Bonus or Not? Learn to Reward in Crowdsourcing

Ming Yin
Harvard University
Cambridge MA, USA
mingyin@fas.harvard.edu

Yiling Chen
Harvard University
Cambridge MA, USA
yiling@seas.harvard.edu

Abstract

Recent work has shown that the quality of work produced in a crowdsourcing working session can be influenced by the presence of performance-contingent financial incentives, such as bonuses for exceptional performance, in the session. We take an algorithmic approach to decide when to offer bonuses in a working session to improve the overall utility that a requester derives from the session. Specifically, we propose and train an input-output hidden Markov model to learn the impact of bonuses on work quality and then use this model to dynamically decide whether to offer a bonus on each task in a working session to maximize a requester's utility. Experiments on Amazon Mechanical Turk show that our approach leads to higher utility for the requester than fixed and random bonus schemes do. Simulations on synthesized data sets further demonstrate the robustness of our approach against different worker population and worker behavior in improving requester utility.

1 Introduction

Recent advances in crowdsourcing present great potentials in harnessing the dispersed crowd intelligence for various purposes, such as data collection [Wah *et al.*, 2011], scientific discovery [Khatib *et al.*, 2011] and disaster relief [Zook *et al.*, 2010]. A crucial factor that affects the success of a crowdsourcing attempt is the incentive design. Many domain-specific crowdsourcing systems are designed smartly to engage participants with their intrinsic motivations such as enjoyment in game playing and curiosity for new knowledge [Von Ahn, 2006; Savage, 2012]. However, on most general crowdsourcing platforms such as Amazon Mechanical Turk (MTurk), the primary type of incentive remains to be extrinsic motivation, that is, crowd workers complete tasks in exchange for monetary compensations.

Many studies have been conducted to understand the effects of financial incentives in crowdsourcing settings. Initial exploration suggested that financial incentives affect only the quantity of tasks that workers complete but not the work quality [Mason and Watts, 2009]. However, recent experiments showed that when a worker's reward is dependent on her performance in the task, financial incentives *can* influence the quality of work. For example, Harris [2011] found that adding performance-contingent bonus on top of the performance-independent base payment in each task leads to higher work quality and Ho *et al.* [2015] further identified that performance based payment can improve work quality when the task is "effort-responsive", that is, workers can produce higher quality work by exerting more efforts in the task. Moreover, it was also observed in previous studies that changing the magnitude of performance-contingent bonus in a task sequence leads to fluctuating work quality [Yin *et al.*, 2013a; 2013b].

While all such evidence suggests that performance-contingent rewards can affect the quality of crowd work, it is not necessarily always beneficial to provide such rewards in a crowdsourcing working session (i.e. a sequence of tasks) as the potentially improved quality comes with an increase in cost. Thus, a key challenge for a crowdsourcing system designer (i.e. requester) to address is whether and when to offer such rewards in a working session to maximize the overall utility he derives from the session. Currently, a common practice among requesters is to follow a fixed or a random scheme to offer a performance-contingent bonus on none of the tasks, all of the tasks or a number of randomly selected tasks in a working session, and each worker is awarded in the same way. Can we improve requester utility by dynamically adjusting the placement of bonuses in a working session?

We provide an initial answer to this question by presenting an algorithmic approach to control bonuses in crowdsourcing working sessions based on a probabilistic model. In particular, our algorithmic approach helps a requester to dynamically decide for each task in a working session, whether or not he should offer an extra performance-contingent bonus to a worker given his observation on the worker's past performance. We make the following contributions:

- We propose to train an input-output hidden Markov model (IOHMM) to characterize the impact of performance-contingent bonuses on work quality in a working session.
- We augment the learned IOHMM with a requester's utility function and turn the bonus placement problem into a problem of utility maximization under uncertainty. Three heuristic algorithms are provided to solve for a utility-maximizing, dynamic policy of bonus placement.

- We design and conduct a randomized experiment on Amazon Mechanical Turk (MTurk) to evaluate our approach of dynamic bonus placement against three other baseline bonus schemes. Our approach improves the requester utility by as much as 27.22% beyond what is achieved by the best-performing baseline bonus scheme. To the best of our knowledge, this is the first time that the effectiveness of algorithmically-controlled bonus schemes is shown with *real* crowd workers.
- We further validate the robustness of our approach through simulations on two synthesized data sets that are generated according to two different worker behavior models respectively. Simulation results show that our approach consistently leads to improved requester utility for both models of worker behavior and different compositions of worker population.

1.1 Related Work

The lack of quality assurance hence the potentially low requester utility is a big concern in crowdsourcing. To tackle this problem, researchers have proposed different solutions from various aspects. For example, to boost the accuracy of labeling tasks based on redundant annotations, a large number of algorithms are designed to infer task difficulty, worker reliability and the underlying true labels simultaneously [Whitehill *et al.*, 2009; Raykar *et al.*, 2009]. Researchers also take real-world constraints such as the limited budget into consideration and investigate how to optimally allocate the budget among tasks and workers to elicit high quality labels [Chen *et al.*, 2013]. In addition, machine learning and decision-theoretic techniques have been adopted to guarantee the requester's utility by dynamically controlling the crowdsourcing process through decisions on worker recruitment [Kamar *et al.*, 2012], task assignment [Dai *et al.*, 2010] and workflow switches [Lin *et al.*, 2012].

One common assumption of these previous studies is that worker performance is decided by the worker's inherent capability level, which is *independent* of her working environment, such as how much she gets paid in each of the completed tasks. The empirical observations that we have mentioned before, however, indicate that worker performance can be *dependent* on the provided incentives in tasks, and thus lead to an interesting question of how to reward workers in an optimal way such that the requester utility is maximized. To this end, Wang and Ipeirotis [2013] proposed a "payment with reimbursement" scheme based on the conjectures that the fluctuation of payments in task sequences is undesirable as workers may interpret a decrease of financial incentives as a punishment. Ho *et al.* [2014] used the classical principal-agent model to characterize how a worker makes strategic decisions when provided with a performance-contingent payment and studied how to adaptively adjust such payment over time. Different from these studies, in this paper, we propose to first learn a model from the data to empirically characterize worker behavior in reaction to financial incentives, and then use this model to make decisions on the placement of bonuses. Huang *et al.* [2010] adopted a similar approach to optimize crowdsourcing task environment. However, they focused on design variables such as how many HITs (i.e. Human Intelligence Tasks on MTurk) to post and how many tasks to be bundled in one HIT, and followed a fixed rate of pay for each task in the HIT.

Finally, our work is different from several pricing mechanisms proposed to elicit more (or faster) work from rational workers given a fixed budget [Singer and Mittal, 2013; Singla and Krause, 2013; Gao and Parameswaran, 2014] — our goal is to elicit *high-quality* work and we have no assumption on the rationality of workers.

2 An Algorithmic Approach

Our algorithmic approach for bonus control in a crowdsourcing working session can be decomposed into two steps: First, we collect a training data set and learn an input-output hidden Markov model to characterize how the presence of performance-contingent bonuses affect the work quality in a working session; second, we incorporate utilities with the learned IOHMM and dynamically determine whether to place a bonus on each task for a new incoming worker by solving a utility maximization problem.

2.1 Characterizing the Impact of Bonuses with an IOHMM

Suppose we have collected a data set on worker behavior in a working session, which is a sequence of tasks that a worker completes without interruption. For each task in the session, we record the provided bonus level and the worker performance in it. To enable the algorithmic bonus control for future workers, we first aim at leveraging this training data set to quantitatively model the impact of bonuses on work quality in a working session.

A natural model choice is the input-output hidden Markov model, which is a generative probabilistic model for sequential data. In our context, consider that time step t in IOHMM corresponds to the t-th task in a session. For each t, there is an input a_t and an output x_t, which represents the level of bonus and the observed work quality in task t, respectively. Furthermore, there is a hidden state z_t, which comes from one of the K states and can capture the unobserved properties of a crowd worker in the t-th task. Worker behavior is controlled by the transition and emission probabilities: $P_{tr}(z_t|z_{t-1}, a_t)$ gives a worker's chance of transiting to state z_t in the t-th task conditioned on that her state in the $(t-1)$-th task is z_{t-1} and the input for the t-th task is a_t, and $P_e(x_t|z_t, a_t)$ represents how likely a worker will produce output x_t in the t-th task given her current state z_t and input a_t.

For simplicity, we illustrate our algorithmic approach in the rest of the paper assuming that the work quality for a task is binary (i.e. incorrect or correct, low-quality or high-quality, etc.) and the requester can choose one of two bonus levels in a task — either place a bonus of fixed magnitude or not provide any bonus at all. Suppose the number of tasks bundled in a working session is T. The IOHMM model details for this simplified problem are as follows:

- **Inputs**: $a_t \in \{0, 1\}$, t = 1, 2, $\cdots$, T, with 0 representing bonus is not placed on the task.
- **Outputs**: $x_t \in \{0, 1\}$, t = 1, 2, $\cdots$, T, with 0 representing an incorrect (or low-quality) answer for the task.

- **Hidden States**: $z_t \in \{1, 2, \cdots, K\}$, $t = 1, 2, \cdots, T$.
- **Transition probability**: $P_{tr}(z_t|z_{t-1}, a_t)$; e.g., $P_{tr}(k'|k, 0)$ gives the probability of a worker transiting from hidden state k in task $t-1$ to hidden state k' in task t given that there is no extra bonus in task t.
- **Emission probability**: $P_e(x_t|z_t, a_t)$; e.g., $P_e(1|k, 0)$ gives the probability of a worker submitting a correct (or high-quality) answer to task t when the current state is k and no extra bonus is offered on task t.

With the training data set that is collected from a group of workers, we can learn the IOHMM for this worker population through an expectation-maximization algorithm, which is similar to the Baum-Welch algorithm for learning hidden Markov models [Bengio and Frasconi, 1996].

2.2 Decision Making with the learned IOHMM

Given the learned IOHMM $\mathcal{M}$ to describe the impact of bonuses on work quality in a working session, the next question is to decide for a new incoming worker whether or not the requester places a bonus on each task in her working session.

To quantify the requester's tradeoff between work quality and cost, we assume that the requester obtains a utility of w_h (or w_l) when he gets a high-quality (or low-quality) answer, while the economic cost for paying a performance-contingent bonus is c. We further assume that the requester has a quasi-linear utility function $U = w_h N_{HQ} + w_l N_{LQ} - cN_B$, where N_{HQ} (or N_{LQ}) and N_B represents the number of high-quality (or low-quality) answers elicited and the number of times a performance-contingent bonus is incurred, respectively. As a requester can continuously make observations on worker performance over time, we are interested in dynamically controlling the placement of bonus in a working session in an *online* fashion. That is, we keep making decisions on whether the requester should provide a bonus to a worker in her *next* task given the history of inputs and outputs in all tasks that the worker has completed so far in the session.

In particular, for a worker who has completed t_c tasks, we estimate the distribution of her current state as $\boldsymbol{b}(t_c)$ (i.e. the "state belief") based on $\mathcal{M}$. We define $EU_{max}(\boldsymbol{b}, a, l)$ as the *maximum* expected utility a requester can obtain in the next l tasks given that the current state belief is $\boldsymbol{b} = (b(1), \cdots, b(K))$, the input level for the next task is a, and input levels for later tasks follow the optimal policy. Thus, the optimal input level for the next task is $a_{t_c+1} = \text{argmax}_{a\in\{0,1\}} EU_{max}(\boldsymbol{b}(t_c), a, T-t_c)$ — when $a_{t_c+1} = 1$, we offer a bonus on the next task; otherwise, we don't.

$EU_{max}(\boldsymbol{b}, a, l)$ can be calculated recursively. Specifically, we denote $R(\boldsymbol{b}, a) = \sum_{i=1}^{K} b(i)(\sum_{j=1}^{K} P_{tr}(j|i, a) \cdot (P_e(0|j, a)w_l + P_e(1|j, a)(w_h - \mathbb{I}(a = 1)c)))$ as the requester's expected utility in the next task when the current state belief is $\boldsymbol{b}$ and the next input is a. When there is only one task left in the session, that is, $l = 1$, $EU_{max}(\boldsymbol{b}, a, l) = R(\boldsymbol{b}, a)$; otherwise, we have:

$$EU_{max}(\boldsymbol{b}, a, l) = R(\boldsymbol{b}, a) + \sum_{x\in\{0,1\}} (\sum_{i=1}^{K} b(i) \sum_{j=1}^{K} P_{tr}(j|i, a)P_e(x|j, a))V(\boldsymbol{b}'_{\boldsymbol{a},\boldsymbol{x}}, l-1)$$

where $V(\boldsymbol{b}, l) = \max_{a\in\{0,1\}} EU_{max}(\boldsymbol{b}, a, l)$ and $\boldsymbol{b}'_{\boldsymbol{a},\boldsymbol{x}}$ is the updated state belief if the input for the next task is a and the observed output is x, which can be computed as follows:

$$b'_{a,x}(j) \propto \sum_{i=1}^{K} b(i)P_{tr}(j|i, a)P_e(x|j, a)$$

And finally, in preparation for the decision making in future tasks, we update the belief state after implementing the input level a_{t_c+1} and observing the output level x_{t_c+1}, that is, $\boldsymbol{b}(t_c + 1) = \boldsymbol{b}'_{\boldsymbol{a}_{t_c+1},\boldsymbol{x}_{t_c+1}}$.

Heuristic Solutions

The decision making problem above is essentially equivalent to solve a finite-horizon partially observable Markov decision process (POMDP), with "action" corresponding to "input" in $\mathcal{M}$ and the reward of taking action a in state k being $R(k, a) = \sum_{i=1}^{K} P_{tr}(i|k, a)(P_e(0|i, a)w_l + P_e(1|i, a)(w_h - \mathbb{I}(a = 1)c))$. In practice, finding exact solutions for POMDPs are often computationally intractable [Papadimitriou and Tsitsiklis, 1987]. Therefore, we list a few heuristic algorithms to solve the problem approximately:

Algorithm 1 (n-step look-ahead) When making decisions for whether to place an extra bonus on the next task, we look ahead for at most n tasks. That is, $a_{t_c+1} = \text{argmax}_{a\in\{0,1\}} EU_{max}(\boldsymbol{b}(t_c), a, n')$, where $n' = \min(n, T-t_c)$. A similar strategy is used in [Dai *et al.*, 2010] for optimal control of crowdsourcing workflows.

Note that if a worker's hidden state z_{t_c} after completing t_c tasks as well as her states in the future tasks can be accurately identified, the finite horizon POMDP degenerates into a finite horizon MDP, for which we can calculate the Q-functions and optimal policies efficiently. We thus consider two algorithms that leverage this advantage:

Algorithm 2 (MLS-MDP) We infer the most likely sequence (MLS) of hidden states up to the current task (i.e. $\hat{\boldsymbol{Z}}_{\boldsymbol{1}}^{\boldsymbol{t_c}} = (\hat{z_1}, \cdots, \hat{z_{t_c}})$) using the Viterbi algorithm [Viterbi, 1967] and estimate z_{t_c} as $\hat{z_{t_c}}$. The input level for the next task a_{t_c+1} is then set to be $\pi_{T-t_c}(z_{t_c})$, that is, the optimal MDP policy on state z_{t_c} when the length of horizon is $T-t_c$.

Algorithm 3 (Q-MDP) We first calculate $Q_{T-t_c}(k, a)$ for the underlying MDP, which is the Q-function value for taking action a on state k with $T-t_c$ steps to go. Then, $a_{t_c+1} = \text{argmax}_{a\in\{0,1\}} \sum_{k=1}^{K} p_k Q_{T-t_c}(k, a)$, with p_k being the k-th element of $\boldsymbol{b}(t_c)$ [Littman *et al.*, 1995].

3 MTurk Experiment

To examine whether our algorithmic approach of placing bonuses can effectively improve requester utility in a real crowdsourcing working session, we design and conduct an online experiment on MTurk.

3.1 Task

We use a word puzzle game as our task in the experiment: In each task, a worker will see a 12×12 board filled with capital letters and a "target" word on the screen. This target word can

be placed on the board horizontally, vertically or diagonally and for multiple times. The worker is asked to identify as many appearance of the target word on the board as possible and specify the location of each identified appearance. Each worker completes 9 tasks in a working session (i.e. one HIT). A similar task is previously used by Mason and Watts [2009] to study the effects of financial incentives on the performance of crowds.

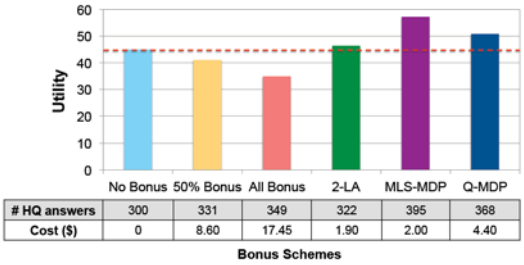
A worker earns a performance-independent reward of 5 cents in each task, that is, the base payment for the HIT is 45 cents. In addition, we also inform the worker that some tasks in the session are "bonus tasks" (specified with a bonus icon), in which she may earn an extra bonus of 5 cents if she submits a *high-quality* answer to it by pointing out more than 80% of all appearances of the target word. By design, each board in our experiment contains the target word 11 times (workers are not aware of this fact, however), which means that a worker can only earn the extra reward in a bonus task if she identifies the target word for at least 9 times.

3.2 Procedure

Corresponding to the two steps in our algorithmic approach, we divide our experiment into two phases.

In the first phase, we collect a training data set by recruiting 50 MTurk workers to participate in our experiment. For each of the 9 tasks that a worker completes in the HIT, we randomly set it as a bonus task with a 20% chance; whether that task is a bonus task and whether the worker submits a high-quality answer to it (i.e. finds out the target word at least 9 times) is recorded. We then learn an IOHMM to understand the impact of bonuses on worker performance in the word puzzle game sequences using the collected data set. Specifically, we run the expectation-maximization algorithm with 100000 random restarts, and each run is terminated after convergence or 500 iterations, whichever is reached earlier. In searching for a parsimonious model, we experiment on a range of values for the number of hidden states ($K = 1 \sim 7$) to train different IOHMMs, and the IOHMM with the maximized Bayesian information criterion (BIC) score [Schwarz and others, 1978] is selected to be used in the second phase. In our experiment, $K = 2$ for the selected IOHMM.

The second phase of our experiment is the testing phase, in which we have 6 experimental treatments and each treatment corresponds to one bonus scheme. In particular, we include 3 dynamic bonus schemes that are designed according to our algorithmic approach using different heuristics (i.e. 2-step look-ahead[1], MLS-MDP and Q-MDP). When these schemes are used in treatments, we keep track of a worker's performance in the session and use the learned IOHMM to strategically make a decision on whether to offer an extra bonus to the worker on the next task. As a comparison, we also consider 3 fixed or random baseline bonus schemes: not placing bonus in any task (No Bonus), always placing bonuses in all tasks (All Bonus) and randomly choosing 50% of the tasks to place bonuses (50% Bonus). The utility parameters we use in the experiment are $w_h = 0.15$, $w_l = 0$ and $c = 0.05$[2].



	No Bonus	50% Bonus	All Bonus	2-LA	MLS-MDP	Q-MDP
# HQ answers	300	331	349	322	395	368
Cost ($)	0	8.60	17.45	1.90	2.00	4.40

Figure 1: The requester's utility across 6 treatments in the second phase MTurk experiment.

To make sure the IOHMM learned from the training phase is useful for the testing phase, we recruit workers from the same pool by running our second phase experiment exactly 2 weeks after the first phase experiment around the same time. Each worker is randomly assigned to one treatment and 50 workers are recruited for each treatment. All workers in a treatment are paid according to the bonus scheme of that treatment. We again collect data on the presence of bonus and work quality for each task and each worker.

Our experiment is limited to U.S. workers and each worker is allowed to take the HIT only once.

3.3 Results

Figure 1 compares the overall utility a requester derives from *all* 50 workers in the working session across the 6 treatments of our second phase experiment. As we can see in the figure, among the 3 baseline bonus schemes, the best scheme is to pay no bonus at all. Yet, following our algorithmic approach, the 3 dynamic bonus schemes (i.e. 2-step look-ahead, MLS-MDP and Q-MDP) lead to an increase of 3.11%, 27.22% and 12.89% in the requester's utility, respectively, compared to the No Bonus scheme. To see whether the utility improvement is material, we decompose the overall requester utility in each treatment into the number of high-quality answers the requester elicits (hence the economic benefits) and the cost the requester pays to encourage better performance (see the table in Figure 1). Results suggest that the requester can elicit *more* high-quality work with *lower* cost by applying our dynamic bonus schemes. For example, compared to the 50% Bonus scheme, a requester using the 2-step look-ahead scheme obtains a similar number of high-quality answers with a 77.9% saving in money; while a requester following the MLS-MDP (or Q-MDP) scheme elicits 19.3% (or 11.2%) more high-quality answers with roughly a quarter (or a half) of the cost.

The statistical significance of the improvement in utility brought by our algorithmic approach is further examined through Wilcoxon rank-sum tests. Specifically, we compute the utility a requester obtains from *each* worker in every treatment. Thus, in total, we have 6 samples of requester utility with 50 data points in each sample. We conduct pairwise comparisons to test whether a pair of samples have the same

[1]We set $n = 2$ to balance between performance and efficiency based on simulations for workers who indeed behave according to the learned IOHMM.

[2]c is set to be 0.05 as the bonus magnitude used in the experiment is \$0.05. We set $w_h = 0.15$ and $w_l = 0$ to ensure that a dynamic bonus scheme doesn't become a No Bonus or All Bonus scheme.

mean value, and the p-values of the tests are reported in Table 1. Almost all pairwise comparisons are statistically significant at the $p = 0.05$ level, which suggests that our approach helps the requester to improve his utility.

Table 1: p-values of the Wilcoxon rank-sum tests for pairwise comparisons.

	No Bonus	50% Bonus	All Bonus
2-LA	1	0.007	<0.001
MLS-MDP	0.03	<0.001	<0.001
Q-MDP	0.38	<0.001	<0.001

Finally, to get a qualitative intuition of how our dynamic bonus schemes work, we pick a few exemplary workers from the treatment in which the MLS-MDP bonus scheme is used and take a close look at how they are awarded in the working session. Figure 2 displays for each of the 4 selected workers, whether a bonus is provided and whether her submitted answer is of high quality for each task in the working session. The comparison between worker A and worker B first suggests that our algorithmic approach can effectively differentiate "diligent" workers from "lazy" workers and reward them differently: For a diligent worker (worker A) who always submits high-quality answers, there is no need for the requester to place extra bonuses in the working session; however, for a lazy worker who can be responsive to financial incentives (worker B), the requester keeps offering bonuses in hope of increasing the work quality through providing additional motivation to the worker[3]. In fact, the MLS-MDP bonus scheme strategically focuses on incentivizing lazy workers: On average, the requester offers a bonus on 6 tasks to a worker who performs well in at most half of the tasks in the working session, while for a worker who performs well in more than half of the tasks, the requester offers a bonus on only 0.49 tasks. Furthermore, our algorithmic approach also seems to offer bonuses at the right timing. On the one hand, for a worker who starts a session with unsatisfying performance (worker C), the requester keeps placing bonus on each task to incentivize better performance until the worker stabilizes in submitting high-quality answers; on the other hand, for a worker who slacks off from her initial good performance (worker D), the requester provides extra incentives in time to bring back hard working from the worker.

4 Simulation

The performance of the dynamic bonus schemes in our MTurk experiment suggests the promise of algorithmically controlling the provision of financial incentives in crowdsourcing. However, one may wonder that following our algorithmic approach, whether the high requester utility can always be obtained in various worker populations where workers potentially behave in different ways. To understand the robustness of our approach, we further run simulations on two synthesized data sets and each data set is generated according to a predefined worker behavior model.

[3]Offering bonus per se is *not* costly; the cost will only be incurred when the work quality in a bonus task meets the predefined standard.

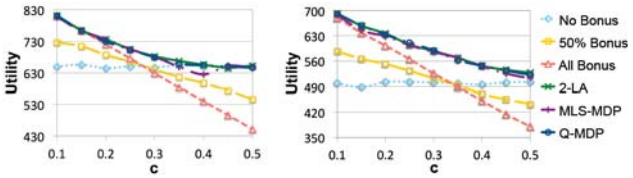
Worker	Inputs & Outputs in the Working Session									
A	Bonus?	✗	✗	✗	✗	✗	✗	✗	✗	✗
	High-quality?	1	1	1	1	1	1	1	1	1
B	Bonus?	✗	✓	✓	✓	✓	✓	✓	✓	✓
	High-quality?	0	0	0	0	0	0	0	0	0
C	Bonus?	✗	✓	✓	✓	✓	✓	✗	✗	✗
	High-quality?	0	0	0	1	1	1	1	1	1
D	Bonus?	✗	✗	✗	✗	✓	✓	✓	✗	✗
	High-quality?	1	1	0	0	1	1	1	1	0

Figure 2: Examples for offering bonus to a worker in the working session based on the MLS-MDP bonus scheme.

Specifically, given a particular worker behavior model, in the training phase, we generate a data set of 3000 workers, while each worker completes a session of 50 tasks (20% of them are randomly selected as bonus tasks) and decides her performance in each task probabilistically according to the given model. In the testing phase, we consider the same 6 bonus schemes as those in our MTurk experiment. Six groups of testing data are thus generated: Each group is assigned to a unique bonus scheme and is composed of 100 workers; each worker completes a session of 10 tasks and is paid according to the bonus scheme of her group, while her performance in each task is controlled by the same behavior model as that used for generating the training data set. The requester's utility under a specific bonus scheme is calculated as the sum of utilities that the requester derives from all 100 workers of the corresponding group, with w_l and w_h set to be 0 and 1, respectively. We repeat the simulation 30 times and report the mean value of the requester's utility for each scheme.

4.1 Model 1: Workers with Two Capability Levels

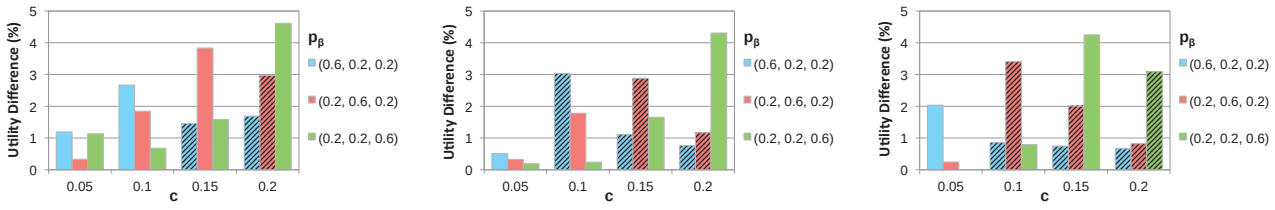
In our first worker behavior model, we assume that when a performance-contingent bonus is placed (or not placed) in a task, the probability for worker i to submit a high-quality answer in a task is acc_i^h (or acc_i^l). This model is consistent with previous empirical observations that the existence of performance-based bonuses can affect the worker performance [Harris, 2011; Ho *et al.*, 2015].



(a) Population 1 (Uniform) (b) Population 2 (Beta)

Figure 3: The requester's utility when workers have two capability levels in reaction to the placement of bonus. Error bars are omitted as they are too small.

We construct two different worker populations based on this worker behavior model. To test the performance of different bonus schemes when a requester's willingness to reward workers differs, we also vary the economic costs (i.e. magnitude) of the performance-contingent bonus c from 0.1 to 0.5 in the simulation. Figure 3(a) demonstrates the simulation results for a population that is composed of 2 types of workers: For the first type, $acc_i^l = 0.5$ and $acc_i^h = 0.9$; while for the second type, $acc_i^l = 0.8$ and $acc_i^h = 0.9$. Each



(a) $\boldsymbol{p_\alpha} = (0.6, 0.2, 0.2)$ (b) $\boldsymbol{p_\alpha} = (0.2, 0.6, 0.2)$ (c) $\boldsymbol{p_\alpha} = (0.2, 0.2, 0.6)$

Figure 4: The requester's average utility increase (in percentage) when following a dynamic bonus scheme rather than the best-performing baseline scheme; workers are influenced by their reference payment levels in mind. Shaded (Unshaded) bars indicate that the best-performing baseline scheme is the "No Bonus" ("All Bonus") scheme in that condition.

worker in the population is drawn uniformly randomly from the two types. Similarly, Figure 3(b) corresponds to another population where each worker draws her accuracy from Beta distributions: $acc_i^l \sim Beta(2,2)$ and $acc_i^h \sim Beta(6,2)$. As the figures suggest, for both populations, when the magnitude of bonus is small (or large) enough such that adding a bonus is always beneficial (or too costly), the dynamic bonus schemes lead to similar requester utility as the All Bonus (or No Bonus) scheme does. However, when the bonus magnitude is moderate, the dynamic bonus schemes robustly result in higher requester utility compared to any single baseline bonus scheme.

4.2 Model 2: Workers Influenced by Reference Payment Levels

Previous literature suggests another possible worker behavior in reaction to financial incentives in a working session. That is, a worker maintains and updates a reference point of "appropriate" payment level when she completes tasks in a working session and decides her performance in each task by comparing the provided payment with the reference of that time [Yin *et al.*, 2013a; 2013b; Popescu and Wu, 2007; Akerlof and Yellen, 1990]. To see how our algorithmic approach performs when workers indeed behave in this way, we define the second worker behavior model. In particular, we assume that each worker behaves according to an IOHMM, with the hidden state z_t corresponding to the reference payment level r_{z_t} in the worker's mind in the t-th task[4]. Each worker i is further characterized by her skill level α_i and her responsiveness to financial incentives β_i, and the emission probability in task t is parameterized as $P_e(1|z_t, a_t) = \frac{1}{1+e^{-\alpha_i - \beta_i(a_t - r_{z_t})}}$. Obviously, the larger α_i or β_i is, the worker is more skilled or more responsive to rewards hence more likely to produce high-quality answers.

For manageability, we assume in the simulation that each worker has $K = 3$ hidden states (i.e. 3 discrete reference payment levels), that is, $\boldsymbol{r} = (r_1, r_2, r_3) = (0.2, 0.6, 1.2)$. Each worker updates her reference payment level in the working session according the transition probability matrices:

$$T^0 = \begin{pmatrix} 0.8 & 0.15 & 0.05 \\ 0.3 & 0.6 & 0.1 \\ 0.2 & 0.4 & 0.4 \end{pmatrix}, T^1 = \begin{pmatrix} 0.4 & 0.4 & 0.2 \\ 0.1 & 0.5 & 0.4 \\ 0.05 & 0.1 & 0.85 \end{pmatrix}$$

[4]The reference payment level is defined relative to the magnitude of the bonus, e.g. $r_k = 0.5$ means that in state k, the worker considers a half of the current bonus as an appropriate payment.

where the (i,j)-th element of matrix T^a represents the probability for a worker to update her reference payment level from r_i to r_j given the current input level a, i.e. $P_{tr}(j|i,a)$.

Furthermore, we assume that there are 3 possible skill levels for a worker, i.e. $\alpha_i \in \{0, 1, 3\}$, and each worker i draws her skill level according to the categorical distribution $\boldsymbol{p_\alpha} = (p_\alpha^1, p_\alpha^2, p_\alpha^3)$, where $p_\alpha^1 + p_\alpha^2 + p_\alpha^3 = 1$ and p_α^1 (or p_α^2, p_α^3) represents the probability that $\alpha_i = 0$ (or $1, 3$). Similarly, we also consider 3 levels of responsiveness to financial incentives, that is, $\beta_i \in \{0, 1, 3\}$, and each worker draws her β_i according to another categorical distribution $\boldsymbol{p_\beta} = (p_\beta^1, p_\beta^2, p_\beta^3)$. Varying $\boldsymbol{p_\alpha}$ and $\boldsymbol{p_\beta}$ thus provides us the flexibility to construct a number of populations in which various types of workers are mixed in different proportions. For each population and 4 selected bonus magnitude (i.e. $c = 0.05, 0.1, 0.15, 0.2$), Figure 4 displays the utility difference (in percentage) a requester obtains by following a dynamic bonus scheme (averaged over the 3 dynamic schemes) over following the *best-performing* baseline scheme to control bonuses in the session: On the one hand, the best-performing baseline scheme *differs* across various populations and magnitude of bonus, suggesting that following a single baseline bonus scheme can't guarantee a high requester utility in all conditions; on the other hand, following our dynamic bonus schemes, the requester can consistently obtain similar or higher utility than what he could have obtained by following the *best-performing* baseline scheme, which again implies that the improved performance of our approach is robust.

5 Conclusions and Future Work

In this paper, we take an algorithmic approach to dynamically control whether and when to place a bonus in a crowdsourcing working session. Our MTurk experiment and simulation results suggest that our approach can robustly lead to significant improvement in requester utility than fixed and random bonus schemes do. There are many interesting future directions for this work. For example, our current approach leads to a policy that provides more bonus opportunities for workers with lower accuracy. This may drive high accuracy workers away in the long term. It will be interesting to understand the long-term impact of our algorithmic approach.

Acknowledgements

We thank the support of the National Science Foundation under grant CCF-1301976 and the Xerox Foundation on this

work. Any opinions, findings, conclusions, or recommendations expressed here are those of the authors alone.

References

[Akerlof and Yellen, 1990] George A Akerlof and Janet L Yellen. The fair wage-effort hypothesis and unemployment. *The Quarterly Journal of Economics*, pages 255–283, 1990.

[Bengio and Frasconi, 1996] Yoshua Bengio and Paolo Frasconi. Input-output hmms for sequence processing. *Neural Networks, IEEE Transactions on*, 7(5):1231–1249, 1996.

[Chen *et al.*, 2013] Xi Chen, Qihang Lin, and Dengyong Zhou. Optimistic knowledge gradient policy for optimal budget allocation in crowdsourcing. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 64–72, 2013.

[Dai *et al.*, 2010] Peng Dai, Daniel Sabey Weld, et al. Decision-theoretic control of crowd-sourced workflows. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[Gao and Parameswaran, 2014] Yihan Gao and Aditya Parameswaran. Finish them!: Pricing algorithms for human computation. *Proc. VLDB Endow.*, 7(14):1965–1976, October 2014.

[Harris, 2011] Christopher Harris. You're Hired! An Examination of Crowdsourcing Incentive Models in Human Resource Tasks. In *Proceedings of the Workshop on Crowdsourcing for Search and Data Mining (CSDM) at the Fourth ACM International Conference on Web Search and Data Mining (WSDM)*, pages 15–18, Hong Kong, China, February 2011.

[Ho *et al.*, 2014] Chien-Ju Ho, Aleksandrs Slivkins, and Jennifer Wortman Vaughan. Adaptive contract design for crowdsourcing markets: bandit algorithms for repeated principal-agent problems. In *Proceedings of the fifteenth ACM conference on Economics and computation*, pages 359–376. ACM, 2014.

[Ho *et al.*, 2015] Chien-Ju Ho, Aleksandrs Slivkins, Siddharth Suri, and Jennifer Wortman Vaughan. Incentivize high quality crowdwork. In *Proceedings of the 24th World Wide Web Conference, WWW*, 2015.

[Huang *et al.*, 2010] Eric Huang, Haoqi Zhang, David C Parkes, Krzysztof Z Gajos, and Yiling Chen. Toward automatic task design: a progress report. In *Proceedings of the ACM SIGKDD workshop on human computation*, pages 77–85. ACM, 2010.

[Kamar *et al.*, 2012] Ece Kamar, Severin Hacker, and Eric Horvitz. Combining human and machine intelligence in large-scale crowdsourcing. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 467–474. International Foundation for Autonomous Agents and Multiagent Systems, 2012.

[Khatib *et al.*, 2011] Firas Khatib, Seth Cooper, Michael D Tyka, Kefan Xu, Ilya Makedon, Zoran Popović, David Baker, and Foldit Players. Algorithm discovery by protein folding game players. *Proceedings of the National Academy of Sciences*, 108(47):18949–18953, 2011.

[Lin *et al.*, 2012] Christopher H. Lin, Mausam Daniel, and S. Weld. Dynamically switching between synergistic workflows for crowdsourcing. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, 2012.

[Littman *et al.*, 1995] Michael L. Littman, Anthony R. Cassandra, and Leslie Pack Kaelbling. Learning policies for partially observable environments: Scaling up. In *International Conference on Machine Learning (ICML)*. Morgan Kaufmann, 1995.

[Mason and Watts, 2009] Winter Mason and Duncan J. Watts. Financial incentives and the "performance of crowds". In *Proceedings of the ACM SIGKDD Workshop on Human Computation*, HCOMP '09, pages 77–85, New York, NY, USA, 2009. ACM.

[Papadimitriou and Tsitsiklis, 1987] Christos H Papadimitriou and John N Tsitsiklis. The complexity of markov decision processes. *Mathematics of operations research*, 12(3):441–450, 1987.

[Popescu and Wu, 2007] Ioana Popescu and Yaozhong Wu. Dynamic pricing strategies with reference effects. *Operations Research*, 55(3):413–429, 2007.

[Raykar *et al.*, 2009] Vikas C. Raykar, Shipeng Yu, Linda H. Zhao, Anna K. Jerebko, Charles Florin, Gerardo Hermosillo Valadez, Luca Bogoni, and Linda Moy. Supervised learning from multiple experts: whom to trust when everyone lies a bit. In *ICML'09*, 2009.

[Savage, 2012] Neil Savage. Gaining wisdom from crowds. *Communications of the ACM*, 55(3):13–15, 2012.

[Schwarz and others, 1978] Gideon Schwarz et al. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

[Singer and Mittal, 2013] Yaron Singer and Manas Mittal. Pricing mechanisms for crowdsourcing markets. In *Proceedings of the 22nd international conference on World Wide Web*, pages 1157–1166. International World Wide Web Conferences Steering Committee, 2013.

[Singla and Krause, 2013] Adish Singla and Andreas Krause. Truthful incentives in crowdsourcing tasks using regret minimization mechanisms. In *Proceedings of the 22nd international conference on World Wide Web*, pages 1167–1178. International World Wide Web Conferences Steering Committee, 2013.

[Viterbi, 1967] Andrew J Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *Information Theory, IEEE Transactions on*, 13(2):260–269, 1967.

[Von Ahn, 2006] Luis Von Ahn. Games with a purpose. *Computer*, 39(6):92–94, 2006.

[Wah *et al.*, 2011] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

[Wang and Ipeirotis, 2013] Jing Wang and Panagiotis Ipeirotis. Quality-based pricing for crowdsourced workers. 2013.

[Whitehill *et al.*, 2009] Jacob Whitehill, Ting fan Wu, Jacob Bergsma, Javier R. Movellan, and Paul L. Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in Neural Information Processing Systems 22*, pages 2035–2043, 2009.

[Yin *et al.*, 2013a] Ming Yin, Yiling Chen, and Yu-An Sun. The effects of performance-contingent financial incentives in online labor markets. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence, AAAI*, volume 13, 2013.

[Yin *et al.*, 2013b] Ming Yin, Yiling Chen, and Yu-An Sun. Task sequence design: Evidence on price and difficulty. In *First AAAI Conference on Human Computation and Crowdsourcing*, 2013.

[Zook *et al.*, 2010] Matthew Zook, Mark Graham, Taylor Shelton, and Sean Gorman. Volunteered geographic information and crowdsourcing disaster relief: a case study of the haitian earthquake. *World Medical & Health Policy*, 2(2):7–33, 2010.