize their payoffs by strategically exerting one of $K + 1$ levels of effort.

We consider the following incentive mechanism. Upon posting a task, a requester must submit its associated reward r and transaction fee T to the crowdsourcing system administrator. The task is solved by exactly n workers. Once a worker solves a task, he submits the solution to the system administrator. After collecting all n solutions, the administrator forwards them to the requester. Upon receiving these n solutions, a requester evaluates their quality and selects the one with the highest quality. If there is a tie, each solution within the tie is selected with equal probability. We refer to the highest-quality solutions as winner(s) and the other solutions as losers. Finally, the requester notifies the system administrator as to the identity of the winners, and the system administrator distributes the reward r evenly to those winners. Workers are encouraged to participate because winners equally share the reward. One important property of this scheme is that the requester cannot refuse to pay workers, as the reward r and the transaction fee T are held by the administrator. Hence, he does not benefit from providing false feedback—for example, notifying the administrator that there is no winner, as the reward is not returned to the requester.

Discussion. Our model applies to *macrotasking* crowdsourcing systems [Wikipedia 2003a], which serve as platforms to outsource innovative and challenging tasks that require special skills (e.g., develop a computer program). Typical examples of such systems include UpWork [Elance 1999] and Fiverr [2010], which are two real-world macrotasking crowdsourcing systems. It is widely practiced in such systems that requesters evaluate the solution quality and reward the best answer, as the cost of finding the best answer is small as compared to solving the task. The other type of crowdsourcing is *microtasking* [Wikipedia 2003b] crowdsourcing systems (e.g., Amazon Mechanical Turk [2008] and Microtask [2003]), where the assumptions of our model may not hold. The reason is that for such tasks, the cost incurred by finding the best answer will be almost as high as solving the task since the tasks are small and repetitive (e.g., image labeling, transcription).

We like to note that our incentive mechanism is rational in expectation—for instance, low-skilled workers may have negative utility even if they provide their maximum effort. This is practical because the solutions submitted by low-skilled workers are of low quality, and in real-world crowdsourcing systems (e.g., UpWork), a worker may not get any reward if he submits a low-quality solution.

The preceding mechanism allows a worker to receive a reward without exerting any effort. For example, when all n workers exert no effort, each of them receives a reward of r/n. We next use the Bayesian game framework to derive the minimum reward needed so that workers exert their maximum (level K) efforts.

3.2. Formulating the Bayesian Game

In our Bayesian game formulation, n workers participate in a task $w_1, \ldots, w_n$. Each worker has the same set of actions $\mathcal{K}$ and can be of any type in $\mathcal{M}$. Recall that $a_i \in \mathcal{K}$ denotes the action of worker w_i and $\theta_i \in \mathcal{M}$ denotes the skill type of w_i. We define $\mathbf{a} = (a_1, \ldots, a_n)$ and $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)$. We use $u_i(\mathbf{a}, \boldsymbol{\theta})$ to denote the payoff function for

worker w_i,

$$u_i(\mathbf{a}, \boldsymbol{\theta}) = \begin{cases} \frac{r}{\sum_{j=1}^{n} \mathbf{I}_{(Q(\theta_j, a_j) = Q(\theta_i, a_i))}} - c_{\theta_i, a_i}, & \text{if } Q(\theta_i, a_i) = \max_j Q(\theta_j, a_j), \\ -c_{\theta_i, a_i}, & \text{if } Q(\theta_i, a_i) < \max_j Q(\theta_j, a_j). \end{cases}$$

Recall that θ_i is private information known only to w_i. Only the joint distribution $\Pr[\boldsymbol{\theta}]$ over the types of workers is known to all workers. Hence, we can express worker w_i's belief on the types of other workers as $\Pr[\theta_{-i}|\theta_i]$, where $\theta_{-i} = [\theta_j]_{j \neq i}$ denotes a vector of types for all other workers except w_i. Since $\theta_1, \ldots, \theta_n$ are independent and identically distributed random variables, with the probability mass function given in Equation (1), we have

$$\Pr[\theta_{-i}|\theta_i] = \Pr[\theta_{-i}] = \prod_{j=1, j \neq i}^{n} \frac{(1 - \eta_{\theta_j})\beta_{\theta_j}}{\sum_{m=1}^{M}(1 - \eta_m)\beta_m}.$$

We now describe the action space of each worker. A worker may randomize his potential actions (i.e., exert at level $k \in \mathcal{K}$ with some probability). We refer to such actions as mixed actions. Formally, we represent each *mixed action* by a $K+1$–dimensional vector $\sigma = (p(0), p(1), \ldots, p(K))$, where $p(k)$ is the probability that a worker exerts level k effort, with $p(k) \geq 0$, $\sum_{k=0}^{K} p(k) = 1$. Define $\boldsymbol{\Sigma}$ to be the action space for a worker, which is the set of all possible mixed actions:

$$\boldsymbol{\Sigma} = \left\{ (p(0), p(1), \ldots, p(K)) | p(k) \geq 0, \sum_{k=1}^{K} p(k) = 1 \right\}.$$

For simplicity, we refer to a mixed action as an action.

We now introduce the concept of a strategy, which prescribes an action for each possible worker type (i.e., skill level).

Definition 3.1. Worker w_i's strategy is a map $\mathbf{s}_i : \mathcal{M} \to \boldsymbol{\Sigma}$ prescribing an action for each possible skill type.

For example, $\mathbf{s}_i(M) = (0, \ldots, 0, 1)$ means that when w_i has skill type $\theta_i = M$, he always exerts level K effort. On the other hand, $\mathbf{s}_i(1) = (\frac{1}{K+1}, \ldots, \frac{1}{K+1})$ means that when w_i has skill type $\theta_i = 1$, he exerts any effort level with equal probability of $1/(K+1)$.

We now introduce expected utility. Each worker knows his own type and tries to maximize his expected utility. Let $\mathbf{s}_{-i}(\cdot) = [\mathbf{s}_j(\cdot)]_{j \neq i}$ denote the vector of strategies for all other workers except w_i. Let $u_i(\mathbf{s}_i(\theta_i), \mathbf{s}_{-i}(\cdot), \theta_i)$ denote the expected utility for w_i given that he has skill type θ_i, under strategy profile $(\mathbf{s}_i(\theta_i), \mathbf{s}_{-i}(\cdot))$. We have

$$u_i(\mathbf{s}_i(\theta_i), \mathbf{s}_{-i}(\cdot), \theta_i) = \sum_{\theta_{-i}} \prod_{\ell=1, \ell \neq i}^{n} \frac{(1 - \eta_{\theta_\ell})\beta_{\theta_\ell}}{\sum_{m=1}^{M}(1 - \eta_m)\beta_m} \sum_{\mathbf{a} \in \mathcal{K}^n} \prod_{j=1}^{n} [\mathbf{s}_j(\theta_j)]_{a_j} u_i(\mathbf{a}, \boldsymbol{\theta}),$$

where $[\mathbf{s}_j(\theta_j)]_{a_j}$ represents the probability that w_j plays $a_j \in \mathcal{K}$ under action $\mathbf{s}_j(\theta_j)$. We now introduce the Bayesian Nash equilibrium in which each worker's strategy must be the best response to the other workers' strategies.

Definition 3.2. $(\mathbf{s}_1^*(\cdot), \ldots, \mathbf{s}_n^*(\cdot))$ is a Bayesian Nash equilibrium if for all $i = 1, \ldots, n$, and for all $\theta_i \in \mathcal{M}$, we have $\mathbf{s}_i^*(\theta_i) \in \arg\max_{\mathbf{s}_i(\theta_i) \in \boldsymbol{\Sigma}} u_i(\mathbf{s}_i(\theta_i), \mathbf{s}_{-i}^*(\cdot), \theta_i)$.

We next show that our incentive mechanism guarantees that each type of worker exerts the maximum (level K) effort, once we properly set the reward.

3.3. Deriving the Minimum Reward

We first show that when $n = 1$ (or single worker), it is impossible to incentivize the worker to exert an effort higher than level 0. For $n \geq 2$, we derive the minimum reward required to ensure that all workers exert their maximum (level K) efforts. We also derive the minimum number of workers needed such that at least one worker with the highest skill level, M, participates with high probability. We then derive the corresponding minimum reward to ensure that this worker exerts his maximum (level K) effort and show that it decreases with respect to β_M.

We first explore the simplest case of $n = 1$, where a task is assigned to only one worker. In this case, a worker plays a single game, where reward r is always distributed to that worker no matter what effort he exerts. Hence, a worker does not refuse a task, because the utility for refusing a task is zero. Furthermore, the utility of a worker is maximized when he exerts no effort (level 0). This implies that the worker should free ride and never exert any effort. We next show that this undesirable result *can be eliminated* when we allow more than one worker to solve a task (i.e., $n \geq 2$).

We now consider the case $n = 2$. We derive the minimum reward needed so that workers are guaranteed to exert their maximum (level K) effort. We first claim that workers do not refuse tasks, as the action of refusing a task is weakly dominated by exerting level 0 effort.

Definition 3.3. Consider worker w_i; an action $\widehat{\mathbf{s}}_i(m) \in \mathbf{\Sigma}$ is weakly dominated given that w_i has skill type m if and only if $\exists \mathbf{s}'_i(m) \in \mathbf{\Sigma}$ such that $\forall \mathbf{s}_{-i}(\cdot)$, $u_i(\mathbf{s}'_i(m), \mathbf{s}_{-i}(\cdot), m) \geq u_i(\widehat{\mathbf{s}}_i(m), \mathbf{s}_{-i}(\cdot), m)$ holds and $\exists \mathbf{s}_{-i}(\cdot)$, $u_i(\mathbf{s}'_i(m), \mathbf{s}_{-i}(\cdot), m) > u_i(\widehat{\mathbf{s}}_i(m), \mathbf{s}_{-i}(\cdot), m)$.

Hence, a weakly dominated action is never the best action. In fact, the utility for refusing a task is zero. However, a worker earns a reward r with probability $1/n$ by exerting level 0 effort if all n workers exert level 0 effort. And he earns 0 if some worker exerts a higher level effort. This implies that workers never refuse tasks, or $\eta_m = 0$, $\forall m = 1, \ldots, M$. With this observation, we next derive the minimum reward needed to guarantee that workers exert their maximum (level K) effort.

LEMMA 3.4. *Consider our proposed incentive mechanism, and assume that two workers work on a task. At least one Bayesian Nash equilibrium exists. Given a skill type m, if $r > \frac{2c_{m,K}}{\beta_m}$, then each such equilibrium $(\mathbf{s}^*_1(\cdot), \ldots, \mathbf{s}^*_n(\cdot))$ satisfies $\mathbf{s}^*_i(m) = (0, \ldots, 0, 1)$ for all $i = 1, \ldots, n$.*

PROOF. The proof can be found in the Appendix. □

Remark. We prove the existence result by applying the work of Fudenberg and Tirole [1991], which states that the Bayesian Nash equilibrium exists in the finite incomplete information setting. We characterize the property of the Bayesian Nash equilibrium by showing the elimination of strictly dominated actions, and we eliminate all but level K effort for players of type m.

Lemma 3.4 implies that when a requester sets a reward of $r \geq 2c_{m,K}/\beta_m$, the *unique* best response for workers having skill level m is guaranteed to be $(0, \ldots, 0, 1)$ (i.e., they exert their maximum (level K) effort). When two workers work on a task, a requester needs to set a reward $r > \max\{2c_{1,K}/\beta_1, \ldots, 2c_{M,K}/\beta_M\}$ to guarantee that each participating worker exerts level K effort. However, even by setting such a reward, there is a probability $(1 - \beta_M)^2$ that two participating workers have skill types smaller than M. In such situations, a requester collects a solution with a benefit less than his cost, or $r + T$. One way to reduce such risk is to assign a task to more workers. This can be achieved by extending Lemma 3.4 to the case of $n \geq 2$.

LEMMA 3.5. *Consider our incentive mechanism, and assume that $n \geq 2$ workers work on a task. At least one Bayesian Nash equilibrium exists. Given a skill type m, if*

$$r > \frac{n\beta_m c_{m,K}}{\left(\sum_{\ell=1}^{m} \beta_\ell\right)^n - \left(\sum_{\ell=1}^{m-1} \beta_\ell\right)^n - n\beta_m\left(\sum_{\ell=1}^{m-1} \beta_\ell\right)^{n-1}}, \tag{2}$$

then each such equilibrium $(\mathbf{s}_1^*(\cdot), \ldots, \mathbf{s}_n^*(\cdot))$ satisfies $\mathbf{s}_i^*(m) = (0, \ldots, 0, 1)$ for all $i = 1, \ldots, n$.

PROOF. The proof can be found in the Appendix. □

Remark. We prove this lemma following a similar proof framework as in Lemma 3.4. The elimination of the strictly dominated strategy becomes more subtle than that of Lemma 3.4. This is a strong result because it specifies the minimum reward needed to guarantee that workers exert their maximum (level K) efforts. The following lemma states that the bound derived in Inequality (2) is tight in general.

LEMMA 3.6 (TIGHTNESS OF BOUND). *Consider our incentive mechanism, and assume that $n \geq 2$ workers work on a task. Given a skill type m, there exists $(\beta_1, \ldots, \beta_M) \in [0, 1]^M$ such that if*

$$r \leq \frac{n\beta_m c_{m,K}}{\left(\sum_{\ell=1}^{m} \beta_\ell\right)^n - \left(\sum_{\ell=1}^{m-1} \beta_\ell\right)^n - n\beta_m\left(\sum_{\ell=1}^{m-1} \beta_\ell\right)^{n-1}}, \tag{3}$$

then there exists at least one Bayesian Nash equilibrium $(\mathbf{s}_1^*(\cdot), \ldots, \mathbf{s}_n^*(\cdot))$, which satisfies $\mathbf{s}_i^*(m) \neq (0, \ldots, 0, 1)$ for some $i \in \{1, \ldots, n\}$.

PROOF. The proof can be found in the Appendix. □

Remark. We prove this lemma by identifying that when $(\beta_1, \ldots, \beta_M) = (0, \ldots, 0, 1)$ (i.e., tasks are assigned to skilled workers), workers may not provide their maximum effort if the reward satisfies (3). If the reward r does not satisfy Inequality (2), then type m workers may not provide their maximum effort. Namely, the bound derived in Inequality (2) is tight in general.

Requesters are only interested in high-quality solutions—that is, those produced by workers having skill level M exerting level K effort. We next derive the minimum number of workers, as well as the corresponding reward to guarantee that at least one such high-quality solution can be collected with high probability.

THEOREM 3.7. *To guarantee that at least one solution is submitted by a worker having skill level M using level K effort with probability greater than $1 - \alpha$, where $0 < \alpha < 1$, we need to assign a task to $n' = \max\{2, \lceil \log_{1-\beta_M} \alpha \rceil\}$ workers and set*

$$r > \frac{n'\beta_M c_{M,K}}{1 - (1 - \beta_M)^{n'} - n'\beta_M(1 - \beta_M)^{n'-1}}.$$

PROOF. The proof is similar to that of Lemma 3.5. □

Remark. When a task is assigned to $\lceil \log_{1-\beta_M} \alpha \rceil$ workers, one can claim with probability greater than $1 - \alpha$ that at least one participating worker has skill level M. One constraint is that a task must be assigned to at least two workers. Hence, we have $n' = \max\{2, \lceil \log_{1-\beta_M} \alpha \rceil\}$. Then one can apply Lemma 3.5 to set the appropriate reward.

Table I presents numerical results on the minimum number of workers needed (n') and the corresponding minimum reward, where $\alpha = 0.001$ (i.e., with probability of at least 0.999 of getting a good solution). When $\beta_M = 0.2$, one needs at least 31 workers to work on a task. The corresponding reward is at least $6.25c_{M,K}$. Note that as we

Table I. Minimum Number of Workers (n') and Minimum Reward (r) to Have a High-Quality Solution with Probability of 0.999

β_M	0.2	0.4	0.6	0.8
n'	31	14	8	5
r	$6.25c_{M,K}$	$5.65c_{M,K}$	$4.84c_{M,K}$	$4.03c_{M,K}$

Table II. Main Notation

$\mathcal{M}$	Worker skill levels $\mathcal{M} \triangleq \{1, \ldots, M\}$
$\mathcal{K}$	Worker effort levels $\mathcal{K} \triangleq \{0, 1, \ldots, K\}$
r, T	Reward, transaction fee of a task
r^*	Optimal reward
n	Number of workers working on a task
β_m	Probability of assigning a task to a worker having skill level m
η_m	Probability of a worker having skill level m refusing a task
$w_1, \ldots, w_n$	n workers participating in a task
θ_i, a_i	Skill type, effort level of w_i
$c_{m,k}$	Cost associated with skill type m and effort level k
$Q(m, k)$	Solution quality by skill type m and effort level k
$\mathbf{a}, \boldsymbol{\theta}$	$\mathbf{a} = (a_1, \ldots, a_n)$, $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)$
$V_{m,k}$	Benefit of a solution with quality $Q(m, k)$
$\boldsymbol{\Sigma}$	Action space of workers
$\mathbf{s}_i : \mathcal{M} \rightarrow \boldsymbol{\Sigma}$	Strategy of worker w_i
$\boldsymbol{\epsilon}$	Error matrix
s^*	Desirable strategy of workers
N_e	Number of errors a worker has committed since last reset
N_b	Number of time slots a worker has been blocked since last penalty
π_{N_e,N_b}	Stationary probability of a worker in state (N_e, N_b)
$\widetilde{N}_b, \widetilde{N}_e$	Blocking window threshold, active window threshold
$\widetilde{N}_b^*, \widetilde{N}_e^*$	Optimal blocking window threshold, optimal active window threshold
$\mathcal{E}$	Crowdsourcing system efficiency
$\widetilde{\mathcal{E}}$	Theoretical maximum system efficiency
N_W	Total number of workers (active and blocked) in the system
$R(\widetilde{N}_e, \widetilde{N}_b)$	Revenue with an $(\widetilde{N}_e, \widetilde{N}_b)$ reputation system
$\widetilde{R}$	Theoretical maximum revenue
$R_{\max}$	Maximum attainable revenue

increase the value of β_M, the minimum number of workers needed decreases as well as the minimum reward. This suggests that a crowdsourcing Web site needs to implement an accurate task assignment algorithm to reduce the reward that a requester pays so as to attract more requesters.

Summary. Our incentive mechanism guarantees that workers exert their maximum (level K) efforts. However, there is a nonzero probability $(1 - \beta_M)^n$ that no worker has skill type M. Even though these lower-skill workers will contribute at effort level K, the requester will be dissatisfied by their solutions. We next propose a reputation system to guarantee that low-skilled workers refuse tasks and that the administrator eventually assigns a task to a worker having skill type M. Table II summarizes the key notations in this article.

4. REPUTATION SYSTEM

We first present the design of our reputation system. We then couple this reputation system with our previously proposed incentive mechanism and apply a repeated game

framework to derive the minimum reward needed to guarantee that at least one high-quality solution is collected.

4.1. Reputation System Design

Our reputation system tracks the historical contributions of long-lived workers, who attempt to solve many tasks over a long period of time. Our system maintains the *reputation* of each worker and penalizes a worker when his reputation falls below a threshold. We integrate this reputation system into our incentive mechanism and use a *repeated game framework* to derive the minimum reward needed to sustain a unique subgame perfect equilibrium: skilled workers (skill type M) provide solutions at their maximum (level K) efforts, and less skilled workers (skill type $< M$) automatically refuse tasks. Let us first define *desirable strategy* and *error*.

Definition 4.1. s^* is a desirable strategy prescribing that a skilled worker (type M) provides a solution using his maximum (level K) effort, whereas less skilled workers (skill type $< M$) refuse the task. Formally,

$$s^* = \begin{cases} \text{exert level } K \text{ effort}, & \text{if a worker has skill level } M \\ \text{refuse a task}, & \text{if a worker has skill level smaller than } M. \end{cases}$$

Definition 4.2. An "error" occurs when a worker submits a solution whose quality is less than $Q(M, K)$.

Our reputation system relies on requesters rating workers using a *feedback rating* $\in \{0, 1\}$, where 0 indicates an error and 1 indicates that the solution has quality $Q(M, K)$ (i.e., the highest quality). We emphasize that a requester does *not* require any prior knowledge of worker skill types or effort levels but simply evaluates the quality of a solution when providing a feedback rating.

Our reputation system operates as follows. Each worker is tagged with a reputation index denoted by N_e, which records the number of errors since the last reset. Initially, each worker is tagged with zero errors, $N_e = 0$. Once a worker is reported to have committed an error (i.e., a requester reports a 0 rating), N_e is increased by one. Once the number of errors of a worker reaches a threshold $\widetilde{N}_e$, the reputation system triggers a punishment by blocking this worker from participating in any crowdsourcing activity for a given number, $\widetilde{N}_b$, of time slots. We refer to $\widetilde{N}_e$ and $\widetilde{N}_b$ as the active window threshold and the blocking window threshold, respectively. Let N_b denote the number of time slots that a worker has been blocked since he was last penalized. When a worker is blocked, N_b is increased by one at each time slot. After being blocked for $\widetilde{N}_b$ time slots, our reputation system activates the worker's account and resets N_e to zero. We denote this system as the $(\widetilde{N}_e, \widetilde{N}_b)$-*reputation system*. Formally, we index a worker's reputation via a pair (N_e, N_b), where $N_b = \widetilde{N}_b$ indicates that a worker is active, $N_e = \widetilde{N}_e$ indicates that a worker is blocked, $N_e \in \{0, 1, \ldots, \widetilde{N}_e\}$, and $N_b \in \{0, 1, \ldots, \widetilde{N}_b\}$. Formally, our reputation system is described by the following transition rules:

$$(N_e, N_b) \xrightarrow{(a_i, \theta_i)} \begin{cases} (N_e, \widetilde{N}_b), & \text{if } N_b = \widetilde{N}_b, N_e < \widetilde{N}_e, Q(a_i, \theta_i) = Q(K, M) \\ (N_e + 1, \widetilde{N}_b), & \text{if } N_b = \widetilde{N}_b, N_e < \widetilde{N}_e - 1, Q(a_i, \theta_i) < Q(K, M) \\ (\widetilde{N}_e, 0), & \text{if } N_b = \widetilde{N}_b, N_e = \widetilde{N}_e - 1, Q(a_i, \theta_i) < Q(K, M) \\ (\widetilde{N}_e, N_b + 1), & \text{if } N_e = \widetilde{N}_e, N_b < \widetilde{N}_b - 1 \\ (0, \widetilde{N}_b), & \text{if } N_e = \widetilde{N}_e, N_b = \widetilde{N}_b - 1. \end{cases}$$

We next integrate the preceding reputation system into our proposed incentive mechanism and use a repeated game framework to characterize long-lived worker strategic behaviors.

4.2. Repeated Game Formulation

We divide time into slots such that during each time slot, a worker can only register to solve at most one task and only one task is recommended to a worker. This can result in a low-skilled worker refusing a task, resulting in this worker being idle for the time slot. We integrate the $(\widetilde{N}_e, \widetilde{N}_b)$-reputation system into our proposed incentive mechanism. Recall from Section 2 that we formulated a Bayesian game to characterize a worker's strategic behavior in solving a task for one time slot. Long-lived workers attempt to maximize their long-term utility by solving many tasks over many time slots. We capture this scenario via a repeated game in which a worker repeatedly plays the Bayesian game. More precisely, consider a worker w_i, and let $\theta_i^t \in \mathcal{M}$ and $\mathbf{s}_i^t(\theta_i^t) \in \mathbf{\Sigma}$ denote his type and action at time t, respectively. Let $\mathbf{s}_{-i}^t(\cdot)$ denote a vector of strategies of the other $n-1$ workers at time slot t. The expected single-shot utility of worker w_i at time t is denoted by $u_i(\mathbf{s}_i^t(\theta_i^t), \mathbf{s}_{-i}^t(\cdot), \theta_i^t)$. A worker may not work on any task at time t because he is blocked or refuses a task. We define the single-shot utility for this idle worker w_i to be 0 ($u_i(\mathbf{s}_i^t(\theta_i^t), \mathbf{s}_{-i}^t(\cdot), \theta_i^t) = 0$). Let $0 < \delta < 1$ be a discount factor. We define the long-term discounted utility for worker w_i to be $u_i^\infty(\{\mathbf{s}_i^t(\theta_i^t)\}_{t=0}^\infty) = \sum_{t=0}^\infty \delta^t(1 - \mathbf{I}_{\{\text{idle}\}}^t) u_i(\mathbf{s}_i^t(\theta_i^t), \mathbf{s}_{-i}^t(\cdot), \theta_i^t)$, where $\mathbf{I}_{\{\text{idle}\}}^t = 1$ indicates that a worker is idle and $\mathbf{I}_{\{\text{idle}\}}^t = 0$ indicates that a worker participates a task. We assume that a worker chooses his strategy independently from slot to slot.

4.3. Sustaining Compliance via Proper Reward

We derive the minimum reward needed so that a unique subgame perfect equilibrium can be sustained where workers play the desirable strategy s^*. Furthermore, we quantify the impact of the active window threshold $\widetilde{N}_e$, blocked window threshold $\widetilde{N}_b$, and the probability of assigning a task to a most skilled worker β_M on this minimum reward. We state these results in the following theorem.

THEOREM 4.3. *Using the $(\widetilde{N}_e, \widetilde{N}_b)$-reputation system in our incentive mechanism and assuming that $n \geq 2$ workers work on a task, (1) if the active window threshold is $\widetilde{N}_e \geq 2$, then a worker with skill level less than M always deviates from s^* when it has reputation index $(N_e, \widetilde{N}_b)$, where $N_e = 0, 1, \ldots, \widetilde{N}_e - 2$, and (2) if $\widetilde{N}_e = 1$, $\delta > (1 + \beta_M(1-\beta_M)^{1-n}/n)^{-1}$, $\widetilde{N}_b \geq \lceil \ln(1 - n(1-\delta)(1-\beta_M)^{n-1}/(\delta\beta_M))/\ln\delta \rceil$, and*

$$r > n c_{M,K} \max\left\{ \frac{1}{\gamma^*}, \left(1 - \frac{n(1-\beta_M)^{n-1}(1-\delta)}{(1-\delta^{\widetilde{N}_b})\beta_M\delta}\right)^{-1} \right\}, \tag{4}$$

where $\gamma^* = \min\{\frac{1-(1-x)^n}{x} - n(1-x)^{n-1} | x \in [\beta_M, 1]\}$, then each worker will not unilaterally deviate from s^*. Furthermore, this subgame perfect equilibrium is unique.

PROOF. The proof can be found in the Appendix. □

Remark. We first show that for workers having skill type M, by eliminating the strictly dominated action we can eliminate all but level K effort. This implies that workers having skill type M playing the desirable strategy s^* is a unique best response. Then we apply the one-shot deviation principle to show that workers with type lower than M will play the desirable strategy s^* irrespective of other workers' actions, under the condition that workers having skill type M playing the desirable strategy s^*.

Table III. Minimum Reward to Ensure That Workers Play the Desirable Strategy s^* ($n = 3$, $\delta = 0.999$)

$\widetilde{N}_b$	20	30	40
$r(\beta_M = 0.2)$	$5.82c_{M,K}$	$5.77c_{M,K}$	$5.77c_{M,K}$
$r(\beta_M = 0.4)$	$3.47c_{M,K}$	$3.41c_{M,K}$	$3.41c_{M,K}$
$r(\beta_M = 0.6)$	$3.13c_{M,K}$	$3.08c_{M,K}$	$3.06c_{M,K}$
$r(\beta_M = 0.8)$	$3.02c_{M,K}$	$3.02c_{M,K}$	$3.01c_{M,K}$

Workers are guaranteed to play s^*, as the subgame perfect equilibrium is unique. Theorem 4.3 states that one needs to set the active window threshold to $\widetilde{N}_e = 1$ because workers committing errors while in states $(0, \widetilde{N}_b), (1, \widetilde{N}_b), \ldots, (\widetilde{N}_e\text{–}2, \widetilde{N}_b)$ incur no penalties. Examining (4), one observes that as we increase the blocking window threshold $\widetilde{N}_b$, we decrease the minimum reward. This suggests that a crowdsourcing provider should set $\widetilde{N}_b$ as large as possible, provided that there are no errors in reputation updating.

Table III presents numerical examples on the minimum reward derived in Equation (4), where $n = 3$, $\delta = 0.999$. Table III depicts β_M, $\widetilde{N}_b$, and the corresponding minimum reward. When $\beta_M = 0.2$ and $\widetilde{N}_b = 20$, the reward is $r = 5.82c_{M,K}$. As $\widetilde{N}_b$ increases from 20 to 40, the reward decreases from $5.82c_{M,K}$ to $5.77c_{M,K}$ (i.e., a 1% decrease). Similar decreases are shown for $\beta_M = 0.4, 0.6, 0.8$. When $\widetilde{N}_b = 20$, as β_M increases from 0.2 to 0.8, the minimum reward decreases from $5.82c_{M,K}$ to $3.02c_{M,K}$, a significant reduction. It is important to observe that the minimum rewards are smaller than in Table I. Namely, our reputation system decreases the reward payment, which is attractive to requesters.

Summary. Our model thus far assumes that requesters must provide perfect feedback ratings to indicate whether a worker committed an error or not. This is difficult to achieve in practice, because human factors such as inherent personal preferences or biases may lead to some erroneous feedback ratings on the part of a requester. We explore the robustness of our reputation system against such human factors in the next section.

5. HUMAN FACTORS

We present a probabilistic model to characterize errors in ratings caused by human factors such as inherent biases. We derive the minimum reward needed to tolerate such errors in ratings. We show that the minimum reward increases in $\widetilde{N}_e$ and decreases in $\widetilde{N}_b$, and the number of active workers increases in $\widetilde{N}_e$ and decreases in $\widetilde{N}_b$.

5.1. Model for Human Factors

In real-world crowdsourcing systems, requester feedback ratings may not accurately reflect the quality of a solution. Feedback ratings may be distorted due to inherent user biases—for instance, some critical requesters may always express low ratings, whereas lenient requesters may always express high ratings. Such factors can lead to the following situation: a solution of quality $Q(M, K)$ receives a feedback rating of zero or a solution of quality lower than $Q(M, K)$ receives a feedback rating of one. We refer to these as erroneous ratings. We assume that a solution of quality $Q(m, k)$, where $m \in \mathcal{M}$, $k \in \mathcal{K}$, receives a rating of zero with probability $\epsilon_{m,k} \in [0, 1]$ ($\Pr[\text{rating } 0 | Q(m, k)] = \epsilon_{m,k}$). We assume that an erroneous rating occurs with a small probability, for instance, $\epsilon_{M,K} << 1$, and that $\epsilon_{m,k} \approx 1$ holds for all m, k such that $Q(m, k) < Q(M, K)$. We also assume that $\epsilon_{m,k} > \epsilon_{m',k'}$ if and only if $Q(m, k) < Q(m', k')$, which means that the higher the quality of a solution, the lower the chance it receives rating 0. We define an M by

the $K+1$ error matrix, $\boldsymbol{\epsilon} = [\epsilon_{m,k}]$. This article focuses on the case that the requester can identify the highest quality solution regardless of whatever he is biased toward providing higher or lower feedback ratings.

5.2. Quantify the Impact of Human Factors

In this section, we first derive the feasible space of $(\widetilde{N}_e, \widetilde{N}_b)$ and show that the incentive and reputation mechanisms can tolerate erroneous feedback ratings provided that we slightly increase the reward. We also show that this reward increases in $\widetilde{N}_e$ and decreases in $\widetilde{N}_b$.

THEOREM 5.1. *Consider a combined incentive / $(\widetilde{N}_e, \widetilde{N}_b)$-reputation system, and assume that $n \geq 2$ workers work on a task. The feasible space of $(\widetilde{N}_e, \widetilde{N}_b)$ is*

$$\widetilde{N}_e \leq \frac{-\ln\left(n\epsilon_{M,K}/((1-\beta_M)^{1-n}-1+\epsilon_{M-1,K})\right)}{\ln(1-\delta+\delta\beta_M\epsilon_{M,K})-\ln(\delta\beta_M\epsilon_{M,K})}, \tag{5}$$

$$\widetilde{N}_b \geq \ln \frac{1-\frac{n\epsilon_{M,K}}{(1-\beta_M)^{1-n}-1+\epsilon_{M-1,K}}\left(\frac{1-\delta+\delta\beta_M\epsilon_{M,K}}{\delta\beta_M\epsilon_{M,K}}\right)^{\widetilde{N}_e}}{1-n\epsilon_{M,K}/((1-\beta_M)^{1-n}-1+\epsilon_{M-1,K})} / \ln\delta. \tag{6}$$

For each feasible pair of $(\widetilde{N}_e, \widetilde{N}_b)$, if reward r satisfies

$$r > nc_{M,K}\max\left\{\frac{1}{\gamma^*}, \left(1-\left(\left(1+(1-\delta)/(\delta\beta_M\epsilon_{M,K})\right)^{\widetilde{N}_e}-\delta^{\widetilde{N}_b}\right)\right.$$
$$\left.\frac{(1-\beta_M)^{n-1}n\epsilon_{M,K}}{(1-\delta^{\widetilde{N}_b})(1-(1-\epsilon_{M-1,K})(1-\beta_M)^{n-1})}\right)^{-1}\Bigg\}, \tag{7}$$

where $\gamma^* = \min\{\frac{1-(1-x)^n}{x} - n(1-x)^{n-1} | x \in [\beta_M, 1]\}$, then s^* is a subgame perfect equilibrium. Furthermore, this subgame perfect equilibrium is unique. Last, r increases with respect to $\widetilde{N}_e$ and decreases with respect to $\widetilde{N}_b$.

PROOF. The proof can be found in the Appendix. □

Remark. We prove this lemma following a similar proof framework as in Theorem 4.3. As we consider human factors (i.e., errors in reputation updating), the analysis becomes more complicated.

Theorem 5.1 shows that our proposed incentive mechanism combined with our reputation system is robust against imperfect feedback and can be practically applied in real-world crowdsourcing systems. Theorem 5.1 suggests that a crowdsourcing provider should decrease $\widetilde{N}_e$ (or increase $\widetilde{N}_b$) so as to decrease the reward paid out by requesters. However, the side effect is a decrease in the number of active workers (as we will show with Theorem 5.3).

Table IV presents the minimum reward derived from (7), when $n=3$, $\epsilon_{M,K}=0.05$, $\epsilon_{M-1,K}=0.95$, $\beta_M=0.6$, $\delta=0.999$. To apply Theorem 5.1, we take $\widetilde{N}_e \leq 114$. Table IV depicts the active window threshold $\widetilde{N}_e$, the blocking window threshold $\widetilde{N}_b$, and the corresponding minimum reward. When $\widetilde{N}_b=100$, as the active window threshold increases from $\widetilde{N}_e=10$ to $\widetilde{N}_e=30$, the reward increases from $3.42c_{M,K}$ to $5.46c_{M,K}$, a significant increase. When $\widetilde{N}_e=20$, an increase in the blocking window threshold from $\widetilde{N}_b=70$ to $\widetilde{N}_b=90$ results in a slight decrease in the reward from $4.66c_{M,K}$ to $4.20c_{M,K}$.

Table IV. Minimum Reward to Guarantee Desirable Strategy s^*, with $\epsilon_{M,K} = 0.05$, $\epsilon_{M-1,K} = 0.95$, $\beta_M = 0.6$, $\delta = 0.999$, $n = 3$

$\widetilde{N}_e$	10	20	30
r ($\widetilde{N}_b = 100$)	$3.42c_{M,K}$	$4.05c_{M,K}$	$5.46c_{M,K}$
$\widetilde{N}_b$	70	80	90
r ($\widetilde{N}_e = 20$)	$4.66c_{M,K}$	$4.39c_{M,K}$	$4.20c_{M,K}$

We next model our combined incentive reputation system as a Markov chain and show that the number of active workers increases in $\widetilde{N}_e$ and decreases in $\widetilde{N}_b$.

5.3. Evolving Dynamics of a Worker's Reputation

We model the evolving dynamics of a worker's reputation (when he plays s^*) in the presence of error matrix $\boldsymbol{\epsilon}$ as a Markov chain. A worker is in state (N_e, N_b) if this worker has reputation index (N_e, N_b). Based on the reputation transition rule, the state space of our Markov chain can be expressed as $\{(\widetilde{N}_e, N_b), N_b = 0, 1, \ldots, \widetilde{N}_b - 1\} \cup \{(N_e, \widetilde{N}_b), N_e = 0, 1, \ldots, \widetilde{N}_e - 1\}$. Applying error matrix $\boldsymbol{\epsilon}$, the one-step state transition probability of our Markov chain can be expressed as follows:

$$\Pr\left[(N_e', N_b')|(N_e, N_b)\right] = \begin{cases} \beta_M \epsilon_{M,K}, & \text{if } N_b = \widetilde{N}_b, (N_e', N_b') = (N_e + 1, \widetilde{N}_b), N_e < \widetilde{N}_e - 1 \\ \beta_M \epsilon_{M,K}, & \text{if } N_b = \widetilde{N}_b, (N_e', N_b') = (\widetilde{N}_e, 0), N_e = \widetilde{N}_e - 1 \\ 1 - \beta_M \epsilon_{M,K}, & \text{if } N_b = \widetilde{N}_b, (N_e', N_b') = (N_e, N_b), \forall N_e \\ 1, & \text{if } N_e = \widetilde{N}_e, (N_e', N_b') = (\widetilde{N}_e, N_b + 1), N_b < \widetilde{N}_b - 1 \\ 1, & \text{if } N_e = \widetilde{N}_e, (N_e', N_b') = (0, \widetilde{N}_b), N_b = \widetilde{N}_b - 1. \end{cases} \tag{8}$$

It is easy to check that the preceding Markov chain is aperiodic and ergodic. We next express its stationary distribution.

Definition 5.2. Denote by π_{N_e,N_b} the stationary probability of a worker in state (N_e, N_b), where $N_e = 0, 1, \ldots, \widetilde{N}_e - 1$ and $N_b = 0, 1, \ldots, \widetilde{N}_b - 1$.

THEOREM 5.3. *The stationary probability of a worker in state* (N_e, N_b) *can be expressed as*

$$\pi_{N_e,N_b} = \begin{cases} \dfrac{1}{\widetilde{N}_e + \widetilde{N}_b \beta_M \epsilon_{M,K}}, & N_e = 0, 1, \ldots, \widetilde{N}_e - 1, N_b = \widetilde{N}_b \\ \dfrac{\beta_M \epsilon_{M,K}}{\widetilde{N}_e + \widetilde{N}_b \beta_M \epsilon_{M,K}}, & N_b = 0, 1, \ldots, \widetilde{N}_b - 1, N_e = \widetilde{N}_e \\ 0, & \textit{otherwise.} \end{cases}$$

PROOF. One can prove this by substituting the expressions for π_{N_e,N_b} into $\boldsymbol{\pi} = \boldsymbol{\pi}\mathbf{P}$, where matrix $\mathbf{P}$ is expressed as $\mathbf{P} = [\Pr[(N_e', N_b')|(N_e, N_b)]]$ with $\Pr[(N_e', N_b')|(N_e, N_b)]$ given by (8). □

Remark. It follows that the probability that a worker is active is $\widetilde{N}_e/(\widetilde{N}_e + \widetilde{N}_b \beta_M \epsilon_{M,K})$, which increases in $\widetilde{N}_e$ and decreases in $\widetilde{N}_b$. Thus, either increasing $\widetilde{N}_e$ or decreasing $\widetilde{N}_b$ increases the total number of active workers.

Summary. Decreasing $\widetilde{N}_e$ or increasing $\widetilde{N}_b$ reduces the reward that a requester must pay to guarantee that workers play the desirable strategy s^*. However, this has the side effect of decreasing the total number of active workers. We next explore the effect of $\widetilde{N}_e, \widetilde{N}_b$ on different design trade-offs (i.e., system efficiency, revenue.).

6. TRADE-OFF IN REPUTATION SYSTEM

We present a class of metrics to quantify system efficiency (i.e., the total number of tasks that can be solved for a given reward) and revenue (i.e., the total amount of transaction fees). We formulate an optimization framework to select appropriate values of $\widetilde{N}_e$ and $\widetilde{N}_b$. We identify extremal design points corresponding to the maximum efficiency and maximum revenue, as well as the optimal trade-off between efficiency and revenue.

6.1. Design Trade-Offs and Metrics

System efficiency. A decrease in the active window threshold $\widetilde{N}_e$, or an increase in the blocking window threshold $\widetilde{N}_b$, leads to an increase in the efficiency of a crowdsourcing system. Specifically, this leads to the solution of more tasks for a given reward budget. We formally quantify system efficiency as follows.

Definition 6.1. We denote by $\mathcal{E}$ the crowdsourcing system efficiency expressed as

$$\mathcal{E} \triangleq \frac{c_{M,K}}{r}.$$

The larger the value of $\mathcal{E}$, the higher the crowdsourcing system efficiency. Increasing system efficiency $\mathcal{E}$ can increase the number of requesters willing to participate. From a crowdsourcing Web site owner' perspective, he may want to maximize the efficiency (i.e., maximize $\mathcal{E}$) subject to the requirement that workers provide their maximum effort. Ideally, a crowdsourcing system can achieve the theoretical maximum efficiency if it is perfect—that is, perfect reputation updates (there is no erroneous ratings in reputation updating) and perfect task assignments ($\beta_M = 1$, tasks are assigned to high-skilled workers with probability 1). We denote this theoretical maximum system efficiency by $\widetilde{\mathcal{E}}$. One of our objectives is to show the effectiveness of our mechanisms via studying the gap between $\mathcal{E}$ and $\widetilde{\mathcal{E}}$.

LEMMA 6.2. *Consider a combined incentive/*$(\widetilde{N}_e, \widetilde{N}_b)$*-reputation system, and assume that* $n \geq 2$ *workers work on a task. The theoretical maximum system efficiency is* $\widetilde{\mathcal{E}} = \frac{1}{n}$.

PROOF. The proof can be found in the Appendix. □

Revenue. However, decreasing $\widetilde{N}_e$ or increasing $\widetilde{N}_b$ also decreases revenue. Theorem 5.3 states that decreasing $\widetilde{N}_e$ (or increasing $\widetilde{N}_b$) results in a decrease in the number of active workers. A decrease in the number of active workers leads to a decrease in the total amount of transaction fees (i.e., revenue loss). We now quantify the long-term revenue of a crowdsourcing system under each selection of $(\widetilde{N}_e, \widetilde{N}_b)$. Suppose that the total number of workers (active and blocked) in the system is N_w. As derived in Theorem 5.3, the steady-state probability of an active worker being in state $(N_e, \widetilde{N}_b)$ is $\pi_{N_e,\widetilde{N}_b} = 1/(\widetilde{N}_e + \widetilde{N}_b\beta_M\epsilon_{M,K})$. Thus, the expected number of active workers at each time slot is $N_w \sum_{N_e=0}^{\widetilde{N}_e-1} \pi_{N_e,\widetilde{N}_b} = N_w\widetilde{N}_e/(\widetilde{N}_e + \widetilde{N}_b\beta_M\epsilon_{M,K})$. Based on this, we introduce a measure to quantify revenue for a crowdsourcing system as follows.

Table V. Reputation System Design to Maximize Efficiency
($\epsilon_{M-1,K} = 1 - \epsilon_{M,K}$, $\delta = 0.999$, $\beta_M = 0.8$, $n = 3$)

$\epsilon_{M,K}$	$\widetilde{N}_e^*$	$\widetilde{N}_b^*$	r^*	$\mathcal{E}^*/\widetilde{\mathcal{E}}$	$R(\widetilde{N}_e^*, \widetilde{N}_b^*)/\widetilde{R}$
0.05	1	∞	$3.02c_{M,K}$	0.9933	0
0.10	1	∞	$3.04c_{M,K}$	0.9867	0
0.15	1	∞	$3.06c_{M,K}$	0.9804	0

Definition 6.3. Denote $R(\widetilde{N}_e, \widetilde{N}_b)$ by the expected revenue in one time slot. We express it as

$$R(\widetilde{N}_e, \widetilde{N}_b) \triangleq \frac{N_w \widetilde{N}_e}{\widetilde{N}_e + \widetilde{N}_b \beta_M \epsilon_{M,K}} \left(\frac{\beta_M T}{n}\right),$$

where T is the transaction fee for a task as defined in Section 2.

A crowdsourcing system can achieve the theoretical maximum revenue if it is perfect—that is, there are no erroneous ratings in reputation updating and tasks are assigned to high-skilled workers with probability 1, or $\beta_M = 1$. We use $\widetilde{R}$ to denote the theoretical maximum revenue. It can be easily expressed as

$$\widetilde{R} \triangleq \frac{N_w T \beta_M}{n}.$$

From a crowdsourcing Web site owner's perspective, he wants to maximize the revenue under the constraint that workers play the desirable strategy s^*. We define the following notations to present our results.

Definition 6.4. Denote $\widetilde{N}_e^*, \widetilde{N}_b^*$ by the optimal active window threshold and optimal blocking window threshold, respectively. Denote r^*, $\mathcal{E}^*$, $R(\widetilde{N}_e^*, \widetilde{N}_b^*)$ by the reward, system efficiency, and revenue respectively corresponding to $\widetilde{N}_e^*, \widetilde{N}_b^*$.

6.2. Reputation System to Maximize Efficiency

One possible goal is to choose $(\widetilde{N}_e, \widetilde{N}_b)$ so as to maximize system efficiency. The optimization formulation is

$$\max_{(\widetilde{N}_e, \widetilde{N}_b)} \mathcal{E} = \frac{c_{M,K}}{r}$$
$$\textit{subject to} \qquad \text{Equations (5), (6), and (7),}$$

where constraints (5), (6), and (7) ensure that workers play s^*.

The preceding optimization problem has a solution if and only if $\delta \geq (1 - \beta_M \epsilon_{M,K} + \beta_M((1-\beta_M)^{1-n} - 1 + \epsilon_{M-1,K})/n)^{-1}$. One observes that the solution is $\widetilde{N}_e^* = 1$ and $\widetilde{N}_b^* = \infty$, because decreasing $\widetilde{N}_e$ or increasing $\widetilde{N}_b$ increases efficiency. This solution implies that a worker will be blocked permanently once he commits an error. In the presence of erroneous ratings, all workers will be blocked. In other words, eventually there will be no active worker, and the revenue is $R(\widetilde{N}_e^*, \widetilde{N}_b^*) = 0$. Table V shows numerical examples on the maximum efficiency. One observes that when $\epsilon_{M,K} = 0.05$, the optimal reward is $3.02c_{M,K}$. The corresponding system efficiency only attains $\mathcal{E}^*/\widetilde{\mathcal{E}} = 0.9933$ of the theoretical maximum system efficiency. This implies a quite high system efficiency. As $\epsilon_{M,K}$ increases from 0.05 to 0.15, the reward increases from $3.02c_{M,K}$ to $3.06c_{M,K}$, and the system efficiency drops from $0.9933\widetilde{\mathcal{E}}$ to $0.9804\widetilde{\mathcal{E}}$. In other words, the maximum efficiency is robust against erroneous ratings. However, there is no revenue, as eventually all workers are blocked out (i.e., $\widetilde{N}_b^* = \infty$).

Table VI. Reputation System Design to Maximize Revenue
($\epsilon_{M-1,K} = 1 - \epsilon_{M,K}, \delta = 0.999, \beta_M = 0.8, n = 3$)

$\epsilon_{M,K}$	$\widetilde{N}_e^*$	$\widetilde{N}_b^*$	r^*	$\mathcal{E}^*/\widetilde{\mathcal{E}}$	$R(\widetilde{N}_e^*, \widetilde{N}_b^*)/\widetilde{R}$
0.05	6	1	$91.22c_{M,K}$	0.0330	0.994
0.1	6	1	$54.88c_{M,K}$	0.0546	0.987
0.15	18	3	$407.37c_{M,K}$	0.0075	0.980

6.3. Reputation System to Maximize Revenue

Another goal is to choose $(\widetilde{N}_e, \widetilde{N}_b)$ so as to maximize revenue. The optimization formulation is

$$\max_{(\widetilde{N}_e, \widetilde{N}_b)} R(\widetilde{N}_e, \widetilde{N}_b) = \frac{N_w \widetilde{N}_e}{\widetilde{N}_e + \widetilde{N}_b \beta_M \epsilon_{M,K}} \left(\frac{\beta_M T}{n}\right)$$

subject to Equations (5), (6), and (7).

The solution to the optimization problem yields the maximum attainable revenue $R_{\max}$ under the constraint that workers play s^*.

The preceding optimization problem has at least one solution. We locate the optimal solution by searching all pairs $(\widetilde{N}_e, \widetilde{N}_b)$, where the value of $\widetilde{N}_b$ equals the right-hand side in (6). In fact, for a given active window threshold $\widetilde{N}_e$, (6) expresses a lower bound for the blocking window threshold $\widetilde{N}_b$. Recall that increasing the blocking window threshold decreases revenue. We therefore set $\widetilde{N}_b$ to the right-hand side in (6).

Table VI presents examples of solutions to the preceding optimization problem. When $\epsilon_{M,K} = 0.05$, the solution is $\widetilde{N}_e^* = 6$ and $\widetilde{N}_b^* = 1$, which achieves a revenue of $0.994\widetilde{R}$. This suggests that the revenue is close to the theoretical upper bound $\widetilde{R}$. As $\epsilon_{M,K}$ increases from 0.05 to 0.15, revenue drops from $0.994\widetilde{R}$ to $0.980\widetilde{R}$. In other words, maximum revenue is robust against erroneous feedback ratings. However, efficiency is low. In fact, reward r^* is at least $54.88c_{M,K}$, which attains $\mathcal{E}^*/\widetilde{\mathcal{E}} = 0.0546$ of the theoretical maximum efficiency. We next explore the optimal trade-off between efficiency and revenue.

6.4. Exploring the Optimal Trade-Off Curve

We identify the optimal efficiency/revenue trade-off curve. This curve enables us to explore the full design space of the reputation system. We show that a small sacrifice in revenue can lead to a significant increase in system efficiency.

We consider the following problem: *given a minimum acceptable revenue* $R(\widetilde{N}_e, \widetilde{N}_b) \geq \xi R_{\max}, \xi \in (0, 1]$, *what values of* $\widetilde{N}_e$, $\widetilde{N}_b$ *maximize system efficiency?* The optimization formulation is

$$\max_{(\widetilde{N}_e, \widetilde{N}_b)} \mathcal{E} = \frac{c_{M,K}}{r}$$

subject to Equations (5), (6), and (7),

$$\frac{N_w \widetilde{N}_e}{\widetilde{N}_e + \widetilde{N}_b \beta_M \epsilon_{M,K}} \frac{\beta_M T}{n} \geq \xi R_{\max}. \tag{9}$$

We find the optimal solution of the preceding optimization problem by searching the feasible solution space. Given an active window threshold $\widetilde{N}_e$, (9) yields an upper bound for $\widetilde{N}_b$ (i.e., $\widetilde{N}_b \leq \lfloor (\frac{N_w T \beta_M}{n \xi R_{\max}} - 1) \frac{\widetilde{N}_e}{\epsilon_{M,K} \beta_M} \rfloor$). Feasible values of $\widetilde{N}_b$ for this given $\widetilde{N}_e$ range from the right-hand side in (6) to this upper bound. If this upper bound is smaller than the

Table VII. Optimal Trade-Offs Between Efficiency and Revenue
($\epsilon_{M-1,K} = 1 - \epsilon_{M,K}$, $\delta = 0.999$, $\beta_M = 0.8$, $n = 3$, $\xi = 0.95$)

$\epsilon_{M,K}$	$\widetilde{N}_e^*$	$\widetilde{N}_b^*$	r^*	$\mathcal{E}^*/\widetilde{\mathcal{E}}$	$R(\widetilde{N}_e^*, \widetilde{N}_b^*)$
0.05	5	7	$3.41c_{M,K}$	0.8796	$0.95R_{\max}$
0.1	6	5	$3.75c_{M,K}$	0.8001	$0.95R_{\max}$
0.15	5	3	$4.13c_{M,K}$	0.7263	$0.95R_{\max}$

right-hand side in (6), then $\widetilde{N}_e$ is invalid. Searching all feasible pairs $(\widetilde{N}_e, \widetilde{N}_b)$ allows us to find the optimal solution.

Table VII shows examples of the optimal trade-offs between efficiency and revenue. We adopt the same setting as for Table VI. We set $\xi = 0.95$ (we allow a 5% decrease in the revenue). One observes that when $\epsilon_{M,K} = 0.05$, the corresponding reputation system design is $\widetilde{N}_e^* = 5$, $\widetilde{N}_b^* = 7$, and we obtain a reward $r^* = 3.41c_{M,K}$, which attains $\mathcal{E}^*/\widetilde{\mathcal{E}} = 0.8976$ of the theoretical maximum system efficiency. As shown in Table VI, the reward for $\xi = 1$ is $r^* = 91.22c_{M,K}$, which attains an efficiency of $\mathcal{E}^* = 0.0330\widetilde{\mathcal{E}}$. This implies a significant decrease in reward, or an improvement on system efficiency. Similar results also hold for $\epsilon_{M,K} = 0.1, 0.15$. This implies that a small sacrifice in the revenue can lead to a significant improvement in system efficiency. Furthermore, our reputation mechanism is robust against erroneous ratings.

7. EXPERIMENTS ON REAL DATA

We carry out experiments on a real dataset from UpWork. We infer model parameters from the dataset and use them to determine the right reward and values of $(\widetilde{N}_e, \widetilde{N}_b)$ for the reputation system in the UpWork setting.

7.1. UpWork Dataset

We developed a crawler based on Scrapy [2008] to obtain historical transaction data from UpWork [Elance 1999] instead of using its API. In UpWork, workers (or freelancers) provide solutions to various types of tasks (e.g., translation, programming, writing). Each worker posts his skills and sets his price—for example, a worker provides an English translation service and sets a price of $10 per hour. Requesters can select workers to solve a specific task. When a task is finished, requesters can rate the quality of the solution using one to five stars. More stars imply higher quality. We crawled historical task transactions from UpWork, which are of type "programmer" and conducted from June 22, 2000 to May 16, 2015. Each transaction contains the task ID, the worker ID, the requester ID, the timestamp, the price of the task, and the rating for the solution. Our dataset contains 150,297 task transactions of type "programming" and 9,592 workers.

7.2. Inferring Model Parameters

We first infer M, K, and worker skill types from the dataset relying on the five-star solution rating scale. A solution submitted by a worker at his maximum effort reflects his skill set. The rating scale can categorize such solutions into five types. Namely, worker skills are classified into $M = 5$ levels. A worker having the highest skill type earns a rating ranging from one star to five stars as an indication of his effort level. We interpret it as a means to classify effort into $K + 1 = 5$ levels yielding $K = 4$.

Now we infer the cost $c_{m,k}$ in providing a service. A worker sets a price for providing a service. We interpret this price as the cost to that worker to exert maximum effort (i.e., $c_{m,4}$) if he has skill type m. This procedure allows us to obtain $c_{m,4}$, $\forall m$. There is insufficient information to identify costs for the other effort levels (i.e., $c_{m,3}, c_{m,2}, c_{m,1}, c_{m,0}$). We will see later that the inferred $c_{m,4}$, $\forall m$ suffices for our purpose. We infer the skill level

$m \in \{0, 1, 2, 3, 4\}$ of a worker via his majority rating, as this rating reflects the overall solution quality that a worker can provide. We can also infer the task assignment probabilities $\beta_1, \ldots, \beta_5$. We infer the value of β_m via the fraction of service transactions that involve workers having skill type m—that is,

$$\beta_m = \frac{\text{\# of service transactions involving skill type } m \text{ worker}}{\text{total \# of service transactions}}.$$

Using this simple rule on our dataset, we obtain $\beta_1 = \frac{1}{150297} = 6.65 \cdot 10^{-6}$, $\beta_2 = \frac{1}{150297} = 6.65 \cdot 10^{-6}$, $\beta_3 = \frac{3}{150297} = 1.996 \cdot 10^{-5}$, $\beta_4 = \frac{361}{150297} = 0.0024$, $\beta_5 = \frac{149931}{150297} = 0.997565$.

We now infer error matrix $\boldsymbol{\epsilon}$. Our model interprets the five-star rating scale as follows: 5 means a high-quality solution (i.e., feedback rating 1), whereas a rating smaller than 5 implies an error (i.e., feedback rating 0). In general, it is impossible to know the exact level of effort that a worker exerted in solving a task. We assume that workers provide the highest effort because there is insufficient information to identify multiple effort levels. We infer $\epsilon_{m,K}$ as the fraction of ratings smaller than 5 received by workers at skill type m:

$$\epsilon_{m,K} = \frac{\text{\# of ratings smaller than 5 for skill type } m \text{ worker}}{\text{\# of ratings for skill type } m \text{ worker}}.$$

Using this simple rule on our dataset, we obtain $\epsilon_{1,4} = \frac{1}{1} = 1, \epsilon_{2,4} = \frac{1}{1} = 1, \epsilon_{3,4} = \frac{3}{3} = 1, \epsilon_{4,4} = \frac{286}{361} = 0.7922, \epsilon_{5,4} = \frac{13386}{149931} = 0.08928$. These values only give partial information of the error matrix $\boldsymbol{\epsilon}$. We are unable to estimate the remaining elements of $\boldsymbol{\epsilon}$. We will see later that this does not cause any problem. We like to point out that the values of $\epsilon_{1,4}, \epsilon_{2,4}, \epsilon_{3,4}$ are noisy because of the size of our dataset. However accurate values of these three parameters are not critical to our results. In fact, only the values of $\epsilon_{4,4}$ and $\epsilon_{5,4}$ are critical, and the dataset from which we estimate them is sufficiently large.

7.3. Reward Settlement and Reputation System Design

We use the preceding inferred model parameters to demonstrate how to determine the reward payment and set $\widetilde{N}_e, \widetilde{N}_b$ for the reputation system in the UpWork setting. Through this, we show the applicability of our incentive mechanism and reputation system. We first determine the maximum efficiency that UpWork can achieve. We set $n = 2$ and $\delta = 0.999$. Applying results in Section 6.2 yields $\widetilde{N}_e^* = 1$ and $\widetilde{N}_b^* = \infty$. The reward, maximum efficiency, and revenue are

$$r^* = 2.00488 c_{5,4}, \ \frac{\mathcal{E}^*}{\widetilde{\mathcal{E}}} = 0.9976, \ \frac{R(\widetilde{N}_e^*, \widetilde{N}_b^*)}{\widetilde{R}} = 0.$$

We observe that the system efficiency attains $\frac{\mathcal{E}^*}{\widetilde{\mathcal{E}}} = 0.9976$ of the theoretical maximum efficiency. This implies a quite high system efficiency. However, the revenue is quite low (i.e., $R(\widetilde{N}_e^*, \widetilde{N}_b^*) = 0$).

We now explore the maximum revenue that UpWork can achieve. Using our inferred model parameters and applying results in Section 6.3 yields the following:

$$\widetilde{N}_e^* = 106, \ \widetilde{N}_b^* = 1, \ r^* = 164.1599 c_{5,4}, \ \frac{\mathcal{E}^*}{\widetilde{\mathcal{E}}} = 0.0122, \ \frac{R(\widetilde{N}_e^*, \widetilde{N}_b^*)}{\widetilde{R}} = 0.99916.$$

UpWork can attain a large revenue (i.e., $0.99916\widetilde{R}$). However, the reward is $r^* = 164.1599 c_{5,4}$, and system efficiency only attains $\mathcal{E}^*/\widetilde{\mathcal{E}} = 0.0122$ of the theoretical maximum system efficiency. This implies a quite low efficiency.

Now we explore the optimal trade-off between efficiency and revenue for UpWork. This is obtained by using our inferred model parameters and applying the results

Table VIII. Reputation System Design to Explore the Optimal Trade-Off Between Efficiency and Revenue for UpWork

ξ	$\widetilde{N}_e^*$	$\widetilde{N}_b^*$	$R(\widetilde{N}_e^*, \widetilde{N}_b^*)$	r^*	$\mathcal{E}^*/\widetilde{\mathcal{E}}$
1	106	1	$R_{\max}$	$164.1599c_{5,4}$	0.0122
0.98	13	3	$0.98R_{\max}$	$2.0474c_{5,4}$	0.9769
0.96	13	6	$0.96R_{\max}$	$2.0239c_{5,4}$	0.9882
0.94	7	5	$0.94R_{\max}$	$2.0152c_{5,4}$	0.9925
0.92	13	12	$0.92R_{\max}$	$2.0124c_{5,4}$	0.9939

in Section 6.4, where we set $\delta = 0.999, n = 2$. We present this optimal trade-off in Table VIII. We observe that a 2% decrease in the revenue from $R_{\max}$ to $0.98R_{\max}$ leads to a significant decrease in reward from $164.1599c_{5,4}$ to $2.0474c_{5,4}$, and consequently a significant increase in system efficiency from $0.0122\widetilde{\mathcal{E}}$ to $0.9769\widetilde{\mathcal{E}}$. An 8% decrease in the revenue to $0.92R_{\max}$ leads to a system efficiency of $\mathcal{E}^* = 0.9939\widetilde{\mathcal{E}}$. It is interesting to observe that a further reduction in the revenue only improves efficiency slightly (i.e., improves system efficiency less than 1%). One observes that $\widetilde{N}_e^* = 13, \widetilde{N}_b^* = 6$ is a good choice for UpWork with reward $r^* = 2.0239c_{5,4}$, where $c_{5,4}$ is inferred in Section 7.2. This choice attains a good balance between system revenue and system efficiency—that is, it achieves 98.82% of the theoretical maximum system efficiency while only sacrificing 4% revenue.

8. RELATED WORK

Research on crowdsourcing has been very active recently, ranging from application design [Heer and Bostock 2010], cheat detection [Hirth et al. 2011], and quality management [Ipeirotis et al. 2010; Karger et al. 2013] to incentive design [Mason and Watts 2010; Jain et al. 2009; Xie et al. 2014; Zhang and van der Schaar 2012], among others. A recent survey can be found in Yuen et al. [2011]. In this work, we study the incentive and reputation mechanism design for crowdsourcing systems.

Much work focuses on determining the minimum reward so as to attract users (workers and requesters). One approach is to estimate worker wages and benefits [Bacon et al. 2012; Horton and Chilton 2010], where models and methods are built and their parameters inferred. However, these works do not consider the strategic behavior of workers. A multiarmed bandit algorithm to dynamically determine the appropriate reward was developed in Tran-Thanh et al. [2012]. The strategic behavior of workers was not considered, and low-skilled workers could participate to contribute low-quality solutions. Another approach is to design auction-based pricing mechanisms [Chawla et al. 2012; DiPalantino and Vojnovic 2009; Singer and Mittal 2013; Singla and Krause 2013]. For example, in Singer and Mittal [2013], tasks are dynamically priced and allocated to workers based on their bids. However, these pricing schemes do not address the dilemma of free riding or denial of payment. Our work addresses this dilemma. In practice, the complexity in deploying these auction-based pricing mechanisms is high, and determining the price usually involves high delays. Our incentive and reputation mechanisms are simple to implement. In addition, Chawla et al. [2012], DiPalantino and Vojnovic [2009], Singer and Mittal [2013], and Singla and Krause [2013] only model the interaction between workers and requesters as a single-shot game, whereas our work further extends it to a repeated game to represent behavior of long-lived workers. Finally, we show that our work is highly applicable to UpWork.

Several works were done to design incentive protocols so that workers provide high-quality contributions. Note that simply attracting participation is not enough, as workers may solve tasks at low levels of effort. Only a few works [Jain et al. 2009; Shaw et al. 2011; Xie et al. 2014] have investigated this important question, and usually

their models are simple (e.g, single-shot game). Jin et al. [2015] designed a mechanism to incentivize user participation for mobile crowdsensing applications. The mechanism is based on reverse combinatorial auctions and considers a single-shot game. We focus on a general crowdsourcing application scenario and formulate a repeated game to understand the strategic behavior of long-lived workers. We also study how to couple a reputation mechanism with an incentive mechanism so as to improve system efficiency. Recently, a few reputation-based incentive protocols [Ho et al. 2012; Xiao et al. 2013; Zhang and van der Schaar 2012] have been proposed. These works were motivated by the seminal work [Kandori 1992] on reputation mechanism and social norm. To the best of our knowledge, this is the first work that combines the incentive mechanism and reputation mechanism. Our work differs from theirs in the following four technical aspects. First, they [Ho et al. 2012; Xiao et al. 2013; Zhang and van der Schaar 2012] assume a payment scheme either before tasks begin or after tasks are complete. We propose a payment mechanism via an administrator to avoid free riding or denial of payment. Second, they consider a simplified model where all workers are highly skilled (one skill type) and there are only two level of efforts (i.e., high effort or no effort), and each task is solved by only one worker. We present a more general model to capture practical scenarios where workers have many different levels of skill and can exert more than two levels of effort to solve a task, and a task can be solved by multiple workers. Third, they did not consider the impact of task assignment. We address this point by presenting a probabilistic model for the task assignment process. Fourth, we show the applicability of our work via performing experiments on UpWork dataset. Their works did not cover this aspect.

9. CONCLUSION

This article presents a class of effective incentive and reputation mechanisms for crowdsourcing applications. Our incentive mechanism allows multiple workers to solve a task and splits the reward among workers based on requester evaluations of the quality of solutions. We derived the minimum reward needed to guarantee that workers provide their maximum efforts. Our reputation mechanism ensures that low-skilled workers do not provide low-quality solutions by tracking workers' historical contributions and penalizing those workers having poor reputation. Our incentive and reputation mechanisms are robust against human biases in evaluating solution quality. We presented an optimization framework to select parameters for our reputation mechanism. We showed that there is a trade-off between system efficiency and revenue, and we presented the optimal trade-off curve between system efficiency and revenue. We also performed experiments using a real-world dataset from UpWork. We inferred model parameters from this data, used them to determine the right reward payment, and selected the parameters of our incentive and reputation mechanisms for UpWork. Our incentive and reputation mechanisms are shown to achieve 98.82% of the maximum system efficiency while only sacrificing 4% revenue.

APPENDIX

A.1. Proof of Lemma 3.4

The Bayesian Nash equilibrium exists in the finite incomplete information setting [Fudenberg and Tirole 1991] (i.e., finite types and finite actions). We complete this proof by showing that by elimination of strictly dominated actions, we eliminate all but level K effort for players of type m.

We first show that level 0 effort is a strictly dominated action for type m players. Consider w_i of type m. Let $\mathbf{e}_\ell = (0, \ldots, 0, 1, 0, \ldots, 0)$ denote a $K+1$-dimensional vector

such that only the ℓ-th entry is 1 and other entries are 0. Then $\mathbf{e}_{K+1}$, $\mathbf{e}_1$ represent level K effort and level 0 effort, respectively. We claim that level 0 effort is strictly dominated by level K effort—that is, $u_i(\mathbf{e}_{K+1}, \mathbf{s}_{-i}(\cdot), m) > u_i(\mathbf{e}_1, \mathbf{s}_{-i}(\cdot), m)$ for all $\mathbf{s}_{-i}(\cdot)$. Let $R_i(\mathbf{e}_k, \mathbf{s}_{-i}(\cdot), m) = u_i(\mathbf{e}_k, \mathbf{s}_{-i}(\cdot), m) + c_{m,k}$ denote the expected reward for player w_i. And let $\Delta R_i(k, k', m) = R_i(\mathbf{e}_k, \mathbf{s}_{-i}(\cdot), m) - R_i(\mathbf{e}_{k'}, \mathbf{s}_{-i}(\cdot), m)$. Then we have $u_i(\mathbf{e}_{K+1}, \mathbf{s}_{-i}(\cdot), m) - u_i(\mathbf{e}_1, \mathbf{s}_{-i}(\cdot), m) = \Delta R_i(K+1, 1, m) - c_{m,K} + c_{m,0}$. Let w_{-i} denote the player that w_i plays against, and let θ_{-i} denote its type. We next show that $\Delta R_i(K+1, 1, m)$ is minimized when w_{-i} exert level 0 effort if of type lower than m and otherwise exert level K effort.

Case 1: $\theta_{-i} < m$. If w_{-i} exerts an effort higher than level 0, then $E[\Delta R_i(K+1, 1, m) | \theta_{-i} < m] = r$. If w_{-i} exerts level 0 effort, then $E[\Delta R_i(K+1, 1, m) | \theta_{-i} < m] = \frac{r}{2}$. Thus, $E[\Delta R_i(K+1, 1, m) | \theta_{-i} < m]$ attains the minimal value $\frac{r}{2}$ when w_{-i} exerts level 0 effort. Note that $\Pr[\theta_{-i} < m] = \sum_{\ell=1}^{m-1} \beta_\ell$.

Case 2: $\theta_{-i} > m$. If w_{-i} exerts the level K effort, then $E[\Delta R_i(K+1, 1, m) | \theta_{-i} > m] = 0$. Note that $E[\Delta R_i(K+1, 1, m) | \theta_{-i} > m] \geq 0$. Hence, $E[\Delta R_i(K+1, 1, m) | \theta_{-i} > m]$ attains the minimal value 0 when w_{-i} exerts level K effort.

Case 3: $\theta_i = m$. If w_{-i} exerts the level K or level 0 effort, then $E[\Delta R_i(K+1, 1, m) | \theta_{-i} = m] = \frac{r}{2}$. Otherwise, $E[\Delta R_i(K+1, 1, m) | \theta_{-i} = m] = r$. Hence, $E[\Delta R_i(K+1, 1, m) | \theta_{-i} = m]$ attains the minimal value $\frac{r}{2}$ when w_{-i} exerts level K effort. Note that $\Pr[\theta_{-i} = m] = \beta_m$. Combine them together, and we have

$$\begin{aligned} & u_i(\mathbf{e}_{K+1}, \mathbf{s}_{-i}(\cdot), m) - u_i(\mathbf{e}_1, \mathbf{s}_{-i}(\cdot), m) \\ &= E[\Delta R_i(K+1, 1, m)] - c_{m,K} + c_{m,0} \\ &\geq \frac{r}{2} \Pr[\theta_{-i} < m] + \frac{r}{2} \Pr[\theta_{-i} = m] - c_{m,K} + c_{m,0} \\ &> 0, \end{aligned}$$

where the last step follows that $r \geq 2c_{m,K}/\beta_m$. Thus, level 0 effort is a strictly dominated action for w_i of type m. Similarly, the same statements holds for w_{-i} of type m.

We now consider the reduced game, where level 0 effort is eliminated for type m players. With a similar derivation as earlier, we obtain that for this reduced game, level 1 effort is a strictly dominated action for type m players. Repeating this elimination of strictly dominated action, we finally eliminate all but the level K effort for skill type m players.

A.2. Proof of Lemma 3.5

This proof extends the proof of Lemma 3.4. The argument for the existence of Bayesian Nash equilibrium is the same as for Lemma 3.4.

We first show that level 0 effort is a strictly dominated action for type m players. Consider player w_i. Let $w_{-i} = [w_j]_{j \neq i}$ denote the other players except w_i. We claim that $\Delta R_i(K+1, 1, m)$ is minimized when players from w_{-i} exert level 0 effort, if of type lower than m, and otherwise exert level K effort. Let $\theta_i' = \max\{\theta_j | j \neq i\}$.

Case 1: $\theta_i' < m$. All players in w_{-i} are of type lower than m. If some w_{-i} exert an effort higher than 0, then $E[\Delta R_i(K+1, 1, m) | \theta_i' < m] = r$. Otherwise, $E[\Delta R_i(K+1, 1, m) | \theta_i' < m] = \frac{(n-1)r}{n}$. Thus, $E[\Delta R_i(K+1, 1, m) | \theta_i' < m]$ attains the minimal value $\frac{(n-1)r}{n}$ when all w_{-i} exert level 0 effort. Note that $\Pr[\theta_i' < m] = (\sum_{\ell=1}^{m-1} \beta_\ell)^{n-1}$.

Case 2: $\theta_i' > m$. All players in w_{-i} are of type higher than m. If some of them exert level K effort, then $E[\Delta R_i(K+1, 1, m) | \theta_i' > m] = 0$. Note that $E[\Delta R_i(K+1, 1, m) | \theta_i' > m] \geq 0$.

Thus, $E[\Delta R_i(K+1,1,m)|\theta_i' > m]$ attains the minimal value 0 when players with type higher than m exert level K effort.

Case 3: $\theta_i' = m$. Suppose that ℓ of w_{-i} are of type m, and if none of them exerts level K effort, then $E[\Delta R_i(K+1,1,m)|\theta_i' = m] \geq \frac{(n-1)r}{n}$, and the the lower bound is attained when all w_{-i} exert level 0 effort. If $\ell' \geq 1$ of type m workers from w_{-i} exert level K effort, then $E[\Delta R_i(K+1,1,m)|\theta_i' = m] = \frac{r}{\ell'+1}$. Thus, $E[\Delta R_i(K+1,1,m)|\theta_i' = m]$ attains the minimum value of $\frac{r}{\ell+1}$ when type m workers in w_{-i} exert level K effort. Note that $\Pr[\ell, \theta_i' = m] = \binom{n-1}{\ell}\beta_m^\ell(\sum_{\ell'=1}^{m-1}\beta_{\ell'})^{n-1-\ell}$.

Combine them together, and we have

$$
\begin{aligned}
&u_i(\mathbf{e}_{K+1}, \mathbf{s}_{-i}(\cdot), m) - u_i(\mathbf{e}_1, \mathbf{s}_{-i}(\cdot), m) \\
&\geq \frac{(n-1)r}{n}\Pr[\theta_i' < m] + \sum_{\ell=1}^{n-1}\frac{r\Pr[\ell, \theta_i' = m]}{\ell+1} - c_{m,K} + c_{m,0} \\
&\geq \frac{r}{n\beta_m}\left(\left(\sum_{\ell=1}^{m}\beta_\ell\right)^n - \left(\sum_{\ell=1}^{m-1}\beta_\ell\right)^n - n\beta_m\left(\sum_{\ell=1}^{m-1}\beta_\ell\right)^{n-1}\right) - c_{m,K} + c_{m,0} \\
&> 0.
\end{aligned}
$$

Repeating this elimination of strictly dominated action, we eliminate all but the level K effort for type m players. This proof is then complete.

A.3. Proof of Lemma 3.6

Consider our incentive mechanism, and assume that $n \geq 2$ workers work on a task. Given a skill type m, consider one possible selection of $(\beta_1, \ldots, \beta_M)$ such that $(\beta_1, \ldots, \beta_M) = (0, \ldots, 0, 1, 0, \ldots, 0)$, where 1 is the m-th entry. Then Inequality (3) can be simplified to $r \leq nc_{m,K}$. In this setting of $(\beta_1, \ldots, \beta_M)$, each participating worker is of skill type m. We first consider $r = nc_{m,K}$. We consider $r < nc_{m,K}$ later. When $r = nc_{m,K}$, then there is an equilibrium that $n-1$ workers provide their maximum effort and one worker provides his minimum effort (i.e., zero effort). This means that Lemma 3.6 holds. Now consider $r < nc_{m,K}$. It is impossible that each worker providing his maximum effort is an equilibrium. Note that there exists at least one equilibrium. This lemma is then complete.

A.4. Proof of Theorem 4.3

We first show that subgame perfect equilibrium exists, and for each such equilibrium, type M players exert level K effort. Applying the one-shot deviation principle, it is easy to check that workers playing s^* is a subgame perfect equilibrium. For type M workers, by elimination of strictly dominated action, we can eliminate all but level K effort. Examining Equation (1), one can observe that if some workers of skill type lower than M refuse tasks, $\Pr[\theta_i = M]$ will be increased. Then with a similar derivation as Lemma 3.4, we conclude that by setting a reward of $r > 1/\gamma^*$, type M players exert level K effort for each subgame perfect equilibrium.

We now show that workers with type lower than M will play s^* irrespective of other workers' actions, under the condition that type M workers exert level K effort. We say that a worker is compliant if he plays s^*. Suppose that all workers are compliant. Let $u^\infty(\mathbf{I}_a, I)$ denote the long-term utility for a player with initial state $(\mathbf{I}_a, I)$. Consider a type $m < M$ player at state $(1, I)$ who is assigned a task. If he refuses the task, then his long-term utility is $\delta u^\infty(1, I)$. If this worker solves this task, then his long-term utility is $\delta u^\infty(1, I+1) + u_i(\mathbf{s}_i(m), \mathbf{s}_{-i}(\cdot), m)$. Since type M workers are compliant, thus we obtain

the upper bound $u_i(\mathbf{s}_i(m), \mathbf{s}_{-i}(\cdot), m) < (1-\beta_M)^{n-1}r$. Hence, this worker will be compliant irrespective of other workers' actions if for all $I = 0, \ldots, W_a - 1$

$$\delta \Delta u(1, I) > (1-\beta_M)^{n-1}r, \tag{10}$$

where $\Delta u(1, I) = u^\infty(1, I) - u^\infty(1, I+1)$, if $I < W_a - 1$, and $\Delta u(1, I) = u^\infty(1, I) - u^\infty(0, 0)$, if $I = W_a - 1$. One can easily obtain that $u^\infty(1, I) = \frac{\beta_M}{1-\delta}\left(\frac{r}{n} - c_{M,K}\right)$ for all $I = 0, \ldots, W_a - 1$. And $u^\infty(0, I) = \delta^{W_b - I}\frac{\beta_M}{1-\delta}(\frac{r}{n} - c_{M,K})$ for all $I = 0, \ldots, W_a - 1$. Applying these expressions for the long-term utilities to Inequality (10), we complete this proof.

A.5. Proof of Theorem 5.1

With a similar derivation as Theorem 4.3, we obtain that subgame perfect equilibrium exists, and for each of such equilibrium, type M workers exert level K effort. We next show that players with lower skill type will play s^* irrespective of the actions of other workers, under the condition that type M workers exert level K effort. Consider a type $m < M$ player at state $(1, I)$ who is assigned a task. If this player refuses this task, then his long-term utility is $\delta u^\infty(1, I)$. If this player solves this task exerting level k effort, then his long-term utility is $\delta(1-\epsilon_{m,k})u^\infty(1, I) + \delta\epsilon_{m,k}u^\infty(1, I+1) + u_i(\mathbf{s}_i(m), \mathbf{s}_{-i}(\cdot), m)$. Observe that $u_i(\mathbf{s}_i(m), \mathbf{s}_{-i}(\cdot), m) < (1-\beta_M)^{n-1}r$. Thus, this player will be compliant irrespective of other workers' actions if $\epsilon_{m,k}\delta[u^\infty(1, I) - u^\infty(1, I+1)] > (1-\beta_M)^{n-1}r$. To make this inequality hold for all $m = 1, \ldots, M-1, k = 0, \ldots, K$, we only need for all $I = 0, \ldots, W_a - 1$

$$\Delta u(1, I) > \frac{(1-\beta_M)^{n-1}r}{\delta\epsilon_{M-1,K}}, \tag{11}$$

where $\Delta u(1, I)$ is defined in the proof of Theorem 4.3. The long-term utility satisfies the following equations:

$$\begin{cases} u^\infty(1, I) = \dfrac{u^\infty(1, I+1) + \left(\frac{r}{n} - c_{M,K}\right)/(\delta\epsilon_{M,K})}{(1 - \delta + \delta\beta_M\epsilon_{M,K})/(\delta\beta_M\epsilon_{M,K})}, \quad I \le W_a - 2 \\ u^\infty(1, W_a - 1) = \dfrac{u^\infty(0, 0) + \left(\frac{r}{n} - c_{M,K}\right)/(\delta\epsilon_{M,K})}{(1 - \delta + \delta\beta_M\epsilon_{M,K})/(\delta\beta_M\epsilon_{M,K})} \\ u^\infty(0, I) = \delta u^\infty(0, I+1), \quad I \le W_b - 2 \\ u^\infty(0, W_b - 1) = \delta u^\infty(1, 0). \end{cases}$$

Solving these equations, we obtain

$$u^\infty(1, I) = \frac{\beta_M}{1-\delta}\left(\frac{r}{n} - c_{M,K}\right)\left[1 + (\delta^{W_b} - 1)\left(\frac{1 - \delta + \delta\beta_M\epsilon_{M,K}}{\delta\beta_M\epsilon_{M,K}}\right)^I \Bigg/ \left(\left(\frac{1 - \delta + \delta\beta_M\epsilon_{M,K}}{\delta\beta_M\epsilon_{M,K}}\right)^{W_a} - \delta^{W_b}\right)\right]$$

$\forall I = 0, \ldots, W_a - 1$, and

$$u^\infty(0, I) = \frac{\delta^{W_b - I}\beta_M}{1-\delta}\left(\frac{r}{n} - c_{M,K}\right)\left[1 + (\delta^{W_b} - 1)\Bigg/\left(\left(\frac{1 - \delta + \delta\beta_M\epsilon_{M,K}}{\delta\beta_M\epsilon_{M,K}}\right)^{W_a} - \delta^{W_b}\right)\right],$$

$\forall I < W_b$. Substituting these expressions for the long-term utilities to Inequality (11), we obtain the expression for reward r active window threshold N_e and blocking window threshold N_b, respectively. Evaluating the first-order derivative of the expression for r in terms of N_e and N_b, respectively, we complete this proof.

A.6. Proof of Lemma 6.2

Note that the crowdsourcing system is perfect (i.e., there are no erroneous ratings in reputation updating and $\beta_M = 1$), and say that tasks are assigned to high-skilled workers with probability 1. This implies that each participating worker has skill type M. In this scenario, to guarantee all of them providing their maximum effort, we need a reward of at least $nc_{M,K}$. This means that the maximum achievable efficiency is $\widetilde{\mathcal{E}} = c_{M,K}/(nc_{M,K}) = 1/n$.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their helpful comments.

REFERENCES

Lada A. Adamic, Jun Zhang, Eytan Bakshy, and Mark S. Ackerman. 2008. Knowledge sharing and Yahoo answers: Everyone knows something. In *Proceedings of the 17th International Conference on World Wide Web (WWW'08)*.

Amazon Mechanical Turk. 2008. Amazon Mechanical Turk Home Page. Retrieved March 29, 2016, from https://www.mturk.com.

David F. Bacon, David C. Parkes, Yiling Chen, Malvika Rao, Ian Kash, and Manu Sridharan. 2012. Predicting your own effort. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'12)*.

Shuchi Chawla, Jason D. Hartline, and Balasubramanian Sivan. 2012. Optimal crowdsourcing contests. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'12)*.

Clickworker. 2001. Clickworker Home Page. Retrieved March 29, 2016, from http://www.clickworker.com/.

Dominic DiPalantino and Milan Vojnovic. 2009. Crowdsourcing and all-pay auctions. In *Proceedings of the 10th ACM Conference on Electronic Commerce (EC'09)*.

Elance. 1999. Elance Home Page. Retrieved March 29, 2016, from https://www.elance.com/.

Fiverr. 2010. Fiverr Home Page. Retrieved March 29, 2016, from https://www.fiverr.com/.

Drew Fudenberg and Jean Tirole. 1991. *Game Theory*. Cambridge.

Jeffrey Heer and Michael Bostock. 2010. Crowdsourcing graphical perception: Using Mechanical Turk to assess visualization design. In *Proceedings of the 28th SIGCHI Conference on Human Factors in Computing Systems (CHI'10)*.

M. Hirth, T. Hossfeld, and P. Tran-Gia. 2011. Cost-optimal validation mechanisms and cheat-detection for crowdsourcing platforms. In *Proceedings of the 5th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS'11)*.

Chien-Ju Ho and Jennifer Wortman Vaughan. 2012. Online task assignment in crowdsourcing markets. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*.

C.-J. Ho, Y. Zhang, J. Vaughan, and M. van der Schaar. 2012. Towards social norm design for crowdsourcing markets. In *Proceedings of the 4th Human Computation Workshop (HCOMP'12)*.

John Horton. 2010. Online labor markets. In *Internet and Network Economics.* Lecture Notes in Computer Science, Vol. 6484. Springer, 515–522.

John Joseph Horton and Lydia B. Chilton. 2010. The labor economics of paid crowdsourcing. In *Proceedings of the 11th ACM Conference on Electronic Commerce (EC'10)*.

Jeff Howe. 2006. The rise of crowdsourcing. *Wired Magazine* 14, 6, 1–4.

Jeff Howe. 2008. *Crowdsourcing: Why the Power of the Crowd Is Driving the Future of Business*. Crown Business.

Panagiotis G. Ipeirotis. 2010. Analyzing the Amazon Mechanical Turk marketplace. *XRDS: Crossroads, The ACM Magazine for Students* 17, 2, 16–21.

Panagiotis G. Ipeirotis, Foster Provost, and Jing Wang. 2010. Quality management on Amazon Mechanical Turk. In *Proceedings of the 2nd Human Computation Workshop (HCOMP'10)*.

Shaili Jain, Yiling Chen, and David C. Parkes. 2009. Designing incentives for online question and answer forums. In *Proceedings of the 10th ACM Conference on Electronic Commerce (EC'09)*.

Haiming Jin, Lu Su, Danyang Chen, Klara Nahrstedt, and Jinhui Xu. 2015. Quality of information aware incentive mechanisms for mobile crowd sensing systems. In *Proceedings of the 16th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'15)*.

Michihiro Kandori. 1992. Social norms and community enforcement. *Review of Economic Studies* 59, 1, 63–80.

David R. Karger, Sewoong Oh, and Devavrat Shah. 2013. Efficient crowdsourcing for multi-class labeling. In *Proceedings of the ACM SIGMETRICS/International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS'13)*.

Winter Mason and Siddharth Suri. 2012. Conducting behavioral research on Amazon Mechanical Turk. *Behavior Research Methods* 44, 1, 1–23. http://dx.doi.org/10.3758/s13428-011-0124-6.

Winter Mason and Duncan J. Watts. 2010. Financial incentives and the "performance of crowds." *ACM SIGKDD Explorations* 11, 2, 100–108.

Microtask. 2003. Microtask Home Page. Retrieved March 29, 2016, from http://www.microtask.com/.

Gabriele Paolacci, Jesse Chandler, and Panagiotis Ipeirotis. 2010. Running experiments on Amazon Mechanical Turk. *Judgment and Decision Making* 5, 5, 411–419.

Scrapy. 2008. Scrapy Home Page. Retrieved March 29, 2016, from http://scrapy.org/.

Aaron D. Shaw, John J. Horton, and Daniel L. Chen. 2011. Designing incentives for inexpert human raters. In *Proceedings of the 14th ACM Conference on Computer Supported Cooperative Work (CSCW'11)*.

Yaron Singer and Manas Mittal. 2013. Pricing mechanisms for crowdsourcing markets. In *Proceedings of the 22nd International Conference on World Wide Web (WWW'13)*.

Adish Singla and Andreas Krause. 2013. Truthful incentives in crowdsourcing tasks using regret minimization mechanisms. In *Proceedings of the 22nd International Conference on World Wide Web (WWW'13)*.

Taskcn. 2010. Taskcn Home Page. Retrieved March 29, 2016, from http://www.taskcn.com/.

Threadless. 2000. Threadless Home Page. Retrieved March 29, 2016, from http://www.threadless.com/.

Long Tran-Thanh, Sebastian Stein, Alex Rogers, and Nicholas R. Jennings. 2012. Efficient crowdsourcing of unknown experts using multi-armed bandits. In *Proceedings of the 20th European Conference on Artificial Intelligence (ECAI'12)*. 768–773.

Luis Von Ahn. 2006. Games with a purpose. *Computer* 39, 6, 92–94.

Wikipedia. 2003a. Macrotasking. Retrieved March 29, 2016, from https://en.wikipedia.org/wiki/Macrotasking.

Wikipedia. 2003b. Microwork. Retrieved March 29, 2016, from https://en.wikipedia.org/wiki/Microwork.

Y. Xiao, Y. Zhang, and M. van der Schaar. 2013. Socially-optimal design of crowdsourcing platforms with reputation update errors. In *Proceedings of the 38th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'13)*.

Hong Xie, John C. S. Lui, and Wenjie Jiang. 2014. Mathematical modeling of crowdsourcing systems: Incentive mechanism and rating system design. In *Proceedings of the IEEE 22nd International Symposium on Modelling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS'14)*.

Hong Xie, John C. S. Lui, and Don Towsley. 2015. Incentive and reputation mechanisms for online crowdsourcing systems. In *Proceedings of the 23rd IEEE/ACM International Symposium on Quality of Service (IWQoS'15)*.

Yahoo! Answers. 2012. Yahoo! Answers Home Page. Retrieved March 29, 2016, from http://answers.yahoo.com.

Man-Ching Yuen, I. King, and Kwong-Sak Leung. 2011. A survey of crowdsourcing systems. In *Proceedings of the IEEE 3rd International Conference on Privacy, Security, Risk, and Trust (PASSAT'11) and Proceedings of the IEEE 3rd International Conference on Social Computing (SocialCom'11)*.

Y. Zhang and M. van der Schaar. 2012. Reputation-based incentive protocols in crowdsourcing applications. In *Proceedings of the 31st Annual IEEE International Conference on Computer Communications (INFOCOM'12)*.

Received September 2015; revised January 2016; accepted February 2016