Incentive and Reputation Mechanisms for Online Crowdsourcing Systems

Hong Xie, John C.S. Lui
Department of Computer Science & Engineering,
The Chinese University of Hong Kong, Hong Kong,
{hxie,cslui}@cse.cuhk.edu.hk

Don Towsley
Department of Computer Science
University of Massachusetts
towsley@cs.umass.edu

***Abstract*—Nowadays, online crowdsourcing services are quite common such as Amazon Mechanical Turk and Google Helpouts. For such online services, it is important to attract "workers" to provide high-quality solutions to the "tasks" outsourced by "requesters". We present a unified study of incentive and reputation mechanisms for online crowdsourcing systems. We first design an mechanism to incentivize workers provide their maximum effort, which allows multiple workers to solve a task, splits the reward among workers based on requester evaluations of the solution quality. We design a *reputation mechanism*, which ensures that low-skilled workers do not provide low-quality solutions by tracking workers' historical contributions, and penalizing those workers having poor reputation. We show that our incentive and reputation mechanisms are robust against human biases in solution quality evaluation.**

***Keywords*—crowdsourcing; incentive; reputation;**

I. Introduction

Nowadays, many tasks, e.g., image labeling, translation, etc., which are easy for humans, 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 [8]. 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 contents [8]. Many online crowdsourcing systems has emerged over the past decade. Typical examples are Amazon Mechanical Turk [1], Google Helpouts [4], Yahoo! Answers [19], etc. A variety of "tasks" can be outsourced to online crowdsourcing systems, e.g., image labeling [9], question answering [19], product design [14]. In general, an online crowdsourcing system creates an online labor markets to solve tasks by soliciting contributions from a large group of users. Formally, a typical online crowdsourcing system classifies users into "*requesters*" and "*workers*". Requesters outsource tasks to workers, who in turn solve tasks and reply to requesters with solutions. Requesters set appropriate rewards for tasks, and the reward will be granted to workers who contribute to solve the task.

In general, online crowdsourcing systems face two fundamental challenges: (1) how to consistently solicit active participation of users (requesters and workers), (2) how to solicit high-quality solutions from workers [7], [12]. *This paper focuses on the design of incentive and reputation mechanisms to solicit active participation, and on how to incentivize high quality solutions by workers in a general setting*. Designing incentive and reputation mechanisms for online crowdsourcing systems faces three challenges. First is the heterogeneity of workers with different skill sets. Only high-skilled workers are capable of providing high-quality solutions. It is challenging to guarantee that high-skilled workers are recruited. The second is the need to incentivize workers to participate and exert their maximum effort. If the reward is small, workers may not participate or just provide a small amount of efforts leading to a task unsolved or low quality solutions. Requesters, on the other hand, always want to minimize the reward payment. Note that simply attracting workers to participate is not enough, because they may not have sufficient skills. The third challenge is how to define and update worker 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. This paper addresses the above challenges. Our contributions are:

- We design a class of mechanisms to incentivize workers to participate and exert 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 guarantee that high-skilled workers provide their maximum effort.
- We design a class of *reputation mechanisms* to prevent low-skilled workers from providing low quality solutions. We show that our incentive and reputation mechanisms are robust against human biases on the part of requesters in evaluating solution quality.

This paper organizes as follows. In Sec.II, we present the crowdsourcing system model. Sec.III and Sec.IV present the design and analysis of our incentive and reputation mechanisms. In Sec.V, we demonstrate the robustness of our incentive and reputation mechanisms against human factors. Related work is given in Sec.VI and Sec.VII concludes.

II. 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. For each task, a requester sets a non-negative

978-1-4673-7113-1/15/$31.00 ©2015 IEEE

reward of r to be split among contributing workers, and the requester also pays a transaction fee of $T \geq 0$ to the crowdsourcing system. Hence, the total payment by the requester is $r+T$. 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 is are only known to workers.

A. 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 [1], allow workers to select tasks, while others like Clickworker [2] require the system administrator to assign tasks to workers. We use a general probabilistic model to represent all possible cases of a 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 $(\beta_1, \ldots, \beta_M)$. For example, $(\beta_1, \ldots, \beta_M) = (0, \ldots, 0, 1)$ states 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.

B. Model for Worker Action

Our model allows $n \geq 1$ workers to solve a task and we denote them 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 re-assigned to another worker until exactly n workers commit to solve this task. Without loss of generality, we denote the probability that a worker having skill type m refuses a tasks by $\eta_m \in [0, 1]$. Using Bayes' rule, one can easily express the probability mass function (pmf) of θ_i as

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

We emphasize that $\theta_1, \ldots, \theta_n$ are independent and identically distributed (IID) random variables.

Worker w_i exerts effort $a_i \in \mathcal{E} \triangleq \{0, 1, \ldots, K\}$ to complete a task. The larger a_i the greater the effort. Moreover, there is a cost associated with each effort level, i.e., $c_{m,k}$ is the cost associated with effort level $k \in \mathcal{E}$ and skill set type $m \in \mathcal{M}$, where $c_{m,K} > \cdots > c_{m,0} = 0$. Here, $c_{m,0} = 0$ corresponds to a type m worker choosing to be a free-rider while incurring zero cost (or no effort at all). Furthermore, the cost of exerting the same level of effort decreases as a worker is more skilled in solving a given task, i.e., $c_{M,k} \leq \cdots \leq c_{1,k}$, where $k \in \mathcal{E}$.

C. 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 $Q(m, k) \in [0, 1]$, $(m \in \mathcal{M}, k \in \mathcal{E})$, 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 the solution has higher quality. Our model assumes that $Q(m, k)$ exhibits three properties: (1) A higher effort level implies higher quality: $Q(m, k) > Q(m, k')$, $\forall k > k'$. (2) A higher worker skill level implies higher quality: $Q(m, k) > Q(m', k), \forall m > m', k > 0$. (3) The quality of a zero effort solution is zero: $Q(M, 0) = \cdots = Q(1, 0) = 0$.

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 $k \in \mathcal{E}$ contributes a benefit of $V_{m,k} \geq 0$ to a requester. Our model assumes that $V_{m,k}$ exhibits the following properties: (1) The higher the quality of a solution the greater benefit to a requester: $V_{m,k} > V_{m',k'}$, if and only if $Q(m, k) > Q(m', k')$. (2) Two solutions bring the same benefit if and only if they are of 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 highest quality one. If a tie occurs, i.e., multiple solutions having 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 $V_{M,K} > r + T$ in order 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 $Q(M, K)$, i.e., $V_{M,k} < r + T$ 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.

III. Incentive Mechanism

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 freeride, i.e., take the reward while exerting no effort. However, if a requester pays after receiving solutions, he may refuse to pay the workers, which discourages workers from participating. Finally, incentive mechanism design is complicated by the heterogeneity of workers with different skill sets, and they aim to maximize their payoffs by strategically exerting one of $K + 1$ levels of effort.

We consider the following incentive mechanism. To post a task, a requester must submit its associated reward r and transaction fee T to the crowdsourcing system administrator. Each 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 the quality of solutions and selects the highest quality one. If there is a tie, one solution is randomly

selected. We refer to the highest quality solutions as winner(s) and the other solutions as *losers*. Finally, the requester notifies the system administrator who the winners are, and the system administrator distributes the reward r evenly to those winners. Workers will be encouraged to participate because winners equally share the reward. One important property of this scheme is that the requester cannot refuse to pay workers because the reward r and the transaction fee T are held by the administrator. Hence he does not benefit from providing false feedback e.g., notifying the administrator there is no winner, since the reward is not returned to the requester.

A. Formulating the Bayesian Game

In our Bayesian game formulation, the crowdsourcing system consists of n workers $w_1, \ldots, w_n$. Each worker has the same set of actions $\mathcal{E}$ and can be of any type in $\mathcal{M}$. Recall that $a_i \in \mathcal{E}$ 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 worker w_i's knowledges of the types of other workers θ_{-i} is $\Pr[\theta_{-i}|\theta_i]$, where $\theta_{-i} = [\theta_j]_{j \neq i}$ denotes a vector of types for all workers except w_i. Since $\theta_1, \ldots, \theta_n$ are IID random variables, with pmf given in Eq. (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 $k \in \mathcal{E}$ with a specific 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 a level k effort, with $p(k) \geq 0, \sum_{k=0}^{K} p(k) = 1$. Define $\boldsymbol{\Sigma}$ as the action space for a worker, which is a set of all possible mixed actions:

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

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

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

Definition III.1. *Worker w_i's strategy is a map $\mathbf{s}_i : \mathcal{M} \to \boldsymbol{\Sigma}$ prescribing an action for each of his possible skill type.*

For example, $\mathbf{s}_i(1) = (\frac{1}{K+1}, \ldots, \frac{1}{K+1})$ 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

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

where $[\mathbf{s}_j(\theta_j)]_{a_j}$ represents the probability that w_j plays $a_j \in \mathcal{E}$ 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 III.2. *$(\mathbf{s}_1^*(\cdot), \ldots, \mathbf{s}_n^*(\cdot))$ is a Bayesian Nash equilibrium if for all $i = 1, ..., n$, and for all $\theta_i \in \mathcal{M}$, we have $\mathbf{s}_i^*(\theta_i) \in \arg\max_{\mathbf{s}_i(\theta_i) \in \boldsymbol{\Sigma}} u_i(\mathbf{s}_i(\theta_i), \mathbf{s}_{-i}^*(\cdot), \theta_i)$.*

B. Deriving the Minimum Reward

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 the 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 freeride and never exerts any effort.

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.

Lemma III.1. *Consider our proposed incentive mechanism, and assume two workers work on a task. Given a skill type m, if $r > \frac{2c_{m,K}}{\beta_m}$, then at least one Bayesian Nash equilibrium exists, and 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: Please refer to [18] for the proof. ∎

Remark: Lemma III.1 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 the 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 III.1 to the case of $n \geq 2$.

Lemma III.2. *Consider our incentive mechanism and assume $n \geq 2$ workers work on a task. Given a skill type m, if*

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

then at least one Bayesian Nash equilibrium exists, and 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: Please refer to [18] for the proof. ∎

Remark. This is a *strong result* because it specifies the minimum reward needed to guarantee that workers exert their maximum (level K) efforts.

Requesters are only interested in high quality solutions, i.e., 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 at least one such high quality solution can be collected with high probability.

Theorem III.1. *To guarantee 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 III.2. ∎

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 $n' = 31$ workers to work on a task. The corresponding reward is at least $6.25c_{M,K}$. Note that as we increase the value of β_M, the minimum number of workers needed decreases as well as the minimum reward. This suggests that a crowdsourcing website needs to implement an accurate task assignment algorithm in order to reduce the reward a requester pays so as to attract more requesters.

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 I. MINIMUM n' AND MINIMUM REWARD r TO HAVE A HIGH QUALITY SOLUTION WITH PROBABILITY OF 0.999.

Summary. Our incentive mechanism guarantees that workers exert their maximum (level K) efforts. However, there is a non-zero probability $(1-\beta_M)^n$ that no worker has skill type M. Although these lower skill workers 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.

IV. REPUTATION SYSTEM DESIGN

We now present a reputation system to guarantee that low-skilled workers refuse the task so that only high-skilled workers participate. 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 once 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 IV.1. *s^* is a desirable strategy prescribing that a skilled worker (type M) provides a solution using his maximum (level K) effort, while less skilled workers (skill type $<M$) refuse the task.*

Definition IV.2. *An "error" occurs when a worker submits a solution whose quality is lower than $Q(M,K)$.*

Our reputation system relies on requesters to rate workers using a *feedback rating* $\in\{0,1\}$, where 0 indicates an error, and 1 indicates 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 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 $\overline{N}_e$, a punishment to this worker will be triggered which blocks this worker from participating in crowdsourcing activities for a given number, $\overline{N}_b$, of time slots. We refer to $\overline{N}_e$ and $\overline{N}_b$ as the *active window size* and the *blocking window size* 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 $\overline{N}_b$ time slots, our reputation system activates the worker's account and resets N_e to zero. We denote this system as the $(\overline{N}_e,\overline{N}_b)$-*reputation system*. Formally, we index a worker's reputation via a pair (N_e,N_b), where $N_b=\overline{N}_b$ indicates that a worker is active, and $N_e=\overline{N}_e$ indicates that a worker is blocked. Our reputation system is described by the following transition rules,

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

A. 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 that a low-skilled worker refusing a task, leading to this worker being idle for the time slot. We integrate the $(\overline{N}_e,\overline{N}_b)$–reputation system into our proposed incentive mechanism. Recall from Section II, 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, 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, the long term discounted utility for worker w_i is $u_i^\infty(\{\mathbf{s}_i^t(\theta_i^t)\}_{t=0}^\infty) = \sum_{t=0}^\infty \delta^t(1 - \mathbf{I}_{\{\text{idle}\}}^t)u_i(\mathbf{s}_i^t(\theta_i^t), \mathbf{s}_{-i}^t(\cdot), \theta_i^t)$, 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.

B. 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 size $\overline{N}_e$, blocked window size $\overline{N}_b$, and β_M, on this minimum reward.

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

$$r > n\, c_{M,K} \max\left\{\frac{1}{\gamma^*}, \left(1 - \frac{n(1-\beta_M)^{n-1}(1-\delta)}{(1-\delta^{\overline{N}_b})\beta_M\delta}\right)^{-1}\right\}, \quad (2)$$

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

Proof: Please refer to [18] for the proof. ∎

Remark. One needs to set the active window size to $\overline{N}_e = 1$ because workers committing errors at state $(0, \overline{N}_b), (1, \overline{N}_b), ..., (\overline{N}_e - 2, \overline{N}_b)$ incur no penalties.

V. 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, i.e., some critical requesters may always express low ratings, while lenient requesters may always express high ratings. Such factors may lead to the following situation: a solution of quality $Q(M, K)$ may receive a feedback rating of zero, or a solution of quality lower than $Q(M, K)$ may receive a feedback rating of one. We refer to these as *erroneous ratings*. We assume that a solution of quality $Q(m, k)$, where $m \in \mathcal{M}, k \in \mathcal{E}$, 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 an erroneous rating occurs with a small probability, i.e., $\epsilon_{M,K} << 1$, and $\epsilon_{m,k} \approx 1$ holds for all m, k such that $Q(m,k) < Q(M,K)$. We also assume $\epsilon_{m,k} > \epsilon_{m',k'}$ 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 $K+1$ variation matrix, $\boldsymbol{\epsilon} = [\epsilon_{m,k}]$. We assume that only the highest quality solutions among the n solutions submitted for a task have a chance to be selected as winners by a biased requester, regardless of whatever he is biased toward expressing higher or lower feedback ratings.

A. Quantify the Impact of Human Factors

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

Theorem V.1. *Consider a combined incentive/$(\overline{N}_e, \overline{N}_b)$-reputation system and assume $n \geq 2$ workers work on a task. The feasible space of $(\overline{N}_e, \overline{N}_b)$ is*

$$\overline{N}_e \leq \frac{-\ln\left(n\epsilon_{M,K}/((1-\beta_M)^{1-n} - 1 + \epsilon_{M-1,K})\right)}{\ln(1 - \delta + \delta\beta_M\epsilon_{M,K}) - \ln(\delta\beta_M\epsilon_{M,K})}, \quad (3)$$

$$\overline{N}_b \geq \ln\frac{1 - \frac{n\epsilon_{M,K}}{(1-\beta_M)^{1-n} - 1 + \epsilon_{M-1,K}}\left(\frac{1-\delta+\delta\beta_M\epsilon_{M,K}}{\delta\beta_M\epsilon_{M,K}}\right)^{\overline{N}_e}}{1 - n\epsilon_{M,K}/((1-\beta_M)^{1-n} - 1 + \epsilon_{M-1,K})} / \ln\delta \quad (4)$$

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

$$r > nc_{M,K} \max\left\{\frac{1}{\gamma^*}, \left(1 - \left((1 + (1-\delta)/(\delta\beta_M\epsilon_{M,K}))^{\overline{N}_e} - \delta^{\overline{N}_b}\right)\frac{(1-\beta_M)^{n-1}n\epsilon_{M,K}}{(1-\delta^{\overline{N}_b})(1-(1-\epsilon_{M-1,K})(1-\beta_M)^{n-1})}\right)^{-1}\right\}, \quad (5)$$

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

Proof: Please refer to [18] for the proof. ∎

Table II presents the minimum reward derived from Eq. (5), when $n = 3$, $\epsilon_{M,K} = 0.05$, $\epsilon_{M-1,K} = 0.95$, $\beta_M = 0.6$, $\delta = 0.999$. In order to apply Theorem V.1, we take $\overline{N}_e \leq 114$. Table II depicts the active window size $\overline{N}_e$, blocking window size $\overline{N}_b$, and the corresponding minimum reward. When $\overline{N}_b = 100$, as the active window size increases from $\overline{N}_e = 10$ to $\overline{N}_e = 30$, the reward increases from $3.42c_{M,K}$ to $5.46c_{M,K}$, a significant increase. When $\overline{N}_e = 20$, an increase in the blocking window size from $\overline{N}_b = 70$ to $\overline{N}_b = 90$, results in a slight decrease in the reward from $4.66c_{M,K}$ to $4.20c_{M,K}$. We next model our combined incentive reputation system as a Markov chain and show that the number of active workers increases in $\overline{N}_e$ and decreases in $\overline{N}_b$.

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

TABLE II. Minimum reward to guarantee desirable strategy s^*, with $\epsilon_{M,K} = 0.05$, $\epsilon_{M-1,K} = 0.95$, $\beta_M = 0.6$, $\delta = 0.999$, $n = 3$.

We formulate a Markov chain to characterize the evolving dynamics of a worker's reputation (when he plays s^*).

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 $\{(\overline{N}_e, N_b), N_b = 0, 1, \ldots, \overline{N}_b - 1\} \cup \{(N_e, \overline{N}_b), N_e = 0, 1, \ldots, \overline{N}_e - 1\}$. Applying variation matrix $\boldsymbol{\epsilon}$, the one-step state transition probability of our Markov chain can be expressed as:

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

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

Theorem V.2. *Denote π_{N_e,N_b} the stationary probability of a worker in state (N_e, N_b). We have $\pi_{N_e,\overline{N}_b} = 1/(\overline{N}_e + \overline{N}_b \beta_M \epsilon_{M,K})$, $N_e = 0, 1, \ldots, \overline{N}_e - 1$, and $\pi_{\overline{N}_e,N_b} = \beta_M \epsilon_{M,K}/(\overline{N}_e + \overline{N}_b \beta_M \epsilon_{M,K})$, $N_b = 0, 1, \ldots, \overline{N}_b - 1$.*

Proof: One can prove this by substituting the expressions for π_{N_e,N_b} into $\boldsymbol{\pi} = \boldsymbol{\pi}\mathbf{P}$ where $\mathbf{P} = [\Pr\left[(N'_e, N'_b)|(N_e, N_b)\right]]$. ∎

Remark: The probability that a worker is active is $\overline{N}_e/(\overline{N}_e + \overline{N}_b \beta_M \epsilon_{M,K})$, which increases in $\overline{N}_e$ and decreases in $\overline{N}_b$. Thus either increasing $\overline{N}_e$ or decreasing $\overline{N}_b$ increases the total number of active workers. However, this has the side effect of increasing the minimum reward (Theorem V.1).

VI. Related work

Research on crowdsourcing has been very active recently, and it ranges from application design [5], cheat detection [6], quality management [10], incentive design [12], etc. Much work focus determining the minimum reward so as to attract users (workers and requesters). A variety of auction based pricing mechanisms were proposed [3], [13]. However, 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, authors in [3], [13] only modeled the interaction between workers and requesters as a single shot game, while our work further extends it to a repeated game to represent behavior of long-lived workers.

Several works were done to design incentive protocols so that workers provide their maximum effort. Only a few works [11], [17] have investigated this important question and usually their models are simple (e.g, single-shot game). We focus on a general crowdsourcing application scenario and we formulate a repeated game to understand the strategic behavior of long-lived workers. Recently, a few reputation protocols [15], [16], [20] have been proposed. Our work considers a more general scenario than theirs. Specifically, [15], [20] assumed that all workers are high-skilled (one skill type), only two level of efforts, and each task is solved by only one worker. Furthermore, their works missed an important element of task assigning. We incorporate this element by presenting a probabilistic model for the task assignment process. Authors in [16] only considered two types of workers, i.e., experts or novice, two levels of efforts, i.e., maximum effort or zero effort, and they did not consider errors in reputation updating.

VII. Conclusion

This paper presents a unified study of incentive and reputation mechanisms for online crowdsourcing systems. Our incentive mechanism allows multiple workers to solve a task, splits the reward among workers based on requester evaluations of the solution quality. We derived the minimum reward needed to guarantee that workers provide their maximum effort. 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.

Acknowledgment

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

References

[1] Amazon Mechanical Turk. https://www.mturk.com.

[2] Clickworker. http://www.clickworker.com/.

[3] D. DiPalantino and M. Vojnovic. Crowdsourcing and all-pay auctions. In *Proc. of ACM EC*, 2009.

[4] Google Helpouts. https://helpouts.google.com/home.

[5] J. Heer and M. Bostock. Crowdsourcing graphical perception: Using mechanical turk to assess visualization design. In *ACM CHI*, 2010.

[6] M. Hirth, T. Hossfeld, and P. Tran-Gia. Cost-optimal validation mechanisms and cheat-detection for crowdsourcing platforms. In *Proc. of IMIS*, 2011.

[7] J. Horton. Online labor markets. *Internet and Network Economics*, pages 515–522, 2010.

[8] J. Howe. The rise of crowdsourcing. *Wired magazine*, 14(6):1–4, 2006.

[9] P. G. Ipeirotis. Analyzing the amazon mechanical turk marketplace. *XRDS*, 17(2):16–21, Dec. 2010.

[10] P. G. Ipeirotis, F. Provost, and J. Wang. Quality management on amazon mechanical turk. In *Proc. of HCOMP*, 2010.

[11] S. Jain, Y. Chen, and D. C. Parkes. Designing incentives for online question and answer forums. In *Proc. of ACM EC*, 2009.

[12] W. Mason and D. J. Watts. Financial incentives and the "performance of crowds". *SIGKDD Explor. Newsl.*, 11(2):100–108, May 2010.

[13] Y. Singer and M. Mittal. Pricing mechanisms for crowdsourcing markets. In *Proc. of WWW*, 2013.

[14] Threadles. http://www.threadless.com/.

[15] Y. Xiao, Y. Zhang, and M. van der Schaar. Socially-optimal design of crowdsourcing platforms with reputation update errors. In *Proc. of IEEE ICASSP*, 2013.

[16] H. Xie, John C.S. Lui, J. W. Jiang, and W. Chen. Incentive mechanism and protocol design for crowdsourcing systems. In *Proc. of IEEE Allerton*, 2014.

[17] H. Xie, John C.S. Lui, and W. Jiang. Mathematical modeling of crowdsourcing systems: Incentive mechanism and rating system design. In *Proc. of IEEE MASCOTS*, 2014.

[18] H. Xie, John C.S. Lui, and D. Towsley. *Incentive and Reputation Mechanisms for Online Crowdsourcing Systems. http://www.cse.cuhk.edu.hk/%7Ehxie/TRCS.pdf.*

[19] Yahoo! Answers. http://answers.yahoo.com.

[20] Y. Zhang and M. van der Schaar. Reputation-based incentive protocols in crowdsourcing applications. In *Proc. of IEEE INFOCOM*, 2012.