SOCIALLY-OPTIMAL DESIGN OF CROWDSOURCING PLATFORMS WITH REPUTATION UPDATE ERRORS

Yuanzhang Xiao, Yu Zhang, and Mihaela van der Schaar

Department of Electrical Engineering, UCLA. Email: yxiao@ee.ucla.edu

ABSTRACT

Crowdsourcing systems (e.g. Yahoo! Answers and Amazon Mechanical Turk) provide a platform for requesters, who have tasks to solve, to ask for help from workers. Vital to the proliferation of crowdsourcing systems is incentivizing the workers to exert high effort to provide high-quality services. Reputation mechanisms have been shown to work effectively as incentive schemes in crowdsourcing systems. A reputation agency updates the reputations of the workers based on the requesters' reports on the quality of the workers' services. A low-reputation worker is less likely to get served when it requests help, which provides incentives for the workers to obtain a high reputation by exerting high effort. However, reputation update errors are inevitable, because of either system errors such as loss of reports, or inaccurate reports, resulting from the difficulty in accurately assessing the quality of a worker's service. The reputation update error prevents existing reputation mechanisms from achieving the social optimum. In this paper, we propose a simple binary reputation mechanism, which has only two reputation labels ("good" and "bad"). To the best of our knowledge, our proposed reputation mechanism is the first that is proven to be able to achieve the social optimum even in the presence of reputation update errors. We provide design guidelines for socially-optimal binary reputation mechanisms.

Index Terms— crowdsourcing, reputation, game theory

1. INTRODUCTION

Crowdsourcing systems, such as Yahoo! Answers and Amazon Mechanical Turk, provide platforms for a user to elicit collective efforts from the other users to solve its task. In a typical crowdsourcing system, a user can either post tasks and request for help as a requester, or solve the tasks posted by others as a worker. In some crowdsourcing systems [1][2], the workers are rewarded by monetary payments from the requesters. The payment is usually paid when the worker is assigned with the task, instead of after the worker completes the task. This creates an incentive problem, namely the worker may want to exert low effort on the task since it has already been paid. In other systems [3], the servers are rewarded by the benefit obtained from other users' services. In such systems without monetary payment, it is even more difficult to provide the servers with the incentive to exert high efforts. In summary, the incentive provision for the servers is vital to the effectiveness of crowdsourcing systems.

One effective incentive mechanism is the reputation mechanism [1]–[3]. In a reputation mechanism, there is a reputation agency, who assigns reputations to all the users based on their behaviors. If a user exerts high efforts in the past as a server, it will receives a high reputation as a summary of its good behaviors in the past. Similarly, a user's low reputation indicates that it used to exert low efforts as a server. Since a low-reputation user may get its task served with low effort, the users have incentives to get high reputation by exerting high efforts to solve the others' tasks. Built on this intuition, the existing reputation mechanisms work well, except when there are reputation update errors. The reputation update error comes from two sources. First, a client cannot perfectly assess the service quality of a server, and hence, will report the service quality as low to the reputation agency, even when the server actually exert high effort. Second, the reputation agency may miss the client's report on the service quality, or update the reputation in a wrong way by mistake. In the presence of reputation update errors, the existing reputation mechanisms [1]–[3] have performance loss that is increasing with the update error probability. This performance loss in [1]–[3] partially comes from the restriction of attention on stationary Markov strategies. As we will see later, once we remove this restriction, the social optimum can be achieved.

In this paper, we show that the social optimum can be achieved in the presence of reputation update errors, if we do not restrict our attention to stationary Markov strategies as in [1]–[3]. In other words, we allow the users to take different actions given the same current state at different time instants, which increases the strategy space to be considered significantly and potentially complicates the optimal strategy. Nevertheless, we rigorously prove that under certain conditions, the social optimum can be achieved by a class of simple reputation mechanisms, namely binary reputation mechanisms that assign binary reputation labels to the users. We derive the conditions under which the social optimum can be achieved, which provide guidelines for the design of reputation update rules. In addition, we further simplify the optimal strategy by proving that strategies of a simple structure can be optimal, and propose an algorithm to construct the optimal strategy.

The rest of the paper is organized as follows. We discuss prior work in Section 2. In Section 3, we describe the model of crowdsourcing systems. Then we design the optimal reputation mechanisms in Section 5. Simulation results in Section 6 demonstrate the performance improvement of the proposed reputation mechanism. Finally, Section 7 concludes the paper.

2. RELATION TO PRIOR WORK

The idea of social norm and reputation mechanism was originally proposed in [4], assuming that there is no reputation update error. The performance loss of the reputation mechanism in [4] is large under reputation update errors. Based on the idea in [4], [5][6][7] did some experiments on different reputation mechanisms. Their experiment results show that some simple reputation mechanisms have performance loss under reputation update errors.

[1]–[3][8] rigorously analyzed reputation mechanisms under

We acknowledge NSF (grant 0830556) for funding this research.

Table 1. The gift-giving game between a requester and a worker.

	high effort	low effort
request	$(b, -c)$	$(0, 0)$

reputation update errors. They designed optimal reputation mechanisms with stationary Markov strategies, and quantifies the performance loss of such reputation mechanisms under reputation update errors. We will show that the performance loss can be eliminated by using reputation mechanisms with nonstationary strategies.

Our paper is the first one that proposes socially optimum reputation mechanisms under reputation update errors. Although we model the system as a stochastic game, our results are different from the folk theorem-type results in repeated games [9] and in stochastic games [10]. In a practical crowdsourcing platform, the individual reputations of all the users cannot and are not allowed (due to privacy considerations) to be known to the users. However, the results in [9][10] are derived without this informational constraint, and hence, cannot be applied in our setting.

3. SYSTEM MODEL

Consider a crowdsourcing platform with a set of N users, denoted by $\mathcal{N} = \{1, \ldots, N\}$. We assume that the number of users N is displayed on the platform and is known to all the users. Each user has plenty of tasks to solve, and possesses some resources valuable to the other users' tasks. Since the users usually stay in the platform for a long period of time, we divide time into periods labeled by $t = 0, 1, 2, \ldots$. In each period t, each user, as a *requester*, first requests help to solve its tasks. Then, each user, as a *worker*, is matched to another user's task, and chooses to solve the task with a high or low effort level. A matching is define as a bijective mapping $m : \mathcal{N} \to \mathcal{N}$, where user i, as a worker, is matched to the task requested by user $m(i)$. Since a user cannot be matched to itself, we denote the set of all possible matchings as $M = \{m : m \text{ bijective}, m(i) \neq i, \forall i \in \mathcal{N}\}$. A matching rule is then defined as a probability distribution μ on the set of all possible matchings M. In this paper, we focus on the uniformly random matching rule, which satisfies $\mu(m) = 1/\left(N! \sum_{i=0}^{N} \frac{(-1)^i}{i!}\right), \forall m \in M$.

Once a requester and a worker are matched, they play the gift-giving game in Table 1, where the row player is the requester and the column player is the worker. We assume that there is no cost for requesting for service. Hence, each user requests service in every period. We normalize the payoffs received by the requester and the worker when a worker exerts low effort to be zero. When a worker exerts high effort, the requester gets a benefit of $b > 0$ and the worker incurs a cost $c \in (0, b)$. We assume that the users are homogeneous in terms of benefits and costs.

We can see that in the unique Nash equilibrium of the gift-giving game, the worker will exert low effort, which results in a zero payoff for both the requester and the worker. We are interested in the scenarios where $b > c$, namely exerting high effort leads to the social optimum. Our goal is to design an incentive scheme such that it is in their self-interests for the workers to exert high effort.

A powerful incentive scheme is the reputation mechanism, which exploits the repeated interaction of the requesters and the workers, and assigns each user a reputation as the summary of its past behavior. In this paper, we focus on the simplest reputation mechanisms, namely *binary* reputation mechanisms, and will show that binary reputation mechanisms can achieve social optimum. A binary reputation mechanism assigns binary reputations to the users. Denote the set of binary reputations by $\Theta = \{0, 1\}$, and the reputation of user i by θ_i, $\forall i \in \mathcal{N}$. The *reputation profile* is defined as the vector of all the users' reputations, $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_N)$. The reputation profile contains the users' private information and is not known to the users. However, an important statistics, namely the *reputation distribution*, can be listed on the platform and observed by all the users. A reputation distribution is defined as a tuple $\boldsymbol{s}(\boldsymbol{\theta}) = (s_0(\boldsymbol{\theta}), s_1(\boldsymbol{\theta}))$, where $s_1(\boldsymbol{\theta}) = \sum_{i \in \mathcal{N}} \theta_i$ is the number of users with reputation 1, and $s_0(\boldsymbol{\theta}) = \sum_{i \in \mathcal{N}} (1 - \theta_i)$ is the number of users with reputation 0. Note, however, that when a worker is matched to a requester, it is informed by the platform of its requester's reputation.

As described before, a user always request for service as a requester because there is no cost to request. Hence, a user only needs to determine its effort level as a worker. Consequently, we model each user's action as a *contingent plan* of exerting high or low effort based on its own reputation and the reputation of the requester matched to it. Formally, each user i's action, denoted by α_i, is a mapping $\alpha_i : \Theta \times \Theta \to Z$, where $Z = \{0, 1\}$ is the set of effort levels with $z = 0$ representing "low effort". Then $\alpha_i(\theta_{m(i)}, \theta_i)$ denotes user i's effort level as a worker when it is matched to a requester with reputation $\theta_{m(i)}$. We write the action set as $A = \{\alpha | \alpha : \Theta \times \Theta \to Z\}$, and the joint action profile of all the users as $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_N)$.

After each worker exerts effort based on its action, the requester reports its assessment of the effort level to the platform. The report is defined as a mapping $R : Z \to \Delta(Z)$, where $\Delta(Z)$ is the probability distribution over Z. For example, $R(1|z)$ is the probability that the requester reports "high effort" given the worker's actual effort level z. An example report could be

$$R(z'|z) = \begin{cases} 1 - \varepsilon, & z' = z \\ \varepsilon, & z' \neq z \end{cases}, \tag{1}$$

where $\varepsilon \in [0, 0.5)$ is the report error probability. This report error may be caused by the requester's inaccurate assessment of the effort level and by the system error of the platform (e.g. loss of reports). Since the users are homogeneous, we assume the same report mapping R for all the users.

Based on the requester's report, the platform will update the worker's reputation based on the *reputation update rule*. Again, since the users are homogeneous, we assume that the reputation update rule is the same for the users. Then the reputation update rule, denoted by τ, is defined as a mapping $\tau : \Theta \times \Theta \times Z \to \Delta(\Theta)$. For example, $\tau(\theta'_w | \theta_r, \theta_w, z)$ is the probability distribution of the worker's updated reputation θ'_w, given the reputation of the requester θ_r, the worker's own reputation θ_w, and the requester's report z. We focus on a class of reputation update rules defined as

$$\tau(\theta'_w | \theta_r, \theta_w, z) = \begin{cases} \beta^+_{\theta_w}, & \theta'_w = 1, z \geq \alpha_0(\theta_r, \theta_w) \\ 1 - \beta^+_{\theta_w}, & \theta'_w = 0, z \geq \alpha_0(\theta_r, \theta_w) \\ 1 - \beta^-_{\theta_w}, & \theta'_w = 1, z < \alpha_0(\theta_r, \theta_w) \\ \beta^-_{\theta_w}, & \theta'_w = 0, z < \alpha_0(\theta_r, \theta_w) \end{cases},$$

where α_0 is the platform's *recommended action*, based on which the reputation is updated. In the above reputation update rule, if the reported effort level is not lower than the one specified by the recommended action, a worker with reputation θ_w will have reputation 1 with probability $\beta^+_{\theta_w}$; otherwise, it will have reputation 0 with probability $\beta^-_{\theta_w}$.

4. STOCHASTIC GAME FORMULATION

In the section, we first formulate the repeated interaction among the users in the crowdsourcing platform with reputation mechanisms as a stochastic game. Then we make some restrictions on the users' strategies based on the users' knowledge of the system, and define the corresponding equilibrium, considering the above restrictions. Finally, we define the platform designer's problem.

A stochastic game is described by the set of players, the set of states with the state transition probability, the set of actions, and the players' stage-game payoff functions. We consider the platform as a player, which is indexed by 0, and define the set of players as $\{0\} \cup \mathcal{N}$. The state is defined as the reputation profile $\boldsymbol{\theta}$, and the set of states as Θ^N. The platform's action is the recommended action, which is defined in the same way as the users' actions, namely $\alpha_0 : \Theta \times \Theta \to \{0, 1\}$. We write the state transition probability as $q(\boldsymbol{\theta}'|\boldsymbol{\theta}, \alpha_0, \boldsymbol{\alpha})$, which is the probability that the next state is $\boldsymbol{\theta}'$ given the current state $\boldsymbol{\theta}$, the platform's recommended action, and the joint action profile $\boldsymbol{\alpha}$. Note that the state transition probability q is determined by the matching rule μ, the report function R, and the reputation update rule τ. Finally, each user i's stage-game payoff function is $u_i(\boldsymbol{\theta}, \alpha_0, \boldsymbol{\alpha})$, which depends on[1] the state $\boldsymbol{\theta}$ and the joint action profile $\boldsymbol{\alpha}$. We define the platform's payoff as a constant $u_0(\boldsymbol{\theta}, \alpha_0, \boldsymbol{\alpha}) = 0, \forall \boldsymbol{\theta}, \alpha_0, \boldsymbol{\alpha}$, such that it will follow the platform designer's decisions on how to choose the recommended action. Note that the platform designer's goal (or payoff) is the social welfare, which will be defined later.

The users interact in the platform repeatedly. Hence, each user should have a *strategy*, which is a contingent plan of which action to take based on the history. The history is the collection of the past and the current states (i.e. reputation profiles). Denote the history at period t as $\boldsymbol{h}^t = (\boldsymbol{\theta}^0, \ldots, \boldsymbol{\theta}^t)$, where $\boldsymbol{\theta}^t$ is the reputation profile at the beginning of period t, and the set of public histories at period t as $\mathcal{H}^t = (\Theta^N)^{t+1}$. Then each user i's strategy $\pi_i : \cup_{t=0}^{\infty} \mathcal{H}^t \to A$ as a mapping from the set of all possible histories to the action set. Similarly, we define the platform's recommended strategy as $\pi_0 : \cup_{t=0}^{\infty} \mathcal{H}^t \to A$. The joint strategy profile of all the users is written as $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_N)$. We write the set of all strategies and the set of all strategy profiles as Π and Π^N, respectively. Given the matching rule μ, the initial reputation profile $\boldsymbol{\theta}^0$, the report function R, and the reputation mechanism (Θ, τ), the recommended strategy π_0 and the joint strategy profile $\boldsymbol{\pi}$ induce a probability distribution over the set of all the histories $\mathcal{H}^\infty$. Taking the expectation with respect to this probability distribution, each user i receives a discounted average payoff $U_i(\boldsymbol{\theta}^0, \pi_0, \boldsymbol{\pi})$ defined as

$$U_i(\boldsymbol{\theta}^0, \pi_0, \boldsymbol{\pi}) = \mathbb{E}_{\boldsymbol{h}^\infty} \left\{ (1-\delta) \sum_{t=0}^{\infty} \delta^t u_i(\boldsymbol{\theta}^t, \pi_0(\boldsymbol{h}^t), \pi(\boldsymbol{h}^t)) \right\},$$

where $\delta \in [0, 1)$ is the common discount factor of all the users. The discount factor δ is the rate at which the users discount future payoffs, and reflects the patience of the users. A more patient user has a larger discount factor.

Now we make some restrictions on the strategies. First, since the users are homogeneous, we assume that all the users adopt the same strategy, namely $\pi_i = \pi, \forall i \in \mathcal{N}$. In particular, we write the symmetric joint strategy profile as $\pi \cdot \mathbf{1}_N$, where $\mathbf{1}_N$ is a vector of 1's with length N. Note that although all the users adopt the same action in each period, they will choose different effort levels since both their own and their requesters' reputations are different.

Second, since the users know the reputation distributions only, but not the reputation profiles, we restrict the users' strategies to be indifferent in reputation profiles that have the same reputation distribution. We call such strategies *informationally-plausible (IP) strategies*, since it is infeasible for the users to adopt strategies that require the knowledge of reputation profiles. Formally, we define the *IP strategies* as follows.

Definition 1 (IP Strategies) *A strategy π is informationally-plausible, if for all $t \geq 0$ and for all $\boldsymbol{h}^t, \tilde{\boldsymbol{h}}^t \in \mathcal{H}^t$, we have*

$$\pi(\boldsymbol{h}^t) = \pi(\tilde{\boldsymbol{h}}^t), \text{ if } \boldsymbol{s}(\boldsymbol{\theta}^k) = \boldsymbol{s}(\tilde{\boldsymbol{\theta}}^k),\ k = 0, 1, \ldots, t. \tag{2}$$

We write the set of all IP strategies as Π_f.

In this paper, we focus on symmetric IP strategy profiles.

Finally, since the action set A has $2^4 = 16$ elements, the complexity of choosing the action is large for the users. Hence, we consider the strategies that choose actions from a subset $B \in A$, and define $\Pi_f(B)$ as the set of symmetric IP strategies restricted on the subset of actions B. We are particularly interested in two subsets of actions, $A^{\text{afs}} \triangleq \{\alpha^{\text{a}}, \alpha^{\text{f}}, \alpha^{\text{s}}\}$ with three actions. Specifically, the action α^{a} is the *altruistic* action, define as $\alpha^{\text{a}}(\theta_r, \theta_w) = 1, \forall \theta_r, \theta_w \in \{0, 1\}$, where the worker exerts high effort regardless of the worker's and the requester's reputations. The action α^{f} is the *fair* action, define as $\alpha^{\text{f}}(\theta_r, \theta_w) = 0$ if $\theta_w > \theta_r$ and $\alpha^{\text{f}}(\theta_r, \theta_w) = 1$ if $\theta_w \leq \theta_r$, where the worker exerts high effort only when the requester has a higher or equal reputation. The action α^{s} is the *selfish* action, define as $\alpha^{\text{s}}(\theta_r, \theta_w) = 0, \forall \theta_r, \theta_w \in \{0, 1\}$, where the worker exerts low effort regardless of the worker's and the requester's reputations.

As we will show later, a pair of a recommended strategy and a strategy profile $(\pi_0, \pi \cdot \mathbf{1}_N) \in \Pi_f(A^{\text{afs}}) \times \Pi_f^N(A^{\text{afs}})$ can achieve social optimum at the equilibrium if we design reputation update rules carefully. The equilibrium is defined as follows.

Definition 2 (Nash equilibrium) *A pair of a IP recommended strategy and a symmetric IP strategy profile $(\pi_0, \pi \cdot \mathbf{1}_N) \in \Pi_f \times \Pi_f^N$ is a Nash equilibrium (NE), if for all $\boldsymbol{\theta}^0 \in \Theta^N$ and for all $i \in \mathcal{N}$,*

$$U_i(\boldsymbol{\theta}^0, \pi_0, \pi \cdot \mathbf{1}_N) \geq U_i(\boldsymbol{\theta}^0, \pi_0, (\pi_i, \pi \cdot \mathbf{1}_{N-1})),\ \forall \pi_i \in \Pi,$$

where $(\pi_i, \pi \cdot \mathbf{1}_{N-1})$ is the strategy profile in which user i deviates to π_i and the other users follow the strategy π.

The goal of the platform designer is to maximize the social welfare at the equilibrium in the worst case (with respect to different initial reputation distributions), which provides a much stronger performance guarantee than maximizing the performance given an initial reputation distribution and than maximizing the expected performance. The platform design problem can be formulated as:

Platform Design Problem :

$$\begin{array}{ll} \max_{\tau, (\pi_0, \pi \cdot \mathbf{1}_N) \in \Pi_f \times \Pi_f^N} \min_{\boldsymbol{\theta}^0 \in \Theta^N} & \frac{1}{N} \sum_{i \in \mathcal{N}} U_i(\boldsymbol{\theta}^0, \pi_0, \pi \cdot \mathbf{1}_N) \\ s.t. & (\pi_0, \pi \cdot \mathbf{1}_N) \text{ is a NE.} \end{array}$$

5. SOCIALLY OPTIMAL DESIGN

The social optimum is $b - c$, achieved by the workers exerting high efforts all the time, which is not an equilibrium strategy. In this section, we will show that under properly designed reputation mechanisms, the social optimum $b - c$ can be asymptotically achieved at the NE in the following sense.

[1]Note that, although the stage-game payoff does not depend on the recommended action, we write the recommended action α_0 as an argument of the payoff function by the convention of game theory. However, the recommended action does affect the long-term payoff through the state transition probability.

Definition 3 (Asymptotically Optimal Reputation Mechanisms) *We say a reputation mechanism* $(\tau, \pi_0(\xi,\delta) \in \Pi_f(A^{\text{afs}}))$ *is asymptotically optimal, if for any small* $\xi > 0$*, we can find* $\underline{\delta}$*, such that for all discount factor* $\delta > \underline{\delta}$*,* $(\pi_0(\xi,\delta), \pi_0(\xi,\delta)\cdot \mathbf{1}_N)$ *is a NE and guarantees* $U_i(\boldsymbol{\theta}^0, \pi_0, \pi_0\cdot\mathbf{1}_N) \geq b-c-\xi,\ \forall i \in \mathcal{N}$*, starting from any initial reputation profile* $\boldsymbol{\theta}^0$*.*

Note that in the optimal reputation mechanism, the reputation update rule can be independent of the tolerated performance loss ξ and the discount factor δ, while the recommended strategy depends on both ξ and δ. Note also that in the above definition, we require the user's strategy to be the same as the recommended strategy. Hence, the platform can announce the recommended action in each period, such that the users can follow the recommended action. In this way, the users do not need to calculate the action to take in each period by themselves.

Theorem For any reputation update error $\varepsilon \in [0, 0.5)$, we can design an asymptotically optimal reputation mechanism. We should design it such that the following conditions are satisfied:

- Condition 1: $\beta_1^+ > 1-\beta_1^-$ and $\beta_0^+ > 1-\beta_0^-$
- Condition 2: $x_1^+ \triangleq (1-\varepsilon)\beta_1^+ + \varepsilon(1-\beta_1^-) > \frac{1}{1+\frac{c}{(N-1)b}}$;
- Condition 3: $x_0^+ \triangleq (1-\varepsilon)\beta_0^+ + \varepsilon(1-\beta_0^-) < \frac{1-\beta_1^+}{\frac{c}{(N-1)b}}$;

Proof: See [13, Appendix A]. □

The theorem proves that for any reputation update error $\varepsilon \in [0, 0.5)$, we can design an asymptotically optimal reputation mechanism. It also shows us how to design it. First, we should give incentive for the users to exert high effort, by setting β_θ^+, the probability that the reputation goes up when the effort level is not lower than the recommended one, to be larger than $1-\beta_\theta^-$, which is the probability that the reputation goes up when the effort level is lower than the recommended one. Second, for the users with reputation 1, the expected probability that the reputation goes up when the effort level is not lower than the recommended one, x_1^+, should be larger than the threshold specified in Condition 2. Meanwhile, for the users with reputation 0, the expected probability that the reputation goes up when the effort level is not lower than the recommended one, x_0^+, should be smaller than the threshold specified in Condition 2. Note that Conditions 2 and 3 imply that $x_1^+ > x_0^+$. In this way, a user will prefer to have reputation 1.

Given ξ and $\delta > \underline{\delta}$, we can construct the optimal recommended strategy $\pi_0(\xi,\delta)$. The optimal equilibrium strategy is not stationary, in that given the same state, the actions taken can be different at different periods. Due to space limitation, we refer interested readers to [13, Appendix B] for the algorithm of constructing the recommended strategy. An overview of the optimal recommended strategy is as follows. If all the users have reputation 1 or 0, the altruistic action α^{a} or the selfish action α^{s} is recommended, respectively. If the users have different reputations, the altruistic action α^{a} or the fair action α^{f} is used, depending on the reputation distribution, as well as *when* this reputation distribution occurs.

6. SIMULATION RESULTS

We compare against the state-of-the-art reputation mechanism, namely the reputation mechanism with optimal stationary Markov (OSM) strategies in [1]–[3]. In a stationary Markov strategy, the action to take depends on the current state only, which is independent of when the current state appears. In contrast, the proposed strategy depends on the history of states. The same current state may lead to different actions in different periods. In the simulation of the optimal stationary Markov strategy, the recommended action (which is designed to be fixed) is the fair action α^{f}.

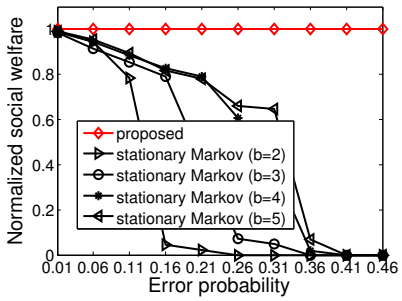

Fig. 1. Social welfare under different reputation update errors.

Table 2. Evolution of states and actions.

Period	State (s_0, s_1) (OSM/Proposed)	Recommended action (OSM/Proposed)	Action taken (OSM/Proposed)
0	(10,0)/(10,0)	Fair/Selfish	Selfish/Selfish
1	(9,1)/(9,1)	Fair/Fair	Selfish/Fair
2	(7,3)/(7,3)	Fair/Altruistic	Selfish/Altruistic
3	(6,4)/(4,6)	Fair/Fair	Selfish/Fair
4	(4,6)/(5,5)	Fair/Altruistic	Selfish/Altruistic
5	(4,6)/(6,4)	Fair/Altruistic	Selfish/Altruistic
6	(3,7)/(4,6)	Fair/Fair	Altruistic/Fair
7	(1,9)/(2,8)	Fair/Altruistic	Altruistic/Altruistic
8	(1,9)/(1,9)	Fair/Fair	Altruistic/Fair
9	(2,8)/(1,9)	Fair/Altruistic	Altruistic/Altruistic

We first consider a system with $N = 100$ users and discount factor $\delta = 0.99$. In Fig. 1, we compare the social welfare (i.e. the average payoff), which has been normalized to the social optimum $b-c$, of the two reputation mechanisms under different reputation update errors. We can see from Fig. 1 that as the error probability grows, the social welfare of the optimal stationary Markov strategy decreases, and drops to 0 when the error probability is large (e.g. when $\varepsilon > 0.4$). However, the proposed reputation mechanism achieves almost full efficiency under all the reputation update errors.

Then, we illustrate how the OSM strategy and the proposed strategy work differently, by comparing the evolution of the states, the recommended actions, and the actions taken by the users. We choose a small number of users, i.e. $N = 10$, such that the state evolution is more obvious in the first few periods. The evolution under the two strategies is shown in Table 2. The first difference is that in the reputation mechanism with OSM strategies, the recommended action is fixed and the action taken can be different from the recommended action. On the contrary, in the proposed reputation mechanism, the recommended action is not fixed over time and the action taken is always the same as the recommended action. The second difference is that in OSM strategies, the actions taken in different periods are the same as long as the current state is the same. In contrast, in the proposed strategy, the actions taken can be different even at the same current state.

7. CONCLUSION

We designed binary reputation mechanisms that can achieve the social optimum in the presence of reputation update errors. Simulation results demonstrate the significant performance gain of the proposed reputation mechanism over the state-of-the-art mechanisms.

8. REFERENCES

[1] Y. Zhang and M. van der Schaar, "Reputation-based incentive protocols in crowdsourcing applications," in *Proceedings of the IEEE INFOCOM*, 2012.

[2] C.-J. Ho, Y. Zhang, and M. van der Schaar, "Towards social norm design for crowdsourcing markets," in *Proceedings of the AAAI HCOMP*, 2012.

[3] Y. Zhang, J. Park, and M. van der Schaar, "Rating Protocols for Online Communities," to appear in *ACM Transaction on Economics and Computation*. Available at: http://arxiv.org/abs/1101.0272

[4] M. Kandori, "Social norms and community enforcement," *Review of Economic Studies*, vol. 59, no. 1, pp. 63 – 80, 1992.

[5] A. Blanc, Y.-K. Liu, and A. Vahdat, "Designing incentives for peer-to-peer routing," in *Proceedings of the IEEE INFOCOM*, 2005.

[6] M. Feldman, C. Papadimitriou, J. Chuang, and I. Stoica, "Free-riding and whitewashing in peer-to-peer systems," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 5, pp. 1010–1019, 2006.

[7] B. Q. Zhao, J. C. S. Lui, and D.-M. Chiu, "Analysis of adaptive incentive protocols for p2p networks," in *Proceedings of the IEEE INFOCOM*, 2009.

[8] C. Dellarocas, "Reputation mechanism design in online trading environments with pure moral hazard," *Information Systems Research*, vol. 16, no. 2, pp. 209–230, 2005.

[9] D. Fudenberg, D. K. Levine, and E. Maskin, "The folk theorem with imperfect public information," *Econometrica*, vol. 62, no. 5, pp. 997–1039, 1994.

[10] J. Hörner, T. Sugaya, S. Takahashi, and N. Vielle, "Recursive methods in discounted stochastic games: An algorithm for $\delta \to 1$ and a folk theorem," *Econometrica*, vol. 79, no. 4, pp. 1277–1318, 2011.

[11] D. Abreu, D. Pearce, and E. Stacchetti, "Toward a theory of discounted repeated games with imperfect monitoring," *Econometrica*, vol. 58, no. 5, pp. 1041–1063, 1990.

[12] G. Mailath and L. Samuelson, *Repeated Games and Reputations: Long-run Relationships*. Oxford, U.K.: Oxford University Press, 2006.

[13] Y. Xiao, Y. Zhang, and M. van der Schaar, "Appendix for SOCIALLY-OPTIMAL DESIGN OF CROWDSOURCING PLATFORMS WITH REPUTATION UPDATE ERRORS," Available at: http://www.ee.ucla.edu/~yxiao/AppendixICASSP2.pdf