Title
Design and Analysis of Crowdsourcing Mechanisms

Author
Ho, Chien-Ju

Publication Date
2015

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

Design and Analysis of Crowdsourcing Mechanisms

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Computer Science

by

Chien-Ju Ho

2015

ABSTRACT OF THE DISSERTATION

Design and Analysis of Crowdsourcing Mechanisms

by

Chien-Ju Ho

Doctor of Philosophy in Computer Science

University of California, Los Angeles, 2015

Professor Jennifer Wortman Vaughan, Co-Chair

Professor Adnan Youssef Darwiche, Co-Chair

Crowdsourcing is a paradigm for utilizing crowd intelligence to help solve problems that computers alone can not yet solve. In recent years, crowdsourcing has gained great success as the Internet makes it easy to engage the crowd to work together. For instance, Wikipedia, the largest encyclopedia in the world, is created with the help of online users. Games with A Purpose, crowdsourcing markets, and online reviewing sites are also platforms that rely on human contributions to accomplish various tasks. However, despite the great success of these platforms, how to better design crowdsourcing mechanisms is still not well understood. Most of the crowdsourcing mechanisms suffer from the prevalence of low quality work generated by humans.

In this dissertation, we explore the design and analysis of crowdsourcing mechanisms, with the goal of understanding how to obtain high-quality information from participating users. We approach the design problem from three interleaving directions. First, we study how to assign tasks to workers with suitable skills using techniques from machine learning and online optimization. Second, we explore how to design incentives to motivate users to provide high quality contribution. Finally, we run behavioral experiments to understand how users behave in real systems with the goal of developing more realistic user behavioral models.

The dissertation of Chien-Ju Ho is approved.

Richard E. Korf

Mihaela van der Schaar

Junghoo Cho

Adnan Youssef Darwiche, Committee Co-Chair

Jennifer Wortman Vaughan, Committee Co-Chair

University of California, Los Angeles

2015

TABLE OF CONTENTS

ACKNOWLEDGMENTS

First and foremost, to my advisor, Jenn Wortman Vaughan, thank you. Jenn is the best advisor I can hope for. She is so supportive in every aspect. She gives me the freedom to do whatever I am interested in and provides me guidance along the way. Even when things are getting more difficult, she always makes sure I can get the necessary resources. I can't express how grateful I am to have her as my advisor.

Thanks to Yiling Chen for hosting me at Harvard and spending so much time to meet with me whenever I needed. I am really fortunate to have another mentor who I feel comfortable to ask for research guidance or career advices. Thanks to Jane Hsu for first showing me what doing research is like while I was an undergraduate student. I miss the time we occasionally bumped into each other in the hallway and then chatted for hours about research, career, and life.

Thanks to my entire committee, Mihaela van der Schaar, Adnan Darwiche, John Cho, and Richard Korf, for the great advices and suggestions.

Thanks to all my collaborators, Alex Slivkins, Sid Suri, Mihaela van der Schaar, Jake Abernethy, Rafael Frongillo, Kuan-Ta Chen, Kwei-Jay Lin, Shahin Jabbari, Bo Waggoner, Yu Zhang, and Tsung-Hsiang Chang. I have learned so much from you.

Thanks to Mike, Shadi, Thibaut, Tom, and Bo for the fun weekly boardgame nights for the past three years. Thanks to Jerry and Lilian for being the best roommates and giving me an unforgettable year living in LA. To Riva, Tsung-Hsiang, Kay, Livia, and Yu-Ying, I am so grateful to always have you guys around. Thanks to the folks at Harvard EconCS, UCLA CS, and UCLA TSA. It is really nice to know all of you during my PhD journey.

Thanks to the UCLA CS department staff, especially Edna Todd, Steve Arbuckle, and Craig Jesson, for helping me remotely dealing with various logistics.

Thanks to my family for the endless support.

VITA

2001 – 2005	B.S. in Computer Science and Information Engineering B.S. in Physics National Taiwan University
2005 – 2007	M.S. in Computer Science and Information Engineering National Taiwan University

PUBLICATIONS

Low-Cost Learning via Active Data Procurement. Jacob Abernethy, Yiling Chen, Chien-Ju Ho, and Bo Waggoner. In the Sixteenth ACM Conference on Economics and Computation (EC), 2015

Incentivizing High Quality Crowdwork. Chien-Ju Ho, Aleksandrs Slivkins, Siddharth Suri, and Jennifer Wortman Vaughan. In the Twenty-Fourth International World Wide Web Conference (WWW), 2015

Adaptive Contract Design for Crowdsourcing Markets: Bandit Algorithms for Repeated Principal-Agent Problems. Chien-Ju Ho, Aleksandrs Slivkins, and Jennifer Wortman Vaughan. In the Fifteenth ACM Conference on Economics and Computation (EC), 2014

Adaptive Task Assignment for Crowdsourced Classification. Chien-Ju Ho, Shahin Jabbari, and Jennifer Wortman Vaughan. In the Thirtieth International Conference on Machine Learning (ICML), 2013

Online Task Assignment in Crowdsourcing Markets. Chien-Ju Ho and Jennifer Wortman Vaughan.

In the Twenty-Sixth Conference on Artificial Intelligence (AAAI), 2012

Towards Social Norm Design for Crowdsourcing Markets. Chien-Ju Ho, Yu Zhang, Jennifer Wortman Vaughan, and Mihaela van der Schaar. In the Fourth Human Computation Workshop (HCOMP), 2012

KissKissBan: A Competitive Human Computation Game for Image Annotation (Short Paper). Chien-Ju Ho, Tao-Hsuan Chang, Jong-Chuan Lee, Jane Yung-jen Hsu, Kuan-Ta Chen. In the ACM SIGKDD Workshop on Human Computation (HCOMP), 2009

On Formal Models for Social Verification. Chien-Ju Ho, Kuan-Ta Chen. In the ACM SIGKDD Workshop on Human Computation (HCOMP), 2009

PhotoSlap: A Multi-player Online Game for Semantic Annotation Chien-Ju Ho, Tsung-Hsiang Chang, Jane Yung-jen Hsu. In the Twenty-Second Conference on Artificial Intelligence (AAAI), 2007

The PhotoSlap Game: Play to Annotate (Intelligent System Demo). Tsung-Hsiang Chang, Chien-Ju Ho, Jane Yung-jen Hsu. In the Twenty-Second Conference on Artificial Intelligence (AAAI), 2007

CHAPTER 1

Introduction

The rapid development of computer technology has helped humans accomplish various tasks in everyday life, including filtering spam emails and finding the shortest driving directions. However, some tasks, such as image tagging and language translation, still pose great challenges for computers despite they are solvable by humans. In this dissertation, we study the design and analysis of crowdsourcing mechanisms, which utilize the intelligence from the human crowd to solve machine-intractable problems.

Background

Crowdsourcing, a term coined by Jeff Howe and Mark Robinson in the Wired Magazine in 2005, is the process of *outsourcing* tasks to a *crowd* of people. Despite the invention of the term is relatively new, the idea of utilizing the crowd to solve problems has appeared way before the Internet age. For instance, in 1714, the British government offered a monetary prize, the Longitude Prize, to the person who can provide the solution of measuring a ship's longitude. In 1783, King Louis XVI offered an award to the person who can produce alkali from sea salt with the most economic way. In 1884, the Oxford English Dictionary relied on around 800 volunteers to help categorize words.

Nowadays, crowdsourcing has transfered mainly to the online environment, since the Internet makes it easy to engage people to work together. Indeed, crowdsourcing has gained great success in this Internet age. For example, Wikipedia becomes the largest encyclopedia in the world with the help from online users to edit the articles. Online Q&A forums, such as Yahoo! Answers and Quora, utilize the intelligence of the crowd to help answer questions from other online users. Galaxy Zoo encourages online users to help contribute on science projects, such as classifying galaxies from

telescope images.

With the increasing popularity of crowdsourcing, various mechanisms have been developed to incentivize online users to contribute their effort and to aggregate their contribution. In Games with A Purpose [111], developers design online games to attract online players to play games while helping solve various hard real-world problems, such as gene folding [27] and image tagging [110]. reCAPTCHA [112] takes advantage of human effort in solving CAPTHCAs, which are distorted images used to identify whether the solver is a human or a robot, and has completed digitizing the archives of The New York Times and books from Google Books. Prediction markets, which allow users to buy or sell securities on whether future events will happen, have shown to be successful in predicting various events, such as Presidential elections [13] and Oscar winners.[1]

In addition to the above mechanisms, crowdsourcing markets are possibly the one with the most general purpose. In crowdsourcing markets, *requesters* can post almost any tasks online along with the payments to *workers* who complete the tasks. A typical task might involve tagging an image, transcribing an audio message, or even designing a website logo. As a result of the flexibility, crowdsourcing markets are shown to be useful in various scenarios, including conducting user studies [77], running behavioral experiments [66, 88], collecting data [65, 113], testing or even building business applications [5, 99]. Moreover, a great number of companies and services are created to fulfill the increasing demand of crowdsourcing services. Examples include Amazon Mechanical Turk, Elance-oDesk, CrowdFlower, 99designs, TopCoder, and more. With this popularity, researchers and companies are eager to understand how to better design crowdsourcing mechanisms and better utilize the wisdom of crowd.

Main Challenges

Crowdsourcing has already shown to be effective at engaging people to work. However, how to better design and utilize crowdsourcing mechanisms is still not well understood. In particular, most of the crowdsourcing mechanisms suffer from the prevalence of low quality crowdwork.

[1] See the website `http://www.predictwise.com` for a wide range of event predictions using prediction markets.

The main challenges come from the human participation. The success of crowdsourcing depends critically on whether we can obtain reliable information from participating users. More specifically, humans might make mistakes and have different skills, and therefore we cannot assume all the user contributions are reliable. We need to develop algorithms to aggregate noisy contributions from users and further learn how to assign tasks to users with right skills. In addition, humans are often self-interested and take actions to maximize their own benefits. We need to design mechanisms to provide incentives and motivate online users to take actions which are desirable for the platforms. Furthermore, humans might not always behave as the theory predicts. We need to better understand humans really behave in the real systems.

This dissertation addresses these challenges by combining theoretical analysis and empirical experiments. Building on techniques from machine learning, game theory, and online optimization, we propose a series of algorithms and frameworks which help obtain high quality contributions from online users. In addition, we conduct online behavioral experiments to understand how humans behave in the real systems and develop a more realistic user model with the support of empirical evidence.

Related Literature

There has been a growing literature on techniques to boost the quality of crowdwork. Here we briefly describe the main research directions. A more thorough literature review and the highlights of how our work advances the state of the art are described in each chapter.

First, there are a large number of research projects focusing on aggregating noisy contribution from multiple users by applying various machine learning techniques, such as Bayesian learning [115], message passing [73], minimax entropy [120], spectral methods [76], and variational inference [86]. In this dissertation (see Chapter 2), we not only study the aggregation of the noisy user contribution but also explore how to assign suitable tasks to workers with the right skill sets.

In addition to aggregating noisy information, researchers have also examined the design of incentive mechanisms to encourage high-quality crowdwork. For example, Jain et al. [70] explore ways in which to award virtual points to users in online question-and-answer forums to improve

the quality of answers. Ghosh and Hummel [47, 48] and Ghosh and McAfee [50] study how to distribute user generated content (e.g., Youtube videos) to users to encourage the production of high-quality internet content by people who are motivated by attention. In this dissertation (see Chapter 3 and 4), we focus on the use of two different incentives, including reputations and performance-based payments, and show how to (near-)optimally choose the design parameters.

There has also been empirical work examining how workers' behavior varies based on the incentives offered in crowdsourcing markets. However, these studies often produced mixed and somewhat contradictory recommendations. Take the use of performance-based payments for example, Harris [57] and Yin et al. [117] suggested that performance-based payments can improve work quality, while Shaw et al. [102] found no improvement and Yin et al. [116] found no difference in quality when varying bonus size. In this dissertation (see Chapter 5), we conduct a comprehensive set of empirical experiments to resolve the contradiction. We empirically examine whether, when, why, and where do workers respond to performance-based financial incentives. Moreover, we propose a novel but simple behavioral model which is consistent with the empirical observation from our work and previous works.

1.1 Overview of this Dissertation

In this dissertation, we explore the design and analysis of crowdsourcing mechanisms. We approach the design problem in three interleaving directions, as depicted in Figure 1.1, with a common goal of obtaining high quality information from users.

In Chapter 2, we first assume workers are non-strategic, but have different skill levels and might provide noisy information. We study how to assign suitable tasks to online arriving workers while aggregating their noisy contribution. In Chapter 3 and Chapter 4, we consider the setting in which workers are strategic and care about their own payoff. We explore how to incentivize workers to contribute high quality information with two types of incentives, reputations and contracts (i.e., performance-based payments). In Chapter 5, we take a step back and think about whether we make the right assumptions about how workers behave. In particular, we run behavioral experiments on Amazon Mechanical Turk to understand how workers respond to performance-based payments

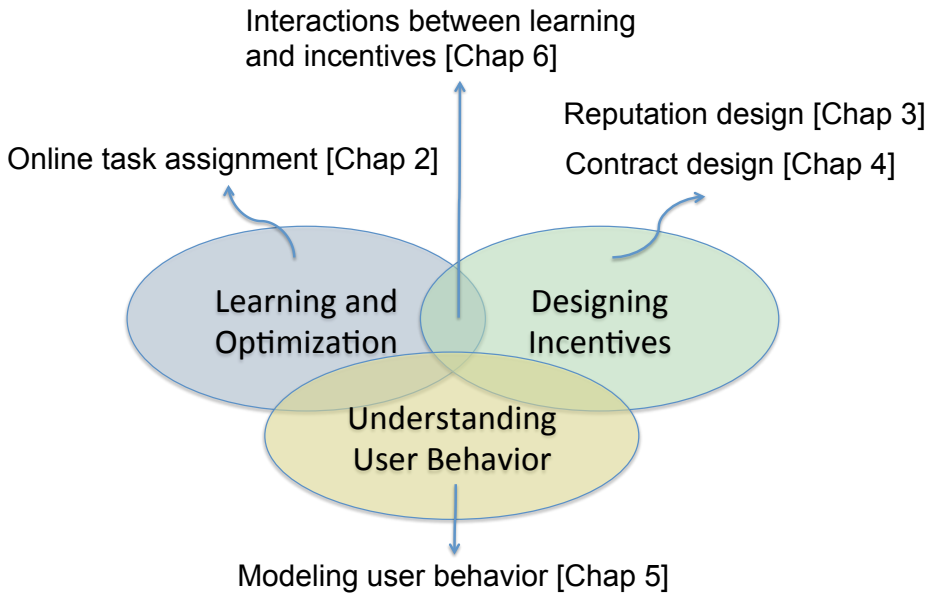


Figure 1.1: The overview of this dissertation.

in real-world scenarios. We then develop a more realistic worker models based on our empirical observation. Finally, in Chapter 6, we explore the setting in which we need to consider the interactions between learning and incentives in crowdsourcing. In particular, we study the problem of procuring data points from strategic users with the goal of solving a machine learning problem.

Below we give high level descriptions of each problem we study and highlight our contributions.

1.1.1 Online Task Assignment

Crowdsourcing markets rely on workers to solve a variety of tasks. However, workers have different skill levels over different tasks and may not be able to produce high quality work for all tasks. Therefore, it is important to design algorithms to learn workers' skills and smartly assign suitable tasks to workers. In Chapter 2, we explore the problem of assigning heterogeneous tasks to workers with different, unknown skill sets. We consider the setting in which workers arrive online one at a time and can only complete a limited number of tasks. The goal is to maximize the utility we get from completed tasks and minimize the budget we spend on hiring workers.

To achieve the goal, we first formulate the problem in the online primal-dual framework. We then propose task assignment algorithms which learn workers' skill levels through historical records

and assign tasks to workers based on what we learned. We prove that our algorithms can achieve near-optimal performance compared to the oracle algorithm which has access to the unknown worker skills and worker arriving sequences. Experiment results also demonstrate our algorithms outperform greedy algorithms consistently.

This chapter is based on joint work with Shahin Jabbari and Jennifer Wortman Vaughan [59, 61].

1.1.2 Designing Incentives - Reputation Systems

Online workers not only have different skills but also are self-interested. Workers care about their own benefits and might choose to exert low efforts on tasks to maximize their utilities. Furthermore, the requesters might also be strategic and might not always honestly pay workers as they promised. Therefore, we need to design appropriate incentive mechanisms to encourage requesters and workers to take desired actions. In Chapter 3, we study the use of reputations as incentives. We focus on reputation design in two-sided crowdsourcing markets, in which we attach reputation values both to requesters who publish tasks and to workers who accept tasks.

We propose a framework for designing optimal reputation systems, which not only encourages workers to exert high effort and requesters to honestly pay workers but also maximizes the revenue of the platform designer. We illustrate the use of this technique to derive the optimal reputation design from within the natural class of threshold-based social strategies paired with maximum punishment reputation update rules, and show that the optimal norm in this class is simple to implement and understand. We believe the framework and techniques will provide the necessary groundwork for future progress towards a complete, robust theory of incentive design for crowdsourcing.

This chapter is based on joint work with Yu Zhang, Jennifer Wortman Vaughan, and Mihaela van der Schaar [60].

1.1.3 Designing Incentives - Performance Based Payments

In addition to reputation systems, we also study the use of performance-based payments to encourage high-quality contributions from workers. In Chapter 4, we explore how to set optimal crowdsourcing contracts, i.e., performance-based payments, in which workers' payments depend on the quality of

their work. We consider the setting in which workers aim to maximize their own utilities and put in effort based on the contracts we post. Furthermore, we have no knowledge of workers' costs for exerting effort and their abilities for the tasks in advance. The goal is to learn the contract which optimizes the requester's utility by observing workers' historical performance over time.

We extend the standard principal-agent model from economic theory to a multi-round online model. We then treat this problem as a *multi-armed bandit problem*, with each "arm" representing a potential contract. To cope with the large (and in fact, infinite) number of arms, we propose a new algorithm, AgnosticZooming, which adaptively discretizes the contract space so that more promising regions of the contract space are eventually discretized more finely. We show that our algorithm performs almost as well as an oracle algorithm which has access to full information despite only observing limited information from users.

This chapter is based on joint work with Aleksandrs Slivkins and Jennifer Wortman Vaughan [62]

1.1.4 Modeling User Behavior

In Chapter 4, our results build on standard worker models in economics. In particular, we assume workers are rational and aim to maximize their expected payments minus their expected costs. However, this assumption might not always be true. In fact, several empirical studies have given mixed and conflicting results. In this chapter, our goal is to resolve the controversy in previous works. Furthermore, we aim to develop more realistic worker models, which can be used as foundations for developing more practical theories.

We empirically explore the causal effects of financial incentives on the quality of crowdwork. In particular, we study how online workers react to different performance-based payments. We conduct a comprehensive set of experiments on Amazon Mechanical Turk and collect results from more than 2,000 workers. With the empirical evidence, we provide a coherent explanation of our results and the seemingly conflicting results from previous work. Moreover, we develop a worker behavior model which introduces the concept of *workers' priors* into the standard economic model. Workers' priors describe workers' beliefs on their subjective probabilities of getting different payments given their performance. This simple model explains all of our empirical results and the results of previous

studies. We bridge the gap between our previous theory work on adaptive contract design and this empirical worker behavior model. We show that our theory still applies with this updated worker model and is robust to a variety of worker models.

This chapter is based on joint work with Aleksandrs Slivkins, Siddharth Suri, and Jennifer Wortman Vaughan [63]

1.1.5 Interactions Between Learning and Incentives

In Chapter 6, we consider the setting in which we need to jointly consider learning from the contribution of the crowd and the incentives of the crowd. In particular, we study the following problem: In a world where data is held by self-interested agents with heterogeneous costs for providing it, and in particular when these costs may be arbitrarily correlated with the underlying data, how can we design mechanisms that are incentive compatible, have robust learning guarantees, and optimize the cost-efficiency trade-offs inherent in the learning problem?

To address this problem, we propose algorithms which identify the *importance* of each data point and decide the payment to offer to each user. We show how to convert a large class of no-regret algorithms into online posted-price and learning mechanisms. Our results parallel classic sample complexity guarantees, but with the key resource constraint being money rather than quantity of data available. With a budget constraint, we give robust risk (predictive error) bounds on the order of one over the square root of the budget. In many cases our guarantees are significantly better due to an active-learning approach that leverages correlations between costs and data.

This chapter is based on joint work with Jacob Abernethy, Yiling Chen, and Bo Waggoner [1].

CHAPTER 2

Online Task Assignment

2.1 Overview

The availability of diverse workers willing to complete tasks inexpensively makes crowdsourcing markets appealing as tools for collecting data [113]. Classification tasks, in which workers are asked to provide a binary label for an instance, are among the most common tasks posted [67]. Unfortunately, due to a mix of human error, carelessness, and fraud — the existence of spammy workers on Mechanical Turk is widely acknowledged — the data collected is often noisy [77, 114]. For classification tasks, this problem can be overcome by collecting labels for each instance from multiple workers and combining these to infer the true label. Indeed, much recent research has focused on developing algorithms for combining labels from heterogeneous labelers [32, 68]. However, this research has typically focused on the inference problem, sidestepping the question of how to assign workers to tasks by assuming that the learner has no control over the assignment. One exception is the work of Karger et al. [73], who focus on the situation in which all tasks are homogeneous (i.e., equally difficult and not requiring specialized skills), in which case they show that it is not possible to do better than using a random assignment.

One might expect the assignment to matter more when the tasks are heterogeneous. Classifying images of dogs versus images of cats is likely easier for the average worker than classifying images of Welsh Terriers versus images of Airedale Terriers. It might be necessary to assign more workers to tasks of the latter type to produce high confidence labels. The assignment can also be important when tasks require specialized skills. A worker who knows little about dogs may not be able to produce high quality labels for the Terrier task, but may have skills that are applicable elsewhere.

In this chapter, we investigate the problem of task assignment and label inference for heteroge-

neous classification tasks. In our model, a task requester has a set of tasks, each of which consists of an instance for which he would like to infer a binary label. Workers arrive online. The learner must decide which tasks to assign to each worker, and then use the noisy labels produced by the workers to infer the true label for each task. The goal of the learner is to output a set of labels with sufficiently low error while requesting as few labels from workers as possible. Building on online primal-dual methods [18], we propose an exploration-exploitation algorithm that is provably competitive with an optimal offline algorithm that has knowledge of the sequence of workers and their skills in advance. We then evaluate this algorithm in a variety of experiments on synthetic data and show that adaptively allocating tasks helps when the worker distribution is diverse or the tasks are heterogeneous.

2.1.1 Related Work

The work in this chapter is mostly closely related to that of Karger et al. [73]. Karger et al. introduced a model in which a requester has a set of homogeneous labeling tasks he must assign to workers who arrive online. They proposed an assignment algorithm based on random graph generation and a message-passing inference algorithm inspired by belief propagation, and showed that their technique is order-optimal in terms of labeling budget. In particular, let p_j be the probability that worker j completes any given task correctly and $q = \mathbb{E}[(2p_j - 1)^2]$, where the expectation is over the choice of a random worker j. They proved that their algorithm requires $O((1/q)\log(1/\epsilon))$ labels per task to achieve error less than ϵ in the limit as the numbers of tasks and workers go to infinity. They also showed that adaptively assigning tasks does not help in their setting, in that $\Omega((1/q)\log(1/\epsilon))$ labels are still needed in general.

We generalize this model to allow heterogeneous tasks, so that the probability that worker j completes a task correctly may depend on the particular task. In this generalized setting, assigning tasks adaptively can provide an advantage both in theory and in practice.

Our techniques build on the online primal-dual framework, which has been used to analyze online optimization problems ranging from the adwords problem [19, 38] to network optimization [4] and paging [12].

Repeated labeling has received considerable empirical attention, dating back to the EM-based algorithm of Dawid and Skene [31]. Sheng et al. [103] considered a setting in which every worker is correct on every task with the same probability, and empirically evaluated how much repeated labeling helps. Ipeirotis et al. [68] extended this idea to heterogeneous workers and provided an algorithm to simultaneously estimate workers' quality and true task labels. More recently, there has been work showing that label inference can be improved by first estimating parameters of the structure underlying the labeling process using techniques such as Bayesian learning [115], minimax entropy [120], and variational inference [86].

On the theoretical side, there have been several results on learning a binary classifier using labeled data contributed by multiple teachers, each of which labels instances according to his own fixed labeling function [28, 29, 32]. These require PAC-style assumptions and focus on filtering out low quality workers. Tran-Thanh et al. [109] used ideas from the multi-armed bandit literature to assign tasks. Bandit ideas cannot be applied in our setting without further assumptions since the reward corresponding to an assignment depends on whether the worker's label is correct, which cannot be inferred until the task has been assigned to others.

Ghosh et al. [52] studied a model similar to that of Karger et al., also with homogeneous tasks, and used eigenvalue decomposition to estimate each worker's quality. Their bounds depend on a quantity essentially identical to the quantity q defined above, which they refer to as the population's *average competence*. A similar quantity plays a role in our analysis.

2.2 The Model

In our model, a task requester has a set of n tasks, indexed $1, \cdots, n$. Each task is a binary classification problem. The true label of task i, denoted ℓ_i, is either 1 or -1, and is unknown to the requester.

Workers arrive online. When worker j arrives, she announces the maximum number of tasks that she is willing to complete, her *capacity*, M_j. No other information is known about each worker when she arrives.

Each worker j has a *skill level*, $p_{i,j} \in [0, 1]$, for each task i. If the algorithm assigns worker j to task i, the worker will produce a label $\ell_{i,j}$ such that $\ell_{i,j} = \ell_i$ with probability $p_{i,j}$ and $\ell_{i,j} = -\ell_i$ with probability $1 - p_{i,j}$, independent of all other labels. The algorithm may assign worker j up to M_j tasks, and may observe her output on each task before deciding whether to assign her to another or move on, but once the algorithm moves on, it cannot access the worker again. This is meant to reflect that crowdsourced workers are neither identifiable nor persistent, so we cannot hope to identify and later reuse highly skilled workers.

Several of our results depend on the quantity $q_{i,j} = (2p_{i,j} - 1)^2$. Intuitively, when this quantity is close to 1, the label of worker j on task i will be informative; when it is close to 0, the label will be random noise.

To model the fact that the requester cannot wait arbitrarily long, we assume that he can only assign tasks to the first m workers who arrive, for some known m. We therefore index workers $1, \cdots, m$. Later we consider an additional γm workers who are used for exploration.

In addition to assigning tasks to workers, the learning algorithm must produce a final estimate $\hat{\ell}_i$ for the label ℓ_i of each task i based on the labels provided by the workers. The goal of the learner is to produce estimates that are correct with high probability while querying workers for as few labels as possible.

Task structure: A clever learning algorithm should infer the worker skill levels $p_{i,j}$ and assign workers to tasks at which they excel. If the skills are arbitrary, then the learner cannot infer them without assigning every worker to every task. Therefore, it is necessary to assume that the $p_{i,j}$ values exhibit some structure. Karger et al. [73] assume that all tasks are identical, i.e., $p_{i,j} = p_{i',j}$ for all j and all i and i'. We consider a more general setting in which the tasks can be divided into T *types*, and assume only that $p_{i,j} = p_{i',j}$ if i and i' are of the same type.

Gold standard tasks: As is common in the literature [94], we assume that the learner has access to "gold standard" tasks of each task type.[1] These are instances for which the learner knows the true label a priori. They can be assigned in order to estimate the $p_{i,j}$ values. Of course the algorithm must pay for these "pure exploration" assignments.

[1] If gold standard tasks are not available, they can be created by assigning a small set of tasks to many workers.

Random permutation model: We analyze our algorithm in the random permutation model as in Devanur and Hayes [37]. The capacities M_j and skills $p_{i,j}$ of each worker j may be chosen adversarially, as long as the assumptions on task structure are satisfied. However, the arrival order is randomly permuted. Since only the order of workers is randomized, the offline optimal allocation is well-defined.

Competitive ratio: To evaluate our algorithm, we use the notion of *competitive ratio*, which is an upper bound on the ratio between the number of labels requested by the algorithm and the number requested by an *optimal offline algorithm* which has access to all worker capacities and skill levels, but must still assign enough workers to each task to obtain a high-confidence guess for the task's label. The optimal offline algorithm is discussed in Sections 2.3 and 2.4.

2.3 An Offline Problem

To gain intuition, we first consider a simplified offline version of our problem in which the learner is provided with a full description of the sequence of m workers who will arrive, including the skill levels $p_{i,j}$ and capacities M_j for all i and j. The learner must decide which tasks to assign to each worker and then infer the task labels. We discuss the inference problem first.

2.3.1 Aggregating Workers' Labels

Suppose that the learner has already assigned tasks to workers and observed the workers' labels for these tasks. How should the learner aggregate this information to infer the true label for each task?

We consider aggregation methods that take a weighted vote of the workers' labels. Fix a task i. Let J_i denote the set of workers assigned to this task. We consider methods that set $\hat{\ell}_i = \operatorname{sign}(\sum_{j \in J_i} w_{i,j} \ell_{i,j})$ for some set of weights $\{w_{i,j}\}$. The following lemma shows that this technique with weights $w_{i,j} = 2p_{i,j} - 1$ is guaranteed to achieve a low error if enough high quality workers are queried. Recall that $q_{i,j} = (2p_{i,j} - 1)^2$.

Lemma 2.3.1. *Let $\hat{\ell}_i = \operatorname{sign}(\sum_{j \in J_i} w_{i,j} \ell_{i,j})$ for some set of weights $\{w_{i,j}\}$. Then $\hat{\ell}_i \neq \ell_i$ with probability at most $e^{-\frac{1}{2}(\sum_{j \in J_i} w_{i,j}(2p_{i,j}-1))^2 / \sum_{j \in J_i} w_{i,j}^2}$. This bound is minimized when $w_{i,j} \propto (2p_{i,j} -$*

1), *in which case the probability that* $\hat{\ell}_i \neq \ell_i$ *is at most* $e^{-\frac{1}{2}\sum_{j \in J_i} q_{i,j}}$.

The proof, which uses a simple application of Hoeffding's inequality, is in the appendix.[2] This tells us that to guarantee that we make an error with probability less than ϵ on a task i, it is sufficient to select a set of labelers J_i such that $\Sigma_{j \in J_i} q_{i,j} \geq 2\ln(1/\epsilon)$ and aggregate labels by setting $\hat{\ell}_i = \text{sign}(\sum_{j \in J_i}(2p_{i,j} - 1)\ell_{i,j})$.

One might ask if it is possible to guarantee an error of ϵ with fewer labels. In some cases, it is; if there exists an i and j such that $p_{i,j} = q_{i,j} = 1$, then one can achieve zero error with only a single label. However, in some cases this method is optimal. For this reason, we restrict our attention to algorithms that query subsets J_i such that $\Sigma_{j \in J_i} q_{i,j} \geq 2\ln(1/\epsilon)$. We use the shorthand $C_\epsilon = 2\ln(1/\epsilon)$.

2.3.2 Integer Programming Formulation

There is a significant benefit that comes from restricting attention to algorithms of the form described above. Let $y_{i,j}$ be a variable that is 1 if task i is assigned to worker j and 0 otherwise. The requirement that $\sum_{j \in J_i} q_{i,j} \geq C_\epsilon$ can be expressed as a linear constraint of these variables. This would not be possible using unweighted majority voting to aggregate labels; weighting by $2p_{i,j} - 1$ is key. This allows us to express the optimal offline assignment strategy as an integer linear program (IP), with variables $y_{i,j}$ for each (i, j):

$$\begin{aligned} \min \ & \Sigma_{i=1}^{n} \Sigma_{j=1}^{m} y_{i,j} \\ \text{s.t.} \ & \Sigma_{i=1}^{n} y_{i,j} \leq M_j \ \ \forall j & (2.1) \\ & \Sigma_{j=1}^{m} q_{i,j} y_{i,j} \geq C_\epsilon \ \ \forall i & (2.2) \\ & y_{i,j} \in \{0, 1\} \ \ \forall (i, j). & (2.3) \end{aligned}$$

Constraint (2.1) guarantees that worker j does not exceed her capacity. Constraint (2.2) guarantees that aggregation will produce the correct label of each task with high probability. Constraint (2.3) implies that a task is either assigned to a worker or not.

[2] An appendix containing all omitted proofs and additional details can be found in the long version of this paper available on the authors' websites.

Note that there may not exist a feasible solution to this IP, in which case it would not be possible to guarantee a probability of error less than ϵ for all tasks using weighted majority voting. For most of this paper, we assume a feasible solution exists; the case in which one does not is discussed in Section 2.4.1.

For computational reasons, instead of working directly with this IP, we will work with a linear programming relaxation obtained by replacing the last constraint with $0 \leq y_{i,j} \leq 1 \ \forall(i,j)$; we will see below that this does not impact the solution too much.

2.3.3 Working with the Dual

Solving the linear program described above requires knowing the values $q_{i,j}$ for the full sequence of workers j up front. When we move to the online setting, it will be more convenient to work with the dual of the relaxed linear program, which can be written as follows, with dual variables x_i, z_j, and $t_{i,j}$ for all (i,j):

$$\begin{aligned} \max \ & C_\epsilon \Sigma_{i=1}^{n} x_i - \Sigma_{j=1}^{m} M_j z_j - \Sigma_{i=1}^{n} \Sigma_{j=1}^{m} t_{i,j} \\ \text{s.t.} \ & 1 - q_{i,j} x_i + z_j + t_{i,j} \geq 0 \ \ \forall(i,j) \\ & x_i, z_j, t_{i,j} \geq 0 \ \ \forall(i,j). \end{aligned}$$

We refer to x_i as the *task weight* for i, and define the *task value* of worker j on task i as $v_{i,j} = q_{i,j} x_i - 1$.

Suppose that we were given access to the task weights x_i for each task i and the values $q_{i,j}$. (We will discuss how to approximate these values later.) Then we could use the following algorithm to approximate the optimal primal solution. Note that to run this algorithm, it is not necessary to have access to all $q_{i,j}$ values at once; we only need information about worker j when it comes time to assign tasks to this worker. This is the advantage of working with the dual.

The following theorem shows that this algorithm produces a near-optimal primal solution to our original IP when given as input the optimal dual solution for the relaxed LP. The condition that $q_{i,j} x_i^* \neq q_{i',j} x_{i'}^*$ for all $i \neq i'$ is needed for technical reasons, but can be relaxed by adding small

Algorithm 1: Primal Approximation Algorithm

Input: Values x_i and $q_{i,j}$ for all (i, j)

For every worker $j \in \{1, \ldots, m\}$, compute the task values, $v_{i,j} = q_{i,j}x_i - 1$, for all tasks i. Let n_j be the number of tasks i such that $v_{i,j} \geq 0$. If $n_j \leq M_j$, then set $y_{ij} \leftarrow 1$ for all n_j tasks with non-negative task value. Otherwise, set $y_{i,j} \leftarrow 1$ for the M_j tasks with highest task value. Set $y_{i,j} \leftarrow 0$ for all other tasks.

random perturbations to $q_{i,j}$ values as in Devanur et al. [38].[3] For the rest of the paper, we assume that perturbations have been added and that the condition above holds. Call this the "perturbation assumption." In our final algorithm, we will perturb our estimates of the $q_{i,j}$ values for this reason.

Theorem 2.3.2. *Let $\mathbf{y}^*$ be the primal optimal of the IP and $\mathbf{x}^*$ be the dual optimal of the relaxed formulation. Let $\mathbf{y}$ be the output of the Primal Approximation Algorithm given input $\mathbf{x}^*$ and the true values $\mathbf{q}$. Then $\mathbf{y}$ is feasible in the IP, and under the perturbation assumption, $\Sigma_{i=1}^{n}\Sigma_{j=1}^{m} y_{i,j} - \Sigma_{i=1}^{n}\Sigma_{j=1}^{m} y_{i,j}^* \leq \min(m, n)$.*

The proof shows that the $y_{i,j}$ values assigned by the Primal Approximation Algorithm differ from the optimal solution of the relaxed LP for at most $\min(m, n)$ pairs (i, j), and that this implies the result.

2.4 Moving to the Online Setting

We have shown that, given access to $\mathbf{q}$ and the optimal task weights $\mathbf{x}^*$, the Primal Approximation Algorithm generates an assignment which is close to the optimal solution of the IP in Section 2.3.2. However, in the online problem that we initially set out to solve, these values are unknown. In this section, we provide methods for estimating these quantities and incorporate these into an algorithm for the online problem.

Our online algorithm combines two varieties of exploration. First, we use exploration to estimate the optimal task weights $\mathbf{x}^*$. To do this, we hire an additional γm workers on top of the m workers

[3] Adding noise will introduce an error, but this error can be made arbitrarily small when C_ϵ is large. To simplify presentation, we do no include the error in our discussion.

we originally planned to hire, for some $\gamma > 0$, and "observe" their $q_{i,j}$ values. (We will actually only estimate these values; see below.) Then, by treating these γm workers as a random sample of the population (which they effectively are under the random permutation model), we can apply online primal-dual methods and obtain estimates of the optimal task weights. These estimates can then be fed to the Primal Approximation Algorithm in order to determine assignments for the remaining m workers, as described in Section 2.4.1.

The second variety is used to estimate workers' skill levels. Each time a new worker arrives (including the γm extras), we require her to complete a set of gold standard tasks of each task type. Based on the labels she provides, we estimate her skill levels $p_{i,j}$ and use these to estimate the $q_{i,j}$ values. The impact of these estimates on performance is discussed in Section 2.4.2.

If we require each worker to complete s gold standard tasks, and we hire an extra γm workers, we need to pay for an extra $(1+\gamma)ms$ assignments beyond those made by the Primal Approximation Algorithm. We precisely quantify how the number of assignments compares with the offline optimal in Section 2.4.3.

2.4.1 Estimating the Task Weights

In this section, we focus on the estimation of task weights in a simplified setting in which we can observe the quality of each worker as she arrives. To estimate the task weights, we borrow an idea from the literature on the online primal-dual framework. We use an initial sampling phase in which we hire γm workers in addition to the primary m workers, for some γ. We observe their skill levels and treat the distribution over skills of the sampled workers as an estimate of the distribution of skills of the m primary workers. Given the $q_{i,j}$ values from the sampled γm workers, we can solve an alternative linear programming problem, which is the same as our relaxed offline linear programming problem, except that m is replaced by γm and C_ϵ is replaced by γC_ϵ. Let $\hat{\mathbf{x}}^*$ be the optimal task weights in this "sampled LP" problem. We show that if ϵ is small enough, running the Primal Approximation Algorithm using $\hat{\mathbf{x}}^*$ and $\mathbf{q}$ yields a near-optimal solution, with a number of assignments close to optimal, and a prediction error close to ϵ after aggregation.

Theorem 2.4.1. *For any $\epsilon, \delta \in (0, 1/2)$, for any $\gamma = \ell/m$ with $\ell \in \{1, 2, \cdots, m\}$ and $\gamma \in [1/C_\epsilon, 1]$,*

let $\hat{\boldsymbol{y}}^{s,*}$ and $\hat{\boldsymbol{x}}^*$ be the primal and dual optimal solutions of the sampled LP with parameters ϵ and γ. Let $\hat{\boldsymbol{y}}^*$ be the output of the Primal Approximation Algorithm with inputs $\hat{\boldsymbol{x}}^*$ and $\boldsymbol{q}$, and let $\bar{\boldsymbol{y}}^*$ be the optimal assignment of the relaxed offline formulation with parameter ϵ. Let $q_{min} = \min_{(i,j):\hat{y}^{s,*}_{i,j}>0} q_{i,j}$. Then under the perturbation assumption, with probability at least $1-\delta$,

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\hat{y}^*_{i,j} \leq \left(1+\frac{\min(m,n)}{q_{min}nC_\epsilon}+\frac{35\ln(2/\delta)}{q_{min}\sqrt{\gamma C_\epsilon}}\right)\sum_{i=1}^{n}\sum_{j=1}^{m}\bar{y}^*_{i,j}.$$

If the labels collected from the resulting assignment are used to estimate the task labels via weighted majority voting, the probability that any given task label is predicted incorrectly is no more than $\epsilon^{1-6\ln(2/\delta)/\sqrt{\gamma C_\epsilon}}$.

The requirement that $\gamma \geq 1/C_\epsilon$ stems from the fact that if C_ϵ is small, the total number of assignments will also be small, and the quality of the assignment is more sensitive to estimation errors. If C_ϵ is large, small estimation errors effect the assignment less and we can set the sampling ratio to a smaller value.

In the proof, we show that the gap between the objectives of the primal solution generated by the Primal Approximation Algorithm using $\hat{\mathbf{x}}^*$ and $\mathbf{q}$ and the corresponding dual solution is exactly the summation of $\hat{x}^*_i(C_\epsilon - \Sigma_{j=1}^m q_{i,j}\hat{y}^*_{i,j})$ over all tasks i, which is small if enough workers are sampled. By weak duality, the optimal number of assignments is between the primal and the dual objectives, so the primal solution output by the algorithm must be near-optimal.

A note on feasibility: We have implicitly assumed that the sampled LP is feasible. In practice, it may not be, or even if it is, there may exist tasks i such that $\min_{j:\hat{y}^{s,*}_{i,j}>0} q_{i,j}$ is very small, leading to a small value of q_{min}. If either of these things happen, the task requester may want to discard some of the tasks or lower his desired error, solve the sampled LP with these modified constraints, and continue from there, as there is no way to guarantee low error on all tasks.

2.4.2 Using Estimates of Skill Levels

We now discuss the effect of estimating worker skills. Given observations of the gold standard tasks of type τ that worker j completed, we can estimate $p_{i,j}$ for any task i of type τ as the fraction of these tasks she labeled correctly. The following lemma, follows from a straightforward application

of the Hoeffding bound; we state it here as it will be useful, but omit the proof.

Lemma 2.4.2. *For any worker j, for any task type τ, and for any $t, \delta \in (0, 1)$, suppose that worker j labels $\ln(2/\delta)/(2t^2)$ gold standard tasks of type τ. Then with probability at least $1 - \delta$, for all tasks i of type τ, if we set $\hat{p}_{i,j}$ to the fraction of gold standard tasks of type τ answered correctly then $|p_{i,j} - \hat{p}_{i,j}| \leq t$.*

This estimate of $p_{i,j}$ can then be used to derive an estimate for $q_{i,j}$, with error bounded as follows.

Lemma 2.4.3. *For any worker j and task i, if $\hat{p}_{i,j}$ is an estimate of $p_{i,j}$ such that $|p_{i,j} - \hat{p}_{i,j}| \leq t$, and $\hat{q}_{i,j}$ is set to $(2\hat{p}_{i,j} - 1)^2$, then $|q_{i,j} - \hat{q}_{i,j}| \leq 4t$.*

Of course the use of estimated values impacts performance. Consider the offline problem discussed in the previous section. One might hope that if we applied the Primal Approximation Algorithm using $\hat{\mathbf{q}}$, the number of assignments would be close to the number made using $\mathbf{q}$. Unfortunately, this is not true. Consider this toy example. Let $q_{i,1} = q_{i,2} = q_{i,3} = 1$ for all i, $q_{i,j} = 10^{-4}$ for all i and $j > 3$, and $M_j = n$ for all j. Set $\epsilon = 0.224$ so that $C_\epsilon \approx 3$. In the optimal solution, each task i should be assigned only to workers $1, 2$, and 3. If we underestimate the $q_{i,j}$ values, we could end up assigning each task to many more workers. This can be made arbitrarily bad.

To address this, instead of solving the relaxed offline formulation directly, we consider an alternative LP which is identical to the relaxed offline formulation, except that $\mathbf{q}$ is replaced with $\hat{\mathbf{q}}$ and C_ϵ is replaced with a smaller value $C_{\epsilon'}$ (corresponding to a higher allowable error ϵ'). We call this *the approximated LP*. We show that, if ϵ' is chosen properly, we can guarantee the optimal solution in the relaxed offline formulation is feasible in the approximated LP, so the optimal solution of the approximated LP will yield an assignment with fewer tasks assigned to workers than the optimal solution of the relaxed offline formulation, even though it is based on estimations.

To set ϵ', we assume the requester has a rough idea of how hard the tasks are and how inaccurate his estimates of worker skills are. The latter can be achieved by applying Lemma 2.4.2 and the union bound to find a value of t such that $|p_{i,j} - \hat{p}_{i,j}| \leq t$ for all (i, j) pairs with high probability,

and setting each $\hat{q}_{i,j} = (2\hat{p}_{i,j} - 1)^2$ as in Lemma 2.4.3. For the former, let $\bar{\mathbf{y}}^*$ be the optimal solution of the relaxed offline formulation. Define $\bar{q}_i^* = \Sigma_{j=1}^m q_{i,j}\bar{y}_{i,j}^* / \Sigma_{j=1}^m \bar{y}_{i,j}^*$. We assume that the requester can produce a value $\bar{q}_{min}^*$ such that $\bar{q}_{min}^* \leq \bar{q}_i^*$ for all i and then set $C_{\epsilon'} = 2\ln(1/\epsilon')$ where $\epsilon' = \epsilon^{1-4t/\bar{q}_{min}^*}$. If the requester doesn't have much information, he can conservatively set $\bar{q}_{min}^*$ much smaller than $\min_i\{\bar{q}_i^*\}$, but will both need more accurate estimates of $p_{i,j}$ and sacrifice some prediction accuracy.

Theorem 2.4.4. *Assume that we have access to a value $\bar{q}_{min}^*$ such that $\bar{q}_{min}^* \leq \bar{q}_i^*$ for all i and values $\hat{p}_{i,j}$ such that $|p_{i,j} - \hat{p}_{i,j}| \leq t$ for all (i,j) pairs for any known value $t < \bar{q}_{min}^*/4$. Then for any $\epsilon > 0$, the optimal solution of the approximated LP with parameter $\epsilon' = \epsilon^{1-4t/\bar{q}_{min}^*}$ and skill levels $\hat{q}_{i,j} = (2\hat{p}_{i,j} - 1)^2$ is no bigger than the optimal solution of the relaxed offline formulation with parameter ϵ and skill levels $q_{i,j}$.*

Of course this guarantee is not free. We pay the price of decreased prediction accuracy since we are using ϵ' in place of ϵ. We also pay when it comes time to aggregate the workers' labels, since we must now use $\hat{\mathbf{q}}$ in place of $\mathbf{q}$ when applying the weighted majority voting method described in Section 2.3.1. This is quantified in the following theorem. Note that this theorem applies to *any* feasible integer solution of the approximated LP and therefore also the best integer solution.

Theorem 2.4.5. *Assume again that we have access to a value $\bar{q}_{min}^*$ such that $\bar{q}_{min}^* \leq \bar{q}_i^*$ for all i and values $\hat{p}_{i,j}$ such that $|p_{i,j} - \hat{p}_{i,j}| \leq t$ for all (i,j) pairs for any known value $t < \bar{q}_{min}^*/4$. For any $\epsilon > 0$, let $\mathbf{y}$ be any feasible integer assignment of the approximated LP with parameter $\epsilon' = \epsilon^{1-4t/\bar{q}_{min}^*}$ and skill levels $\hat{q}_{i,j} = (2\hat{p}_{i,j} - 1)^2$. Let $J_i = \{j : y_{i,j} = 1\}$ denote the set of workers that are assigned to task i according to $\mathbf{y}$, and define $\hat{q}_i = \Sigma_{j \in J_i} q_{i,j} / |J_i|$. If the tasks are assigned according to $\mathbf{y}$ and the results aggregated using weighted majority voting with weights $w_{i,j} = 2\hat{p}_{i,j} - 1$, the error probability of the predicted task label for task i is bounded by $\epsilon^{(1-4t/\bar{q}_{min}^*)(1-4t/\hat{q}_i)}$.*

Theorem 2.4.5 tells us that if our estimates of the worker skills are accurate (i.e., if t is small), then our prediction error will be close to the error we would have achieved if we had known the true worker skills. How good the estimates need to be depends on the quality of the workers, as measured by $\bar{q}_{min}^*$ and $\hat{q}_i$. Intuitively, if $\bar{q}_{min}^*$ is small, there may exist some task i at which workers perform poorly in the optimal solution. In this case, the assignment will be very sensitive to the

value of ϵ' chosen, and it will be necessary to set ϵ' larger to guarantee that the true optimal solution is feasible in the approximated LP. If $\hat{q}_i$ is small, then a small amount of error in estimated worker quality would dramatically change the weights used in the weighted majority voting aggregation scheme.

2.4.3 Putting it All Together

We have separately considered relaxations of the task assignment and label inference problem in which the optimal task weights or worker skill levels are already known. We now put all these pieces together, give a combined algorithm, and state our main theorem.

Algorithm 2: Main Algorithm

Input: Values (ϵ, γ, s, and $\bar{q}^*_{min}$)

Hire γm preliminary workers.

for each preliminary worker **do**

  Assign s gold standard tasks of each task type.

  Calculate $\hat{q}_{i,j}$ values as in Section 2.4.2 and perturb with a negligible amount of noise.

end for

Calculate $C_{\epsilon'}$ and solve the sampled LP with $\hat{\mathbf{q}}$ to obtain primal $\mathbf{y}^s$ and dual $\hat{\mathbf{x}}^*$ as in Section 2.4.1.

for each worker $j \in \{1, ..., m\}$ **do**

  Assign s gold standard tasks of each task type.

  Calculate $\hat{q}_{i,j}$ values as in Section 2.4.2 and perturb with a negligible amount of noise.

  Run the Primal Approximation Algorithm with inputs $\hat{\mathbf{x}}^*$ and (perturbed) $\hat{\mathbf{q}}$ to $\mathbf{y}$.

  Assign worker j to all tasks i with $y_{i,j} = 1$.

end for

Aggregate the workers' labels using weighted majority voting as in Section 2.3.1.

The complete algorithm is stated in Algorithm 2, and its performance guarantee is given below. Recall that T is the number of task types. Again, we assume that the optimization problems are feasible.

Theorem 2.4.6. *For any $\epsilon, \delta \in (0, 1/2)$, for any $\gamma = \ell/m$ for an $\ell \in \{1, 2, ..., m\}$ such that $\gamma \in [1/C_\epsilon, 1]$, assume we have access to a value $\bar{q}^*_{min}$ satisfying the condition in Theorem 2.4.4, let s be any integer satisfying $s \geq 8\ln(4T(1+\gamma)m/\delta)/\bar{q}^{*2}_{min}$, and let $\epsilon' = \epsilon^{1-4\sqrt{\ln(4T(1+\gamma)m/\delta)/(2s)}/\bar{q}^*_{min}}$. Then under the perturbation assumption, with probability at least $1-\delta$, when the Main Algorithm is executed with input $(\epsilon, \gamma, s, \bar{q}^*_{min})$, the following two things hold:*

1) The number of assignments of to non-gold standard tasks is no more than

$$\left(1 + \frac{\min(m,n)}{\hat{q}_{min} n C_{\epsilon'}} + \frac{35\ln(4/\delta)}{\hat{q}_{min}\sqrt{\gamma C_{\epsilon'}}}\right)$$

times the optimal objective of the IP, where $\hat{q}_{min} = \min_{(i,j):y^s_{i,j}=1} \hat{q}^s_{i,j}$.

2) The probability that the aggregated label for each task i is incorrect is bounded by $\epsilon^{(1-l_1)(1-l_2)(1-l_{3,i})}$, where $l_1 = 4t/\bar{q}^*_{min}, l_2 = 6\ln(4/\delta)/\sqrt{\gamma C_{\epsilon'}}$, $l_{3,i} = 4t/\hat{q}_i$, and $t = \sqrt{\ln(4T(1+\gamma)m/\delta)/(2s)}$.

When ϵ is small, $C_{\epsilon'}$ is large, and l_2 approaches 0. The competitive ratio may shrink, but if ϵ is too small, $\hat{q}_{min}$ will shrink as well, and at some point the problem may become infeasible. When s is large, t is small, and so l_1 and $l_{3,i}$ approach 0, leading to error almost as low as if we knew the true **q** values, as we would expect.

2.5 Synthetic Experiments

In this section, we evaluate the performance of our algorithm through simulations on synthetically generated data. As a comparison, we also run the message-passing inference algorithm of Karger et al. [73] on the same data sets. As described in Section 4.1.1, Karger et al. use a non-adaptive, random assignment strategy in conjunction with this inference algorithm. We show that adaptively allocating tasks to workers using our algorithm can outperform random task assignment in settings in which *(i)* the worker distribution is diverse, or *(ii)* the set of tasks is heterogeneous.

We create $n = 1,000$ tasks and $m = 300$ workers with capacity $M_j = 200$ for all j, and vary the distribution over skill levels $p_{i,j}$. We would like to compare the error rates of the algorithms when given access to the same total number of labels. In the message-passing algorithm, we can directly set the number of labels by altering the number of assignments. In our algorithm, we change the parameter ϵ and observe the number of labels (including exploration) and the prediction error.

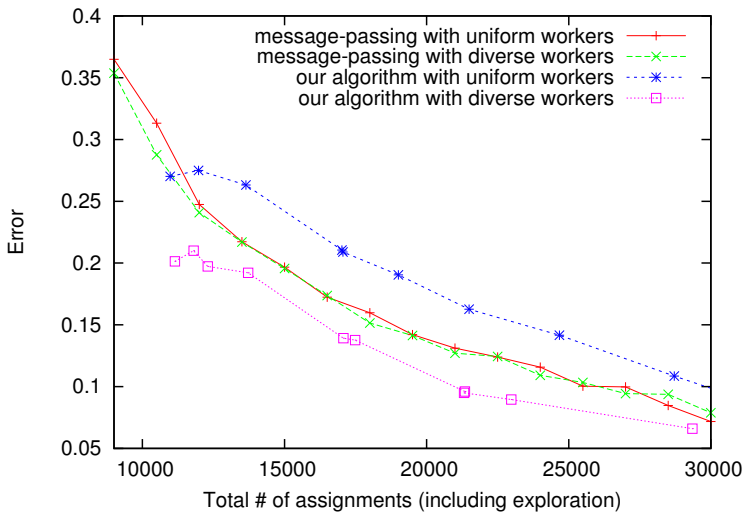


Figure 2.1: Uniform tasks with one or two worker types.

2.5.1 Worker Diversity

In their analysis, Karger et al. assume there is only one task type (that is, $p_{i,j} = p_{i',j}$ for all i, i', and j), and claim that in this setting adaptively assigning tasks does not yield much of an advantage. Our first experiment simulates this setting. We would like to see if our algorithm can perform better if the worker distribution is diverse, even though it requires some "pure exploration" — we need to pay each worker to complete the gold standard tasks, and we need to hire an extra γm workers to estimate the task weights.

For our algorithm, we set $\gamma = 0.3$ and sample 90 extra workers from the same distribution to learn task weights. Each worker is required to complete $s = 20$ gold standard tasks of each type when she arrives. These values were not optimized, and performance could likely be improved by tuning these parameters.

We examine two settings. In the first, every worker gives us a correct label with probability 0.6414 for all tasks. In the second, the population is 50% spammers and 50% hammers. The spammers give random answers, while the hammers answer correctly with probability 0.7. Note that these values are chosen such that $E[q_{i,j}] = E[(2p_{i,j} - 1)^2]$ is the same in both settings.

The results are shown in Figure 2.1. The performance of the message-passing algorithm is almost identical in the two settings. Our algorithm performs relatively poorly in the setting with

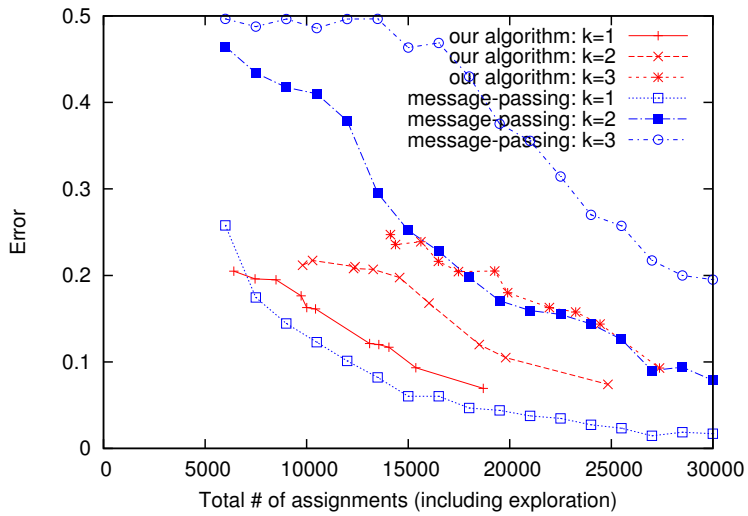


Figure 2.2: Heterogeneous tasks.

uniform workers since we can't benefit from adaptive assignments but still pay the exploration costs. However, our algorithm outperforms message passing in the setting with two types of workers, quickly learning not to assign any tasks to the spammers beyond those used for exploration.

2.5.2 Heterogeneous Tasks

We next examine a setting in which there are multiple types of tasks, and every worker is skilled at exactly one type. We generate k task types and k corresponding worker types, for $k = 1$, 2, and 3. Type α workers complete type α tasks correctly with probability 0.7, but other tasks correctly with probability 0.5.

For our algorithm, we set $\gamma = 0.3$. Each worker completes $s = 10$ gold standard tasks of each type.

The results are shown in Figure 2.2. Not surprisingly, since the message-passing algorithm does not attempt to match tasks to suitable workers, its performance degrades quickly when k grows. Since our algorithm attempts to find the best match between workers and tasks, the performance degrades much more slowly when k grows, even with the extra exploration costs.

2.6 Summary

We conclude this chapter by mentioning several extensions of our model. We have assumed that the requester pays the same price for any label. Our results can be extended to handle the case in which different workers charge different prices. Let $c_{i,j}$ denote the cost of obtaining a label for task i from worker j. The objective in the integer program would become $\Sigma_{i=1}^{n}\Sigma_{j=1}^{m} c_{i,j} y_{i,j}$. This is linear and the same techniques would apply.

The framework can also be extended to handle more intricate assumptions about the structure of tasks. We have assumed that there are T task types, with $p_{i,j} = p_{i',j}$ whenever i and i' are of the same type. However, this assumption is used only in the exploration phase in which workers' skills are estimated. While the amount of exploration required by the algorithm depends on the particular task structure assumed, the derivation of our algorithm and the general analysis are independent of the task structure.

CHAPTER 3

Designing Incentives - Reputation Systems

3.1 Overview

In principle, crowdsourcing markets could revolutionize the way in which projects are completed by giving individuals immediate access to large, diverse, and flexible pools of workers any time of day or night. In practice, they are plagued by workers who attempt to exert as little effort as possible [68, 77, 114], and requesters who advertise spammy tasks or deny workers payment for tasks completed.[1] This behavior must be discouraged in order for crowdsourcing markets to succeed.

In this chapter, we propose and study a class of incentive mechanisms based on *social norms* [72, 93]. A social norm consists of a set of rules that market participants are expected to follow, and a mechanism for updating participants' public reputations based on whether or not they do. For example, in a crowdsourcing market, workers may be encouraged to exert high effort only for tasks posted by requesters who have treated other workers fairly in the past. Our goal is to develop a formal framework to help platform designers (e.g., Amazon, in the case of Amazon Mechanical Turk) identify optimal social norms for their applications, taking into account parameters about the environment and the market participants.

Social norms have several advantages over more traditional reputation systems. For example, a typical reputation system can have a large (or even infinite) number of equilibria, some of which lead to high social welfare and revenue, and some of which do not (e.g., the "everyone behave badly" equilibrium). Even if a good equilibrium exists, it is not clear how to guarantee that participants will

[1] See, for example, the "Requesters Hall of Fame/Shame" forum on `turkernation.com`.

select it. With social norms, the equilibrium selection problem is easy since the platform designer explicitly recommends strategies to the participants. The participants only need to verify that it is in their own best interest to follow the recommended strategies.

We consider the large and general class of social norms with threshold-based social strategies. In a threshold-based social strategy, workers are expected to exert high effort on tasks posted by any requester whose reputation exceeds a specified value. Similarly, requesters are expected to pay those workers whose reputation exceeds a specified value. We show that under the class of maximum punishment reputation update rules, in which users who violate the social strategy have their reputation reset to the minimum possible value, the optimal threshold-based social strategy from the platform designer's perspective is surprisingly simple and intuitive. In particular, the optimal social strategy says that workers are expected to exert high effort on tasks posted by any requester who followed the social strategy on the previous round, and requesters are expected to pay those workers who followed the social strategy on the previous round. This can be viewed as a variant of tit-for-tat in the sense that participants punish other participants who did not adhere to the social strategy in the previous round. This result is reassuring in that it tells us we do not need to construct complicated incentive mechanisms with heavily optimized parameters to obtain optimal results. It is also somewhat surprising, given that complex parameter tuning is necessary to obtain optimality in P2P systems [118].

3.1.1 Related Work

There are a variety of options one might consider when designing an incentive mechanism, such as using virtual currency [74, 75] or virtual points [69]. In this chapter, we focus our discussion on a mechanism based on *reputation* [34, 35, 44, 90, 96]. We discuss the adaptive design of financial incentives in Chapter 4. In a reputation-based mechanism, rewards and punishments are typically determined according to a differential service scheme, which might require, for example, that a user who behaved well in the past (e.g., by making payments to workers on time) should receive more resources or better service than a user who did not. This preferential treatment provides an incentive for users to behave well.

The particular reputation mechanisms we propose are based on social norms [72, 93], which consist of a set of prescribed rules that market participants are asked to follow, and a mechanism for updating reputations based on whether or not they do. Our model and analysis differ in several important ways from most existing work on reputation systems, such as the influential work of Dellarocas [34, 35] on sanctioning mechanisms for environments with pure moral hazard. The traditional work on sanctioning mechanisms considers a one-sided market with one long-lived seller and many short-lived buyers, while we consider a two-sided market with a large number of long-lived workers and requesters. This distinction is especially important in scenarios with asymmetric information. In Dellarocas' model, in order to reach equilibrium, the seller's strategy must be publicly known. In our model, only the prescribed social norm must be known, and this can easily be announced by the market operator. Additionally, the analytical techniques used differ. Dellarocas' analysis proceeds by first finding the optimal objective value obtainable at equilibrium and then deriving a reputation mechanism to achieve this value. We first identify a parameterized family of social norms, and for every member of the family, derive the stationary distribution over users' reputations assuming they all follow the social strategy. We then use this stationary distribution to identify the set of sustainable mechanisms, and derive the optimal mechanism from this set. Both of these techniques have merit and each may be more appropriate in certain settings. For example, our technique may be more appropriate when the designer would like to restrict attention to classes of mechanisms that are particularly easy to implement or computationally tractable to run.

One advantage of social norms over traditional reputation systems is that the equilibrium selection problem is easy by design. The market operator announces the strategy that everyone should follow, and participants only need to verify that following is in their own best interest. This differs from typical reputation systems which may have a large or infinite number of equilibria, some of which lead to low payoffs for all.

To the best of our knowledge, we are the first to consider incorporating social norms into crowdsourcing markets. The most relevant related work is that of Zhang et al. [119] and Zhang and van der Schaar [118] on social norm design for P2P systems. In P2P systems, all participating users play similar roles, and only the transmitters take actions. This differs from crowdsourcing markets, in which the set of workers accepting tasks is typically mostly disjoint from the set of

task requesters, and both sets can take actions. Interactions in P2P systems are naturally modeled as gift-giving games, while interactions in crowdsourcing markets are more naturally modeled as instances of the Prisoner's Dilemma, as we see below. These apparently small differences lead to dramatically different results; to achieve optimality in P2P systems, it is necessary to carefully tune parameters based on the environment, while in our setting, a simple and intuitive mechanism is optimal for a wide range of environments.

3.2 Problem Formulation

We study the problem of designing social norms for crowdsourcing markets. In this chapter, we take the point of view of the *platform designer* (e.g., Amazon in the case of Amazon Mechanical Turk) who runs the market and typically receives a fixed percentage of all payments that are made. We therefore consider the goal of maximizing the total amount of money exchanged in the system, which can be viewed as a proxy for the platform designer's profit. (We discuss a simple modification of our model that incorporates the platform designer's profit more explicitly in Section 3.4.3.)

For clarity of presentation, we initially make several simplifying assumptions in our model. First, we assume that the user population is large and static, with the number of workers in the population equal to the number of requesters. (Dynamic populations are considered in Section 3.4.) Second, we assume that at each point in time, workers and requesters are randomly matched. The problem of assigning tasks to workers is of great importance, but is beyond the scope of this chapter. Third, we assume that workers and requesters are homogeneous in the sense that all workers pay the same cost (in terms of time and effort) to complete any task, and all requesters receive the same utility for tasks completed. Under these assumptions, it is natural to set one fixed price for all tasks as well.

When a worker and a requester are matched, the worker chooses to exert either high or low effort for the requester's task, and the requester chooses whether or not to pay the worker. Ideally, a requester would like to be able to carefully evaluate the worker's work before choosing whether or not to pay. However, in reality, the requester does not always have the ability to do this. Payment decisions are generally made relatively quickly, while it may take significant time for the requester

to evaluate the work. For this reason, we assume that a requester must choose whether or not to pay a worker *before* the action of the worker is revealed. However, afterwords, when the requester has had a chance to evaluate the work, the worker and requester can report each other's actions to the platform designer. The platform designer can then use these reports to update reputations.

Even after time has passed, it may be difficult for a requester to accurately evaluate the quality of a worker's work, especially for advanced tasks, such as translation. For this reason, we allow some noise in the reports. In particular, we assume that with probability ϵ_w, the action of a worker is misreported. Similarly, with probability ϵ_r, the action of a requester is misreported. Both ϵ_w and ϵ_r are common knowledge.

One might ask why we should expect users to report the actions of others truthfully. It is easy to see that in our setting, there is no incentive for workers or requesters to report dishonestly. These reports have no direct impact on the reputation of the reporter. Additionally, because we are considering a very large population, any indirect effects of a dishonest report would be negligible.

For the majority of this chapter, we assume that all market participants are rational and would like to maximize their long-term discounted sum of utilities. When dealing with rational participants, we assume that all workers and requesters have the same discount factor, δ.

3.2.1 Single Stage Game

Since workers and requesters are unable to observe each other's action before choosing their own actions, it is natural to model their interaction as a normal-form game as in Table 3.1. In this game, C denotes a worker's cost for exerting high effort on the requester's task, V denotes the benefit a requester receives when a worker exerts high effort, and P denotes the monetary payment associated with the task. We implicitly assume the worker's cost for exerting low effort is 0, but this assumption is without loss of generality since our analysis depends only on the difference between the cost of exerting high effort and the cost of exerting low effort. Similarly, it is without loss of generality to assume the requester's value for receiving low effort work is 0.

We are interested only in the natural setting in which the action pair (High Effort, Pay) leads to higher payoffs than the action pair (Low Effort, Don't Pay) for both the worker and the requester. If

		Requester	
		Pay	Don't Pay
Worker	High Effort	$P - C, V - P$	$-C, V$
	Low Effort	$P, -P$	$0, 0$

Table 3.1: Payoff matrix of the stage game.

this were not the case, then social welfare would be maximized by everyone choosing Low Effort or Don't Pay, and there would be no need to design a social norm. Therefore, we assume that $0 < C < P < V$.

This game is a variation of the well-studied Prisoner's Dilemma. It is easy to see the only Nash equilibrium in this game is the action pair (Low Effort, Don't Pay).

3.2.2 Repeated Game with Social Norm

In the single-stage game described above, the only Nash equilibrium results in a low payoff for both the requester and the worker. Fortunately, in crowdsourcing markets, workers and requesters typically participate in the market many times. Therefore, it is natural to model a crowdsourcing market as a repeated game, in which workers and requesters play the normal-form game above with a randomly chosen opponent at each point in time, as is common in the literature [42]. The social norms that we propose will be for this repeated game setting.

In the social norms that we design, both requesters and workers are assigned reputation values that are updated over time. The platform designer announces a prescribed social strategy (e.g., "don't exert effort for/pay participants with zero reputation") and updates participants' reputations based on how well they follow this strategy.

Formally speaking, a social norm consists of two parts:

- The *social strategies* σ_w and σ_r are functions that define the prescribed actions for the worker and requester respectively when a worker with reputation θ_w is matched with a requester with reputation θ_r.

- The *reputation update rules* τ_w and τ_r define how to update the reputation of the worker and requester respectively after each transaction. We assume the reputation values are bounded between 0 and an upper bound L.

Designing a social norm involves specifying both the social strategies and the reputation update rules.

3.3 Optimal Social Norm Design

In this section, we illustrate how to design optimal social norms under the most basic version of our model. For now, we limit our attention to the most natural class of social strategies, threshold-based social strategies, paired with maximum punishment reputation update rules, both defined below. We first derive a set of conditions that can be used to determine whether or not a particular social norm is sustainable, i.e., whether or not each user has incentive to follow the social norm if she believes that everyone else is following. We then formalize the objective function of the platform designer and show how to find the optimal sustainable social norm with respect to this objective function. The general technique that we illustrate can also be applied to more elaborate variants of our model, alternative objective functions, or alternative classes of social norms.

A *threshold-based social strategy* is based on a pair of threshold values (k_r, k_w). Workers are expected to exert high effort when they face requesters with reputation at least k_r, and exert low effort otherwise. Similarly, requesters are expected to pay the payment only when they are matched with workers with reputation at least k_w. This is formalized in the following definition.

Definition 1. *The threshold-based social strategies σ_w and σ_r with parameters k_r and k_w are defined as:*

$$\sigma_w(\theta_w, \theta_r) = \begin{cases} \textit{“High Effort”} & \textit{if } \theta_r \geq k_r, \\ \textit{“Low Effort”} & \textit{otherwise}. \end{cases}$$

$$\sigma_r(\theta_w, \theta_r) = \begin{cases} \textit{“Pay”} & \textit{if } \theta_w \geq k_w, \\ \textit{“Don't Pay”} & \textit{otherwise}. \end{cases}$$

We initially consider a very simple class of update rules. Each time the user follows the social strategy, her reputation is increased by 1 until it hits the limit L.[2] Each time the user deviates from the social strategy, her reputation is reset to 0. We call this the *maximum punishment reputation update rule* and define it more formally as follows.

Definition 2. *The maximum punishment reputation update rules τ_w and τ_r with parameter L are defined as:*

$$\tau_w(\theta_w, \theta_r, a) = \begin{cases} \min\{\theta_w + 1, L\} & \text{if } a = \sigma_w(\theta_w, \theta_r), \\ 0 & \text{otherwise.} \end{cases}$$

$$\tau_r(\theta_w, \theta_r, a) = \begin{cases} \min\{\theta_r + 1, L\} & \text{if } a = \sigma_r(\theta_w, \theta_r), \\ 0 & \text{otherwise.} \end{cases}$$

We first show that there exists a unique stationary reputation distribution for the maximum punishment reputation update rules. We then derive the sustainable conditions and find the optimal values of the parameters k_r, k_w, L, and P. We show that the resulting optimal social norm is both simple and intuitive for market participants.

3.3.1 The Stationary Reputation Distribution

Our first lemma states that there exists a stationary distribution of worker and requester reputations under the assumption that all users follow the prescribed social strategy. Recall that every worker's (requester's) action at a given time step is misreported with probability ϵ_w (ϵ_r).

Lemma 3.3.1. *In the basic model, under the maximum punishment reputation update rule with any $L > 0$ paired with* any *social strategy, if all users follow the social strategy, then there exists a unique stationary distribution $\{\eta_w(\theta)\}$ over worker reputations and a unique stationary distribution*

[2] We could define separate limits for workers and requesters. However, it turns out that this would not change the optimal social norm. To simplify presentation, we therefore stick to one parameter L.

$\{\eta_r(\theta)\}$ *over requester reputations given by*

$$\eta_w(\theta) = (1-\epsilon_w)^\theta \epsilon_w \text{ , for } \theta \in \{0, \cdots, L\}$$
$$\eta_w(L) = (1-\epsilon_w)^L$$

$$\eta_r(\theta) = (1-\epsilon_r)^\theta \epsilon_r \text{ , for } \theta \in \{0, \cdots, L\}$$
$$\eta_r(L) = (1-\epsilon_r)^L.$$

Proof. We show how to derive the stationary distribution $\{\eta_w(\theta)\}$ over worker reputations. The derivation for the distribution over requester reputations is identical.

Let $\eta_w^t(\theta)$ denote the fraction of workers with reputation θ at time t. Assuming that all workers in the market follows the social strategy σ_w, the transition of worker reputations over time can be described as follows. At any time t,

$$\eta_w^{t+1}(0) = \epsilon_w$$
$$\eta_w^{t+1}(\theta) = (1-\epsilon_w)\eta_w^t(\theta-1) \text{ , for } 1 \leq \theta \leq L-1$$
$$\eta_w^{t+1}(L) = (1-\epsilon_w)\{\eta_w^t(L-1) + \eta_w^t(L)\}.$$

When the distribution is stationary, $\eta_r^{t+1}(\theta) = \eta_r^t(\theta)$ for all θ. If for each value of θ, we substitute a single variable $\eta_r(\theta)$ for $\eta_r^{t+1}(\theta)$ and $\eta_r^t(\theta)$ in the equations above, we end up with $L+1$ independent linear equations with $L+1$ unknowns. These linear equations can be uniquely solved, yielding the values of $\eta_r^t(\theta)$ in the lemma statement.

We can verify that these values do indeed form a probability distribution by checking that $\sum_{\theta=0}^{L} \eta_r(\theta) = 1$. When $\eta_r^t(\theta) = \eta_r^{t+1}(\theta) = \eta_r(\theta)$ for all θ, summing over the $L+1$ constraints given above gives us exactly what we need. □

The stationary distributions do not depend on the particular social strategy used. Note that in the special case in which the reporting errors are 0, all workers and requesters have the highest reputation value.

3.3.2 Sustainability Conditions

We next investigate the conditions under which users have incentive to follow the threshold-based social norm. If the social strategy states that requesters never pay and workers never exert high effort, it would be sustainable trivially, but this bad equilibrium is undesirable. We therefore focus on the non-trivial case where $k_w \leq L$ and $k_r \leq L$.

To check if a social norm is sustainable, we check whether a user has incentive to deviate from the social norm given all other users are following. Since we assume the population of users is large, the deviation of single agent will not influence the overall distribution. Therefore, we can calculate the long-term expected payoff using the stationary reputation distribution. Below we show the sustainable conditions for the threshold-based social norms. Recall that δ denotes the discount factor, which is common among all users.

Lemma 3.3.2. *In the basic model, the threshold-based social strategy with parameters k_w and k_r paired with the maximum punishment reputation update rule with $L > 0$ is sustainable if and only if $k_w > 0$, $k_r > 0$, and*

$$\frac{1}{\delta^{k_w}(1-\epsilon_w)^{k_w-1}(1-2\epsilon_w)}\ C \leq P \leq \delta^{k_r}(1-\epsilon_r)^{k_r-1}(1-2\epsilon_r)\ V.$$

The proof of this lemma, which uses the one-shot deviation principle of game theory, appears in the appendix.

To gain intuition about Lemma 3.3.2, consider the case in which the reports of behavior are noise-free, that is, $\epsilon_r = \epsilon_w = 0$. The lemma then implies that the social norm is sustainable if $C \leq \delta^{k_w} P$ and $P \leq \delta^{k_r} V$. These are precisely the conditions under which a worker or requester with reputation 0 has incentive to follow the social norm. The left-hand side of each equation is the current maximum additional payoff a worker (first equation) or requester (second equation) can get by deviating from the social norm, and the right-hand side is the difference of expected future payoff she can get if she does not deviate from the social norm in the current stage. Since users with reputation 0 have the least incentive to follow (they have no reputation to lose), these conditions suffice to ensure that all users have incentive to follow.

This result gives us a feasible interval in which we can set the price P. If the ratio of value to cost or the discount factor is too small, (i.e., if $V/C \leq 1/\delta^{k_r+k_w}$), then there is no way to set the price to obtain a sustainable social norm.

3.3.3 The Optimal Social Norm

With the sustainability conditions in place, we are now ready to design the optimal threshold-based social norm. We define optimality in terms of the total of all payments exchanged, which is a proxy for the revenue of the platform designer. (In Section 3.4.3 we look at the platform designer's revenue directly.) The designer has control of four parameters: the payment P, the threshold values k_r and k_w, and the maximum reputation L.

We first formally define the objective function of the design problem. Let $P(\theta_w, \theta_r) = P$ if the requester of reputation θ_r is asked to pay in a transaction with worker of reputation θ_w, i.e., if $\sigma_r(\theta_w, \theta_r) =$ "Pay", and $P(\theta_w, \theta_r) = 0$ otherwise. Assuming that the population has already converged to the unique stationary distribution, we can express the objective function U as

$$U = \sum_{\theta_r=0}^{L} \sum_{\theta_w=0}^{L} \eta_w(\theta_w)\eta_r(\theta_r)P(\theta_w, \theta_r).$$

Note that $P(\theta_w, \theta_r)$ depends on the choice of k_w, but not k_r. If everyone follows the social strategy, the stationary distributions do not depend on k_w or k_r. We can therefore conclude that under the assumption that all users follow the social strategies:

- For fixed values P, k_r, and L, U is non-increasing in k_w.
- For fixed values P, k_w, and L, U is constant in k_r.

This allows us to derive the optimal parameter settings.

Theorem 3.3.3. *In the basic model, restricting attention to threshold-based social strategies paired with maximum punishment reputation update rules, if*

$$\frac{V}{C} \geq \frac{1}{\delta^2(1-2\epsilon_w)(1-2\epsilon_r)}$$

then total payments are maximized by setting $k_r = k_w = L = 1$ and $P = \delta(1-2\epsilon_r)V$. The optimal

value of the objective U is then $\delta(1-\epsilon_w)(1-2\epsilon_r)V$. Otherwise, there is no sustainable social norm.

Proof. Fix any $L > 0$. From Lemma 3.3.2, we know that if a sustainable social norm exists, then regardless of the values of k_r and k_w, U is maximized if we set $P = \delta(1-2\epsilon_r)V$. Furthermore, examining the condition in Lemma 3.3.2, we can see that if the social norm is not sustainable with $k_r = k_w = 1$, then it will not be sustainable with any setting of k_r and k_w. As we argued above, U is non-increasing in k_w and constant in k_r, so setting $k_r = k_w = 1$ maximizes U.

Finally, consider the choice of L. With $k_r = k_w = 1$, the value of maximum reputation L does not affect the objective at all. We can therefore set $L = 1$ for simplicity. □

With both of the thresholds and the maximum reputation L equal to 1, the optimal social norm from Theorem 3.3.3 is surprisingly simple and intuitive. In this social norm, there are only two different reputation values. Modulo the effects of noise, users who followed the social norm on the previous time step have reputation 1, while users who did not follow have reputation 0. The social strategy then says that users should play a tit-for-tat-like strategy, exerting high effort for or paying those users who obeyed the social norm on the previous time step, and punishing those who did not.

This result is both reassuring and somewhat surprising. It tells us that there is no need to construct complicated incentive mechanisms that are difficult for users to understand, or to heavily optimize parameters of the mechanism based on properties of the market participants. More interestingly, in the next section, we show that this intuitive social norm is still optimal even if we relax some of the assumptions of the basic model. This is in stark contrast to the P2P setting [118, 119] in which in order to derive optimal social norms it is necessary to tune complex parameters based on properties of the environment.

3.4 Beyond the Basic Model

We now show that the simple and intuitive tit-for-tat-like social norm described above remains optimal when we extend the basic model to allow a dynamic population, consider some fraction of nonstrategic users in the market, or explicitly take into account the fees imposed by the platform

designer. Only the optimal price P changes. Proofs of these results appear in the appendix.

3.4.1 Dynamic Population and Whitewashing

Until now, we have assumed that the user population is static. However, in real crowdsourcing markets, participants enter and exit the market over time. This affects user incentives in several ways. First, if users have the ability to leave and rejoin the market with a new identity, they may be tempted to engage in *whitewashing*, creating a new identity to escape a bad reputation. This is an important issue in reputation systems [45], since creating new identities is usually easy in online settings. Second, if a user knows he will not stay in the market forever, he may be less willing to work hard to earn a high reputation.

We first consider the problem of whitewashing. To some extent, this could be prevented outside the social norm by requiring personal identification, such as a credit card number, to create a new account. However, it may be undesirable to require such identification as it could prevent some users from signing up, so it is not always required in practice. Luckily, it is easy to prevent whitewashing using social norms by setting the reputation of any new user to 0. It is clear that doing this removes any incentive that a user might have to erase his history.

With this problem under control, we need only to consider how the turnover rate affects the stationary reputation distributions and users' expectations about their future payoffs. To simplify analysis, we assume the size of worker and requester populations stay the same at all times, but at every time step, some fraction of users leave the market and are replaced by new users (with reputation 0). Let α_w be the fraction of workers who leave and are replaced by new workers at each time step, and α_r be the fraction of requesters who leave and are replaced. We obtain the following.

Theorem 3.4.1. *Restricting attention to threshold-based social strategies paired with maximum punishment reputation update rules, for any turnover rates α_w and $\alpha_r \in [0, 1)$, if*

$$\frac{V}{C} \geq \frac{1}{\delta^2(1-\alpha_r)(1-\alpha_w)(1-2\epsilon_w)(1-2\epsilon_r)}$$

then total payments are maximized by setting $k_r = k_w = L = 1$ and $P = \delta(1-\alpha_r)(1-2\epsilon_r)V$. Otherwise, there is no sustainable social norm.

The proof is similar to the proof of Theorem 3.3.3 with two modifications. First, the stationary distribution changes in this setting. Second, it is necessary to take into account the fact that users will discount their expected payoff by the turnover rate, since they expect to leave the market with probability α_w or α_r in the next time step.

3.4.2 Nonstrategic Users

Until this point, we have assumed that all users are strategic and select actions to maximize their long-term utilities. While this assumption is relatively natural and quite useful analytically, in real-world markets there exist users who do not behave rationally. For example, a system may have *altruists*, who always perform "good" actions (i.e., making payments or exerting high effort), *malicious* users, who always perform "bad" actions (i.e., not paying or exerting low effort), or users who choose actions arbitrarily, ignoring social strategies and their own utility. Any reputation mechanisms should be robust to the existence of nonstrategic users in order to be useful in practice.

The following theorem applies whenever some fraction of the population is nonstrategic, but choose actions independent of their opponents' reputation at each point in time. Call such a user *oblivious-nonstrategic*. We assume that the fraction of nonstrategic users is known, but the identities of the nonstrategic users are not.

Theorem 3.4.2. *Restricting attention to threshold-based social strategies paired with maximum punishment reputation update rules, for any $f_r, f_w \in [0, 1)$, if a fraction f_r of requesters and a fraction f_w of workers are oblivious-nonstrategic, then if*

$$\frac{V}{C} \geq \frac{1}{\delta^2(1-f_w)(1-f_r)(1-2\epsilon_r)(1-2\epsilon_w)}$$

then total payments are maximized by setting $k_r = k_w = L = 1$ and $P = \delta(1-f_w)(1-2\epsilon_r)V$. Otherwise, there is no sustainable social norm.

As we can see in the theorem, the existence of nonstrategic users will shorten the interval of feasible prices. However, it does not affect the social strategy design; the simple tit-for-tat-like social norm is still optimal. Notice that altruistic and malicious users have the same impact on the sustainability conditions, though clearly the strategies of these users will impact the total payment.

3.4.3 Transaction Fees

We now discuss how to explicitly incorporate the platform designer's fee into the analysis. We assume the platform designer always takes a fixed portion m from the payment as the transaction fee. Since the objective function is $1/m$ of the total payment, introducing this fee does not change the design problem in terms of the objective. However, there is now a difference between the payment paid by requesters (P) and the payment received by workers ($P(1-m)$). The optimal social norm is characterized as follows.

Theorem 3.4.3. *Suppose that the platform designer takes a fixed portion $m \in [0,1)$ of all payments. Restricting attention to threshold-based social strategies paired with maximum punishment reputation update rules, if*

$$\frac{V}{C} \geq \frac{1}{\delta^2(1-2\epsilon_w)(1-2\epsilon_r)(1-m)}$$

then total transaction fees are maximized by setting $k_r = k_w = L = 1$, $P = \delta(1-2\epsilon_r)V$, and

$$m = 1 - \frac{1}{\delta^2(1-2\epsilon_w)(1-2\epsilon_r)}\frac{C}{V}.$$

Otherwise, there is no sustainable social norm.

Incorporating the fee only introduces an additional parameter in the sustainability condition, leading to a slightly reduced space of feasible prices. The optimal social norm remains unchanged. The optimal value of the fee m is the largest possible value which still sustains the social norm.

3.5 Summary

In this chapter, we introduced a framework for the design and analysis of incentive schemes for crowdsourcing markets based on social norms, and described a general technique that can be used to derive the optimal social norm from within a class of interest.

We illustrated the use of this technique to derive the optimal social norm from within the natural class of threshold-based social strategies paired with maximum punishment reputation update rules, and showed that the optimal norm in this class is simple to implement and understand. Furthermore,

the optimal social strategy does not depend on features of the environment such as the amount of noise in reports, the turnover rate of the population, or the fraction of non-strategic users, making it applicable in a variety of settings.

This chapter presents a first step towards a complete, robust theory of incentive design for crowdsourcing systems. An important next step in developing this theory is to build upon our illustrative examples of heterogeneity to derive techniques for obtaining optimal social norms for any distribution over worker costs and requester values. Allowing full heterogeneity would also introduce interesting questions related to the problem of optimally matching workers and requesters. Finally, developing a full theory would require digging into issues related to learning and convergence to the stationary distribution. Our hope is that the framework and techniques in this chapter will provide the necessary groundwork for future progress towards this goal.

CHAPTER 4

Designing Incentives - Performance Based Payments

4.1 Overview

In this chapter, we explore the use of another kind of incentive, performance-based payments, to motivate workers to perform high-quality work. In particular, we examine the requester's problem of dynamically setting performance-based payments for tasks. We consider a setting in which time evolves in rounds. In each round, the requester posts a new contract, a performance-based payment rule which specifies different levels of payment for different levels of output. A random, unidentifiable worker then arrives in the market and strategically decides whether to accept the requester's task and how much effort to exert; the choice of effort level is not directly observable by the requester. After the worker completes the task (or chooses not to complete it), the requester observes the worker's output, pays the worker according to the offered contract, and adjusts the contract for the next round. The properties of a random worker (formally: the distribution over the workers' types) are not known to the requester, but may be learned over time. The goal of the requester is to maximize his expected utility, the value he receives from completed work minus the payments made. We call it the *dynamic contract design* problem.

For concreteness, consider a special case in which a worker can strategically choose to perform a task with low effort or with high effort, and the task may be completed either at low quality or at high quality. The low effort incurs no cost and results in low quality, which in turn brings no value to the requester. The high effort leads to high quality with some positive probability (which may vary from one worker to another, and is unknown to the requester). The requester only observes the quality of completed tasks, and therefore cannot infer the effort level. This example captures the two main tenets of our model: that the properties of a random worker are unknown to the requester

and that workers' strategic decisions are unobservable.

We treat the dynamic contract design problem as a multi-armed bandit (MAB) problem, with each arm representing a potential contract. Since the action space is large (potentially infinite) and has a well-defined real-valued structure, it is natural to consider an algorithm that uses *discretization*. Our algorithm, `AgnosticZooming`, divides the action space into regions, and chooses among these regions, effectively treating each region as a single "meta-arm." The discretization is defined *adaptively*, so that the more promising areas of the action space are eventually discretized more finely than the less promising areas. While the general idea of adaptive discretization has appeared in prior work on MAB [17, 80, 106, 107], our approach to adaptive discretization is new and problem-specific. The main difficulty, compared to this prior work, is that an algorithm is not given any information that links the observable numerical structure of contracts and the expected utilities thereof.

To analyze performance, we propose a concept called "width dimension" which measures how "nice" a particular problem instance is. We show that `AgnosticZooming` achieves regret sublinear in the time horizon for problem instances with small width dimension. In particular, if the width dimension is d, it achieves regret $O(\log T \cdot T^{(d+1)/(d+2)})$ after T rounds. For problem instances with large width dimension, `AgnosticZooming` matches the performance of the naive algorithm which uniformly discretizes the space and runs a standard bandit algorithm. We illustrate our general results via some corollaries and special cases, including the high-low example described above. We support the theoretical results with simulations.

Further, we consider a special case of our setting where each worker only chooses whether to accept or reject a given task. This special case corresponds to a dynamic pricing problem previously studied in the literature. Our results significantly improve over the prior work on this problem.

Our contributions can be summarized as follows. We define a broad, practically important setting in crowdsourcing markets; identify novel problem-specific structure, for both the algorithm and the regret bounds; distill ideas from prior work to work with these structures; argue that our approach is productive by deriving corollaries and comparing to prior work; and identify and analyze specific examples where our theory applies. The main conceptual contributions are the model itself and the

adaptive discretization approach mentioned above. Finally, this paper prompts further research on dynamic contract design along several directions that we outline in the conclusion.

4.1.1 Related Work

Our work builds on three areas of research. First, our model can be viewed as a multi-round version of the classical *principal-agent* model from contract theory [82]. A single round of our model corresponds to the basic principal-agent setting, with *adverse selection* (unknown worker's type) and *moral hazard* (unobservable worker's decisions). Unlike much of the prior work in contract theory, the prior over worker types is not known to the principal, but may be learned over time. Accordingly, our techniques are very different from those employed in contract theory.

Second, our methods build on those developed in the rich literature on MAB with continuous outcome spaces. The closest line of work is that on *Lipschitz MAB* [80], in which the algorithm is given a distance function on the arms, and the expected rewards of the arms are assumed to satisfy Lipschitz-continuity (or a relaxation thereof) with respect to this distance function, [2, 8, 17, 78, 80, 107]. Most related to our techniques is the idea of adaptive discretization [17, 80, 107], and in particular, the *zooming algorithm* [80, 107]. However, the zooming algorithm cannot be applied directly in our setting because the required numerical similarity information is not immediately available. This problem also arises in web search and advertising, where it is natural to assume that an algorithm can only observe a tree-shaped taxonomy on arms [81, 92, 95] which can be used to explicitly reconstruct relevant parts of the underlying metric space [21, 106]. We take a different approach, using a notion of "virtual width" to estimate similarity information. Explicit comparisons between our results and prior MAB work are made throughout the paper.

Finally, our work follows several other theoretical papers on pricing in crowdsourcing markets. The problem closest to ours which has been studied in this context is *dynamic task pricing*, which is essentially the special case of our setting where in each round a worker is offered to perform a task at a specified price, and can either accept or reject this offer [9, 10, 79, 104, 105].[1] In particular, in this setting the worker's strategic choice is directly observable.

[1] In Badanidiyuru et al. [10], this problem is called "dynamic procurement".

4.2 Problem Formulation

In this section, we formally define the problem that we set out to solve. We start by describing a *static model*, which captures what happens in a single round of interaction between a requester and a worker. As described above, this is a version of the standard *principal-agent* model [82]. We then define our *dynamic model*, an extension of the static model to multiple rounds, with a new worker arriving each round. We then detail the objective of our pricing algorithm and the simplifying assumptions that we make throughout the paper. Finally, we compare our setting to the classic multi-armed bandit problem.

Static model. We begin with a description of what occurs during each interaction between the requester and a single worker. The requester first posts a task which may be completed by the worker, and a *contract* specifying how the worker will be paid if she completes the task. If the task is completed, the requester pays the worker as specified in the contract, and the requester derives value from the completed task; for normalization, we assume that the value derived is in $[0, 1]$. The requester's utility from a given task is this value minus the payment to the worker.

When the worker observes the contract and decides whether or not to complete the task, she also chooses a level of effort to exert, which in turn determines her cost (in terms of time, energy, or missed opportunities) and a distribution over the quality of her work. To model quality, we assume that there is a (small) finite set of possible *outcomes* that result from the worker completing the task (or choosing not to complete it), and that the realized outcome determines the value that the requester derives from the task. The realized outcome is observed by the requester, and the contract that the requester offers is a mapping from outcomes to payments for the worker.

We emphasize two crucial (and related) features of the principal-agent model: that the mapping from effort level to outcomes can be randomized, and that the effort level is not directly observed by the requester. This is in line with a standard observation in crowdsourcing that even honest, high-effort workers occasionally make errors.

The worker's utility from a given task is the payment from the requester minus the cost corresponding to her chosen effort level. Given the contract she is offered, the worker chooses her effort

level strategically so as to maximize her expected utility. Crucially, the chosen effort level is not directly observable by the requester.

The worker's choice *not* to perform a task is modeled as a separate effort level of zero cost (called the *null* effort level) and a separate outcome of zero value and zero payment (called the *null* outcome) such that the null effort level deterministically leads to the null outcome, and it is the only effort level that can lead to this outcome.

The mapping from outcomes to the requester's value is called the requester's *value function*. The mapping from effort levels to costs is called the *cost function*, and the mapping from effort levels to distributions over outcomes is called the *production function*. For the purposes of this paper, a worker is completely specified by these two functions; we say that the cost function and the production function comprise the worker's *type*. Unlike some traditional versions of the principal-agent problem, in our setting a worker's type is not observable by the requester, nor is any prior given.

Dynamic model. The dynamic model we consider in this paper is a natural extension of the static model to multiple rounds and multiple workers. We are still concerned with just a single requester. In each round, a new worker arrives. We assume a *stochastic environment* in which the worker's type in each round is an i.i.d. sample from some fixed and unknown distribution over types, called the *supply distribution*. The requester posts a new task and a contract for this task. All tasks are of the same type, in the sense that the set of possible effort levels and the set of possible outcomes are the same for all tasks. The worker strategically chooses her effort level so as to maximize her expected utility from this task. Based on the chosen effort level and the worker's production function, an outcome is realized. The requester observes this outcome (but not the worker's effort level) and pays the worker the amount specified by the contract. The type of the arriving worker is never revealed to the requester. The requester can adjust the contract from one round to another, and his total utility is the sum of his utility over all rounds. For simplicity, we assume that the number of rounds is known in advance, though this assumption can be relaxed using standard tricks.

The dynamic contract design problem. Throughout this paper, we take the point of view of the requester interacting with workers in the dynamic model. The algorithms we examine dynamically

choose contracts to offer on each round with the goal of maximizing the requester's expected utility. A problem instance consists of several quantities, some of which are known to the algorithm, and some of which are not. The known quantities are the number of outcomes, the requester's value function, and the time horizon T (i.e., the number of rounds). The latent quantities are the number of effort levels, the set of worker types, and the supply distribution. The algorithm adjusts the contract from round to round and observes the realized outcomes but receives no other feedback.

We focus on contracts that are *bounded* (offer payments in $[0, 1]$), and *monotone* (assign equal or higher payments for outcomes with higher value for the requester). Let X be the set of all bounded, monotone contracts. We compare a given algorithm against a given subset of "candidate contracts" $X_{\texttt{cand}} \subset X$. Letting $\texttt{OPT}(X_{\texttt{cand}})$ be the optimal utility over all contracts in $X_{\texttt{cand}}$, the goal is to minimize the algorithm's *regret* $R(T|X_{\texttt{cand}})$, defined as $T \times \texttt{OPT}(X_{\texttt{cand}})$ minus the algorithm's expected utility.

The subset $X_{\texttt{cand}}$ may be finite or infinite, possibly $X_{\texttt{cand}} = X$. The most natural example of a finite $X_{\texttt{cand}}$ is the set of all bounded, monotone contracts with payments that are integer multiples of some $\psi > 0$; we call it the *uniform mesh* with granularity ψ, and denote it $X_{\texttt{cand}}(\psi)$.

Notation. Let $v(\cdot)$ be the value function of the requester, with $v(\pi)$ denoting the value of outcome π. Let $\mathcal{O}$ be the set of all outcomes and let m be the number of non-null outcomes. We will index the outcomes as $\mathcal{O} = \{0, 1, 2, \ \ldots \ , m\}$ in the order of increasing value (ties broken arbitrarily), with a convention that 0 is the null outcome.

Let $c_i(\cdot)$ and $f_i(\cdot)$ be the cost function and production function for type i. Then the cost of choosing effort level e is $c_i(e)$, and the probability of obtaining outcome π having chosen effort e is $f_i(\pi|e)$. Let $F_i(\pi|e) = \sum_{\pi' \geq \pi} f_i(\pi'|e)$.

Recall that a contract x is a function from outcomes to (non-negative) payments. If contract x is offered to a worker sampled i.i.d. from the supply distribution, $V(x)$ is the expected value to the requester, $P(x) \geq 0$ is the expected payment, and $U(x) = V(x) - P(x)$ is the expected utility of the requester. Let $\texttt{OPT}(X_{\texttt{cand}}) = \sup_{x \in X_{\texttt{cand}}} U(x)$.

Assumption: First-order stochastic dominance (FOSD). Given two effort levels e and e', we say that e has FOSD over e' for type i if $F_i(\pi|e) \geq F_i(\pi|e')$ for all outcomes π, with a strict inequality

for at least one outcome.[2] We say that type i satisfies the FOSD assumption if for any two distinct effort levels, one effort level has FOSD over the other for type i. We assume that all types satisfy this assumption.

Assumption: Consistent tie-breaking. If multiple effort levels maximize the expected utility of a given worker for a contract x, we assume the tie is broken consistently in the sense that this worker chooses the same effort level for any contract that leads to this particular tie. This assumption is minor; it can be avoided (with minor technical complications) by adding random perturbations to the contracts. This assumption is implicit throughout the paper.

4.2.1 Discussion

Number of outcomes. Our results assume a small number of outcomes. This regime is important in practice, as the quality of submitted work is typically difficult to evaluate in a very fine granularity. Even with $m = 2$ non-null outcomes, our setting has not been studied before. The special case $m = 1$ is equivalent to the dynamic pricing problem from Kleinberg and Leighton [79]; we obtain improved results for it, too.

The benchmark. Our benchmark $\texttt{OPT}(\cdot)$ only considers contracts that are bounded and monotone. In practice, restricting to such contracts may be appealing to all human parties involved. However, this restriction is not without loss of generality: there are problem instances in which monotone contracts are not optimal; see Appendix A.3.5 for an example. Further, it is not clear whether bounded monotone contracts are optimal among monotone contracts.

Our benchmark $\texttt{OPT}(X_{\texttt{cand}})$ is relative to a given set $X_{\texttt{cand}}$, which is typically a finite discretization of the contract space. There are two reasons for this. First, crowdsourcing platforms may require the payments to be multiples of some minimum unit (e.g., one cent), in which case it is natural to restrict our attention to contracts satisfying the same constraint. Second, achieving guarantees relative to $\texttt{OPT}(X)$ for the full generality of our problem appears beyond the reach of our techniques. As in many other machine learning scenarios, it is useful to consider a restricted

[2]This mimics the standard notion of FOSD between two distributions over a linearly ordered set.

"benchmark set" – set of alternatives to compare to.[3] In such settings, it is considered important to handle *arbitrary* benchmark sets, which is what we do.

One known approach to obtain guarantees relative to $\texttt{OPT}(X)$ is to start with some finite $X_{\texttt{cand}} \subset X$, design an algorithm with guarantees relative to $\texttt{OPT}(X_{\texttt{cand}})$, and then, as a separate result, bound the discretization error $\texttt{OPT}(X) - \texttt{OPT}(X_{\texttt{cand}})$. Then the choice of $X_{\texttt{cand}}$ drives the tradeoff between the discretization error and regret $R(T|X_{\texttt{cand}})$, and one can choose $X_{\texttt{cand}}$ to optimize this tradeoff. However, while one can upper-bound the discretization error in some (very) simple special cases, it is unclear whether this can be extended to the full generality of dynamic contract design.

Alternative worker models. One of the crucial tenets in our model is that the workers maximize their expected utility. This "rationality assumption" is very standard in Economics, and is often used to make the problem amenable to rigorous analysis. However, there is a considerable literature suggesting that in practice workers may deviate from this "rational" behavior. Thus, it is worth pointing out that our results do not rely heavily on the rationality assumption. The FOSD assumption (which is also fairly standard) can be circumvented, too. In fact, all our assumptions regarding worker behavior serve *only* to enable us to prove Lemma 4.3.1, and more specifically to guarantee that the collective worker behavior satisfies the following natural property (which is used in the proof of Lemma 4.3.1): if the requester increases the "increment payment" (as described in the next section) for a particular outcome, the probability of obtaining an outcome at least that good also increases.

Minimum wage. For ethical or legal reasons one may want to enforce some form of minimum wage. This can be expressed within our model as a *minimal payment* θ for a completed task, i.e., for any non-null outcome. Our algorithm can be easily modified to accommodate this constraint. Essentially, it suffices to restrict the action space to contracts that pay at least θ for a completed task. Formally, the "increment space" defined in Section 4.3 should be $[\theta, 1] \times [0, 1]^{m-1}$ rather than $[0, 1]^m$, and the "quadrants" of each "cell" are defined by splitting the cell in half in each dimension. All our results easily carry over to this version (restricting $X_{\texttt{cand}}$ to contracts that pay at least θ for a

[3] A particularly relevant analogy is contextual bandits with policy sets, e.g., Dudik et al. [39].

completed task). We omit further discussion of this issue for the sake of simplicity.

Comparison to multi-armed bandits (MAB). Dynamic contract design can be modeled as special case of the MAB problem with some additional, problem-specific structure. The basic MAB problem is defined as follows. An algorithm repeatedly chooses actions from a fixed *action space* and collects rewards for the chosen actions; the available actions are traditionally called *arms*. More specifically, time is partitioned into rounds, so that in each round the algorithm selects an arm and receives a reward for the chosen arm. No other information, such as the reward the algorithm would have received for choosing an alternative arm, is revealed. In an MAB problem with *stochastic rewards*, the reward of each arm in a given round is an i.i.d. sample from some distribution which depends on the arm but not on the round. A standard measure of algorithm's performance is regret with respect to the best fixed arm, defined as the difference in expected total reward between a benchmark (usually the best fixed arm) and the algorithm.

Thus, dynamic contract design can be naturally modeled as an MAB problem with stochastic rewards, in which arms correspond to monotone contracts. The prior work on MAB with large / infinite action spaces often assumes known upper bounds on similarity between arms. More precisely, this prior work would assume that an algorithm is given a metric $\mathcal{D}$ on contracts such that expected rewards are Lipschitz-continuous with respect to $\mathcal{D}$, i.e., we have upper bounds $|U(x) - U(y)| \leq \mathcal{D}(x, y)$ for any two contracts x, y.[4] However, in our setting such upper bounds are absent. On the other hand, our problem has some supplementary structure compared to the standard MAB setting. In particular, the algorithm's reward decomposes into value and payment, both of which are determined by the outcome, which in turn is probabilistically determined by the worker's strategic choice of the effort level. Effectively, this supplementary structure provides some "soft" information on similarity between contracts, in the sense that numerically similar contracts are usually (but not always) similar to one another.

[4]Such upper bound is informative if and only if $\mathcal{D}(x, y) < 1$.

4.3 Our Algorithm: `AgnosticZooming`

In this section, we specify our algorithm. We call it `AgnosticZooming` because it "zooms in" on more promising areas of the action space, and does so without knowing a precise measure of the similarity between contracts. This zooming can be viewed as a dynamic form of discretization. Before stating the algorithm itself, we discuss the discretization of the action space in more detail, laying the groundwork for our approach.

4.3.1 Discretization of the action space

In each round, the `AgnosticZooming` algorithm partitions the action space into several regions and chooses among these regions, effectively treating each region as a "meta-arm." In this section, we discuss which subsets of the action space are used as regions, and introduce some useful notions and properties of such subsets.

Increment space and cells. To describe our approach to discretization, it is useful to think of contracts in terms of *increment payments*. Specifically, we represent each monotone contract $x : \mathcal{O} \to [0, \infty)$ as a vector $\mathbf{x} \in [0, \infty)^m$, where m is the number of non-null outcomes and $\mathbf{x}_\pi = x(\pi) - x(\pi - 1) \geq 0$ for each non-null outcome π. (Recall that by convention 0 is the null outcome and $x(0) = 0$.) We call this vector the *increment representation* of contract x, and denote it $\texttt{incr}(x)$. Note that if x is bounded, then $\texttt{incr}(x) \in [0, 1]^m$. Conversely, call a contract *weakly bounded* if it is monotone and its increment representation lies in $[0, 1]^m$. Such a contract is not necessarily bounded.

We discretize the space of all weakly bounded contracts, viewed as a multi-dimensional unit cube. More precisely, we define the *increment space* as $[0, 1]^m$ with a convention that every vector represents the corresponding weakly bounded contract. Each region in the discretization is a closed, axis-aligned m-dimensional cube in the increment space; henceforth, such cubes are called *cells*. A cell is called *relevant* if it contains at least one candidate contract. A relevant cell is called *atomic* if it contains exactly one candidate contract, and *composite* otherwise.

In each composite cell C, the algorithm will only use two contracts: the *maximal corner*, denoted

$x^+(C)$, in which all increment payments are maximal, and the *minimal corner*, denoted $x^-(C)$, in which all increment payments are minimal. These two contracts are called the *anchors* of C. In each atomic cell C, the algorithm will only use one contract: the unique candidate contract, also called the *anchor* of C.

Virtual width. To take advantage of the problem structure, it is essential to estimate how similar the contracts within a given composite cell C are. Ideally, we would like to know the maximal difference in expected utility:

$$\texttt{width}(C) = \sup_{x,y \in C} \left| U(x) - U(y) \right|.$$

We estimate the width using a proxy, called *virtual width*, which is expressed in terms of the anchors:

$$\texttt{VirtWidth}(C) = \big(V(x^+(C)) - P(x^-(C))\big) - \big(V(x^-(C)) - P(x^+(C))\big). \tag{4.1}$$

This definition is one crucial place where the problem structure is used. (Note that it is *not* the difference in utility at the anchors.) It is useful due to the following lemma (proved in Section A.3.1).

Lemma 4.3.1. *If all types satisfy the FOSD assumption and consistent tie-breaking holds, then* $\texttt{width}(C) \leq \texttt{VirtWidth}(C)$ *for each composite cell* C.

Recall that the proof of this lemma is the only place in the paper where we use our assumptions on worker behavior. All further developments hold for any model of worker behavior which satisfies Lemma 4.3.1.

4.3.2 Description of the algorithm

With these ideas in place, we are now ready to describe our algorithm. The high-level outline of `AgnosticZooming` is very simple. The algorithm maintains a set of *active* cells which cover the increment space at all times. Initially, there is only a single active cell comprising the entire increment space. In each round t, the algorithm chooses one active cell C_t using an upper confidence index and posts contract x_t sampled uniformly at random among the anchors of this cell. After observing the feedback, the algorithm may choose to *zoom in* on C_t, removing C_t from the set of

active cells and activating all relevant quadrants thereof, where the *quadrants* of cell C are defined as the 2^m sub-cells of half the size for which one of the corners is the center of C. In the remainder of this section, we specify how the cell C_t is chosen (the *selection rule*), and how the algorithm decides whether to zoom in on C_t (the *zooming rule*).

Let us first introduce some notation. Consider cell C that is active in some round t. Let $U(C)$ be the expected utility from a single round in which C is chosen by the algorithm, i.e., the average expected utility of the anchor(s) of C. Let $n_t(C)$ be the number of times this cell has been chosen before round t. Consider all rounds in which C is chosen by the algorithm before round t. Let $U_t(C)$ be the average utility over these rounds. For a composite cell C, let $V_t^+(C)$ and $P_t^+(C)$ be the average value and average payment over all rounds when anchor $x^+(C)$ is chosen. Similarly, let $V_t^-(C)$ and $P_t^-(C)$ be the average value and average payment over all rounds when anchor $x^-(C)$ is chosen. Accordingly, we can estimate the virtual width of composite cell C at time t as

$$W_t(C) = \left(V_t^+(C) - P_t^-(C)\right) - \left(V_t^-(C) - P_t^+(C)\right). \tag{4.2}$$

To bound the deviations, we define the *confidence radius* as

$$\texttt{rad}_t(C) = \sqrt{c_{\texttt{rad}} \log(T)/n_t(C)}, \tag{4.3}$$

for some absolute constant $c_{\texttt{rad}}$; in our analysis, $c_{\texttt{rad}} \geq 16$ suffices. We will show that with high probability all sample averages defined above will stay within $\texttt{rad}_t(C)$ of the respective expectations. If this high probability event holds, the width estimate $W_t(C)$ will always be within $4\,\texttt{rad}_t(C)$ of $\texttt{VirtWidth}(C)$.

Selection rule. Now we are ready to complete the algorithm. The selection rule is as follows. In each round t, the algorithm chooses an active cell C with maximal *index* $I_t(\cdot)$. $I_t(C)$ is an upper confidence bound on the expected utility of *any* candidate contract in C, defined as

$$I_t(C) = \begin{cases} U_t(C) + \texttt{rad}_t(C) & \text{if } C \text{ is an atomic cell,} \\ U_t(C) + W_t(C) + 5\,\texttt{rad}_t(C) & \text{otherwise.} \end{cases} \tag{4.4}$$

Zooming rule. We zoom in on a composite cell C_t if

$$W_{t+1}(C_t) > 5\,\texttt{rad}_{t+1}(C_t),$$

i.e., the uncertainty due to random sampling, expressed by the confidence radius, becomes sufficiently small compared to the uncertainty due to discretization, expressed by the virtual width. We never zoom in on atomic cells. The pseudocode is summarized in Algorithm 3.

ALGORITHM 3: `AgnosticZooming`

Inputs: subset $X_{\texttt{cand}} \subset X$ of *candidate contracts*.

Data structure: Collection $\mathcal{A}$ of cells. Initially, $\mathcal{A} = \{ [0,1]^m \}$.

For each round $t = 1$ to T

Let $C_t = \text{argmax}_{C \in \mathcal{A}} \; I_t(C)$, where $I_t(\cdot)$ is defined as in (4.4).

Sample contract x_t u.a.r. among the anchors of C_t. \\ Anchors are defined in Section 4.3.1.

Post contract x_t and observe feedback.

If $|C \cap X_{\texttt{cand}}| > 1$ **a**nd $5\,\texttt{rad}_{t+1}(C_t) < W_{t+1}(C_t)$ **t**hen

$\mathcal{A} \leftarrow \mathcal{A} \cup \{\text{all relevant quadrants of } C_t\} \setminus \{C_t\}$. \\ C is *relevant* if $|C \cap X_{\texttt{cand}}| \geq 1$.

Integer payments. In practice it may be necessary to only allow contracts in which all payments are integer multiples of some amount ψ, e.g., whole cents. (In this case we can assume that candidate contracts have this property, too.) Then we can redefine the two anchors of each composite cell: the maximal (resp., minimal) anchor is the nearest allowed contract to the maximal (resp., minimal) corner. Width can be redefined as a $\sup$ over all allowed contracts in a given cell. With these modifications, the analysis goes through without significant changes. We omit further discussion of this issue.

4.4 Analysis and discussion

We present the main regret bound for `AgnosticZooming`. Formulating this result requires some new, problem-specific structure. Stated in terms of this structure, the result is somewhat difficult to access. To explain its significance, we state several corollaries, and compare our results to prior work.

The main result. We start with the main regret bound. Like the algorithm itself, this regret bound is parameterized by the set $X_{\texttt{cand}}$ of candidate contracts; our goal is to bound the algorithm's regret

with respect to candidate contracts.

Recall that $\texttt{OPT}(X_{\texttt{cand}}) = \sup_{x \in X_{\texttt{cand}}} U(x)$ is the optimal expected utility over candidate contracts. The algorithm's regret with respect to candidate contracts is $R(T|X_{\texttt{cand}}) = T\,\texttt{OPT}(X_{\texttt{cand}}) - U$, where T is the time horizon and U is the expected cumulative utility of the algorithm.

Define the *badness* $\Delta(x)$ of a contract $x \in X$ as the difference in expected utility between an optimal candidate contract and x: $\Delta(x) = \texttt{OPT}(X_{\texttt{cand}}) - U(x)$. Let $X_\epsilon = \{x \in X_{\texttt{cand}} : \Delta(x) \leq \epsilon\}$.

We will only be interested in cells that can potentially be used by `AgnosticZooming`. Formally, we recursively define a collection of *feasible* cells as follows: (i) the cell $[0,1]^m$ is feasible, (ii) for each feasible cell C, all relevant quadrants of C are feasible. Note that the definition of a feasible cell implicitly depends on the set $X_{\texttt{cand}}$ of candidate contracts.

Let $\mathcal{F}_\epsilon$ denote the collection of all feasible, composite cells C such that $\texttt{VirtWidth}(C) \geq \epsilon$. For $Y \subset X_{\texttt{cand}}$, let $\mathcal{F}_\epsilon(Y)$ be the collection of all cells $C \in \mathcal{F}_\epsilon$ that overlap with Y, and let $N_\epsilon(Y) = |\mathcal{F}_\epsilon(Y)|$; sometimes we will write $N_\epsilon(Y|X_{\texttt{cand}})$ in place of $N_\epsilon(Y)$ to emphasize the dependence on $X_{\texttt{cand}}$.

Using the structure defined above, the main theorem is stated as follows. We prove this theorem in Section A.3.2.

Theorem 4.4.1. *Consider the dynamic contract design problem with all types satisfying the FOSD assumption and a constant number of outcomes. Consider* `AgnosticZooming`*, parameterized by some set $X_{\texttt{cand}}$ of candidate contracts. Assume $T \geq \max(2^m + 1, 18)$. There is an absolute constant $\beta_0 > 0$ such that for any $\delta > 0$,*

$$R(T|X_{\texttt{cand}}) \leq \delta T + O(\log T) \sum_{\epsilon = 2^{-j} \geq \delta:\ j \in \mathbb{N}} \frac{N_{\epsilon\,\beta_0}(X_\epsilon|X_{\texttt{cand}})}{\epsilon}. \tag{4.5}$$

Remark 1. As discussed in Section 4.2.1, we target the practically important case of a small number of outcomes. The impact of larger m is an exponential dependence on m in the $O()$ notation, and, more importantly, increased number of candidate policies (typically exponential in m for a given granularity).

Remark 2. Our regret bounds do not depend on the number of worker types, in line with prior work on dynamic pricing. Essentially, this is because bandit approaches tend to depend only on

expected reward of a given "arm" (and perhaps also on the variance), not the finer properties of the distribution.

(4.5) has a shape similar to several other regret bounds in the literature, as discussed below. To make this more apparent, we observe that regret bounds in "bandits in metric spaces" are often stated in terms of covering numbers. (For a fixed collection $\mathcal{F}$ of subsets of a given ground set X, the *covering number* of a subset $Y \subset X$ relative to $\mathcal{F}$ is the smallest number of subsets in $\mathcal{F}$ that is sufficient to cover Y.) The numbers $N_\epsilon(Y|X_{\texttt{cand}})$ are, essentially, about covering Y with feasible cells with virtual width close to ϵ. We make this point more precise as follows. Let an *ϵ-minimal cell* be a cell in $\mathcal{F}_\epsilon$ which does not contain any other cell in $\mathcal{F}_\epsilon$. Let $N_\epsilon^{\texttt{min}}(Y)$ be the covering number of Y relative to the collection of ϵ-minimal cells, i.e., the smallest number of ϵ-minimal cells sufficient to cover Y. Then

$$N_\epsilon(Y) \leq \lceil \log \tfrac{1}{\psi} \rceil \; N_\epsilon^{\texttt{min}}(Y) \text{ for any } Y \subset X_{\texttt{cand}} \text{ and } \epsilon \geq 0, \tag{4.6}$$

where ψ is the smallest size of a feasible cell.[5] Thus, (4.5) can be easily restated using the covering numbers $N_\epsilon^{\texttt{min}}(\cdot)$ instead of $N_\epsilon(\cdot)$.

Corollary: Polynomial regret. Literature on regret-minimization often states "polynomial" regret bounds of the form $R(T) = \tilde{O}(T^\gamma)$, $\gamma < 1$. While covering-number regret bounds are more precise and versatile, the exponent γ in a polynomial regret bound expresses algorithms' performance in a particularly succinct and lucid way.

For "bandits in metric spaces" the exponent γ is typically determined by an appropriately defined notion of "dimension", such as the covering dimension,[6] which succinctly captures the difficulty of the problem instance. Interestingly, the dependence of γ on the dimension d is typically of the same shape; $\gamma = (d+1)/(d+2)$, for several different notions of "dimension". In line with this tradition,

[5] To prove (4.6), observe that for each cell $C \in \mathcal{F}_\epsilon(Y)$ there exists an ϵ-minimal cell $C' \subset C$, and for each ϵ-minimal cell C' there exist at most $\lceil \log \frac{1}{\psi} \rceil$ cells $C \in \mathcal{F}_\epsilon(Y)$ such that $C' \subset C$.

[6] Given covering numbers $N_\epsilon(\cdot)$, the *covering dimension* of Y is the smallest $d \geq 0$ such that $N_\epsilon(Y) = O(\epsilon^{-d})$ for all $\epsilon > 0$.

we define the *width dimension*:

$$\texttt{WidthDim}_\alpha = \inf\left\{d \geq 0 :\ N_{\epsilon\,\beta_0}(X_\epsilon|X_{\texttt{cand}}) \leq \alpha\,\epsilon^{-d} \text{ for all } \epsilon > 0\right\},\ \alpha > 0. \tag{4.7}$$

Note that the width dimension depends on $X_{\texttt{cand}}$ and the problem instance, and is parameterized by a constant $\alpha > 0$. By optimizing the choice of δ in (4.5), we obtain the following corollary.

Corollary 4.4.2. *Consider the the setting of Theorem 4.4.1. For any $\alpha > 0$, let $d = \texttt{WidthDim}_\alpha$. Then*

$$R(T|X_{\texttt{cand}}) \leq O(\alpha\,\log T)\ T^{(1+d)/(2+d)}. \tag{4.8}$$

The width dimension is similar to the "zooming dimension" in Kleinberg et al. [80] and "near-optimality dimension" in Bubeck et al. [17] in the work on "bandits in metric spaces".

4.4.1 Comparison to prior work

Non-adaptive discretization. One approach from prior work that is directly applicable to the dynamic contract design problem is *non-adaptive discretization*. This is an algorithm, call it `NonAdaptive`, which runs an off-the-shelf MAB algorithm, treating a set of candidate contracts $X_{\texttt{cand}}$ as arms.[7] For concreteness, and following the prior work [78–80], we use a well-known algorithm `UCB1` [7] as an off-the-shelf MAB algorithm.

To compare `AgnosticZooming` with `NonAdaptive`, it is useful to derive several "worst-case" corollaries of Theorem 4.4.1, replacing $N_\epsilon(X_\epsilon)$ with various (loose) upper bounds.[8]

Corollary 4.4.3. *In the setting of Theorem 4.4.1, the regret of* `AgnosticZooming` *can be upper-bounded as follows:*

(a) $R(T|X_{\texttt{cand}}) \leq \delta T + \sum_{\epsilon=2^{-j}\geq\delta:\ j\in\mathbb{N}} \tilde{O}(|X_\epsilon|/\epsilon)$, *for each* $\delta \in (0,1)$.

(b) $R(T|X_{\texttt{cand}}) \leq \tilde{O}(\sqrt{T\ |X_{\texttt{cand}}|})$.

[7] To simplify the proofs of the lower bounds, we assume that the candidate contracts are randomly permuted when given to the MAB algorithm.

[8] We use the facts that $X_\epsilon \subset X_{\texttt{cand}}$, $N_\epsilon(Y) \leq N_0(Y)$, and $N_0^{\texttt{min}}(Y) \leq |Y|$ for all subsets $Y \subset X$.

Here the $\tilde{O}()$ notation hides the logarithmic dependence on T and δ.

The best known regret bounds for `NonAdaptive` coincide with those in Corollary 4.4.3 up to poly-logarithmic factors. However, the regret bounds in Theorem 4.4.1 may be significantly better than the ones in Corollary 4.4.3. We further discuss this in the next section, in the context of a specific example.

Bandits in metric spaces. Consider a variant of dynamic contract design in which an algorithm is given a priori information on similarity between contracts: a function $\mathcal{D} : X_{\texttt{cand}} \times X_{\texttt{cand}} \to [0,1]$ such that $|U(x) - U(y)| \leq \mathcal{D}(x,y)$ for any two candidate contracts x, y. If an algorithm is given this function $\mathcal{D}$ (call such algorithm $\mathcal{D}$-*aware*), the machinery from "bandits in metric spaces" [17, 80] can be used to perform adaptive discretization and obtain a significant advantage over `NonAdaptive`. We argue that we obtain similar results with `AgnosticZooming` without knowing the $\mathcal{D}$.

In practice, the similarity information $\mathcal{D}$ would be coarse, probably aggregated according to some predefined hierarchy. To formalize this idea, the hierarchy can be represented as a collection $\mathcal{F}$ of subsets of $X_{\texttt{cand}}$, so that $\mathcal{D}(x,y)$ is a function of the smallest subset in $\mathcal{F}$ containing both x and y. The hierarchy $\mathcal{F}$ should be natural given the structure of the contract space. One such natural hierarchy is the collection of all feasible cells, which corresponds to splitting the cells in half in each dimension. Formally, $\mathcal{D}(x,y) = f(C_{x,y})$ for some f with $f(C_{x,y}) \geq \texttt{width}(C_{x,y})$, where $C_{x,y}$ is the smallest feasible cell containing both x and y.

Given this shape of $\mathcal{D}$, let us state the regret bounds for $\mathcal{D}$-aware algorithms in Kleinberg et al. [80] and Bubeck et al. [17]. To simplify the notation, we assume that the action space is restricted to $X_{\texttt{cand}}$. The regret bounds have a similar "shape" as that in Theorem 4.4.1:

$$R(T|X_{\texttt{cand}}) \leq \delta T + O(\log T) \sum_{\epsilon = 2^{-j} \geq \delta:\ j \in \mathbb{N}} \frac{N^*_{\Omega(\epsilon)}(X_\epsilon)}{\epsilon}, \tag{4.9}$$

where the numbers $N^*_\epsilon(\cdot)$ have a similar high-level meaning as $N_\epsilon(\cdot)$, and nearly coincide with $N^{\texttt{min}}_\epsilon(\cdot)$ when $\mathcal{D}(x,y) = \texttt{VirtWidth}(C_{x,y})$. One can use (4.9) to derive a polynomial regret bound like (4.8).

For a more precise comparison, we focus on the results in Kleinberg et al. [80] . (The regret

bounds in Bubeck et al. [17] are very similar in spirit, but are stated in terms of a slightly different structure.) The "covering-type" regret bound in Kleinberg et al. [80] focuses on balls of radius at most ϵ according to distance $\mathcal{D}$, so that $N^*_\epsilon(Y)$ is the smallest number of such balls that is sufficient to cover Y. In the special case $\mathcal{D}(x,y) = \texttt{VirtWidth}(C_{x,y})$ balls of radius $\leq \epsilon$ are precisely feasible cells of virtual width $\leq \epsilon$. This is very similar (albeit not technically the same) as the ϵ-minimal cells in the definition of $N^{\texttt{min}}_\epsilon(\cdot)$.

Further, the covering numbers $N^*_\epsilon(Y)$ determine the "zooming dimension":

$$\texttt{ZoomDim}_\alpha = \inf\left\{d \geq 0 :\ N^*_{\epsilon/8}(X_\epsilon) \leq \alpha\,\epsilon^{-d} \text{ for all } \epsilon > 0\right\},\ \alpha > 0. \tag{4.10}$$

This definition coincides with the covering dimension in the worst case, and can be much smaller for "nice" problem instances in which X_ϵ is a significantly small subset of $X_{\texttt{cand}}$. With this definition, one obtains a polynomial regret bound which is version of (4.8) with $d = \texttt{ZoomDim}_\alpha$.

We conclude that `AgnosticZooming` essentially matches the regret bounds for $\mathcal{D}$-aware algorithms, despite the fact that $\mathcal{D}$-aware algorithms have access to much more information.

4.5 Summary

Motivated by applications to crowdsourcing markets, we define the *dynamic contract design problem*, a multi-round version of the principal-agent model with unobservable strategic decisions. We treat this problem as a multi-armed bandit problem, design an algorithm for this problem, and derive regret bounds which compare favorably to prior work. Our main conceptual contribution, aside from identifying the model, is the adaptive discretization approach that does not rely on Lipschitz-continuity assumptions. We provably improve on the uniform discretization approach from prior work, both in the general case and in some illustrative special cases. These theoretical results are supported by simulations.

We believe that the dynamic contract design problem deserves further study, in several directions that we outline below.

1. It is not clear whether our provable results can be improved, perhaps using substantially different algorithms and relative to different problem-specific structures. In particular, one needs to establish

lower bounds in order to argue about optimality; no lower bounds for dynamic contract design are currently known.

2. Our adaptive discretization approach may be fine-tuned to improve its performance in practice. In particular, the definition of the "index" $I_t(C)$ of a given feasible cell C may be re-defined in several different ways. First, it can use the information from C in a more sophisticated way, similar to the more sophisticated indices for the basic K-armed bandit problem; for example, see Garivier and Cappé [46]. Second, the index can incorporate information from other cells. Third, it can be defined in a "smoother", probabilistic way, e.g., as in Thompson Sampling [108].

3. Deeper insights into the structure of the (static) principal-agent problem are needed, primarily in order to optimize the choice of $X_{\texttt{cand}}$, the set of candidate contracts. The most natural target here is the uniform mesh $X_{\texttt{cand}}(\epsilon)$. To optimize the granularity ϵ, one needs to upper-bound the discretization error $\texttt{OPT}(X_{\texttt{cand}}) - \texttt{OPT}(X_{\texttt{cand}}(\epsilon))$ in terms of some function $f(\epsilon)$ such that $f(\epsilon) \to 0$ as $\epsilon \to 0$. The first-order open question is to resolve whether this can be done in the general case, or provide a specific example when it cannot. A related open question concerns the effect of increasing the granularity: upper-bound the difference $\texttt{OPT}(X_{\texttt{cand}}(\epsilon)) - \texttt{OPT}(X_{\texttt{cand}}(\epsilon'))$, $\epsilon > \epsilon' > 0$, in terms of some function of ϵ and ϵ'. Further, it is not known whether the optimal mesh of contracts is in fact a uniform mesh.

Also of interest is the effect of restricting our attention to monotone contracts. While we prove that monotone contracts may not be optimal (Appendix A.3.5), the significance of this phenomenon is unclear. One would like to characterize the scenarios when restricting to monotone contracts is alright (in the sense that the best monotone contract is as good, or not much worse, than the best contract), and the scenarios when this restriction results in a significant loss. For the latter scenarios, different algorithms may be needed.

4. A much more extensive analysis of special cases is in order. Our general results are difficult to access (which appears to be an inherent property of the general problem), so the most immediate direction for special cases is deriving lucid corollaries from the current regret bounds. In particular, it is desirable to optimize the choice of candidate contracts. Apart from "massaging" the current results, one can also design improved algorithms and derive specialized lower bounds. Particularly

appealing special cases concern supply distributions that are mixtures of a small number of types, and supply distributions that belong to a (simple) parameterized family with unknown parameter.

Going beyond our current model, a natural direction is to incorporate a budget constraint, extending the corresponding results on dynamic task pricing. The main difficulty for such settings is that a *distribution* over two contracts may perform much better than any fixed contract; see Badanidiyuru et al. [10] for discussion. Effectively, an algorithm needs to optimize over the distributions. As a first step, one can use non-adaptive discretization in conjunction with the general algorithms for bandits with budget constraints (sometimes called "bandits with knapsacks" [3, 10]). However, it is not clear how to choose an optimal mesh of contracts (as we discussed throughout the paper), and this mesh is not likely to be uniform (because it is not uniform for the special case of dynamic task pricing with a budget [10]). The eventual target in this research direction is to marry adaptive discretization and the techniques from prior work on "bandits with knapsacks."

CHAPTER 5

Modeling Human Behavior

5.1 Overview

In the previous two chapters, we assume workers are strategic and respond to incentives rationally. However, this may not be the case in real-world settings, especially when the amount of payments is relatively small in crowdsourcing markets. In this chapter, we study the causal effects of financial incentives on the quality of crowdwork in real-world environments. We focus on performance-based payments (PBPs), bonus payments awarded to workers for producing high quality work. We design and run randomized behavioral experiments on the popular crowdsourcing platform Amazon Mechanical Turk with the goal of understanding when, where, and why PBPs help, identifying properties of the payment, payment structure, and the task itself that make them most effective.

Previous empirical studies of performance-based payments in crowdsourcing markets have produced mixed and somewhat contradictory recommendations. Harris [57] and Yin et al. [117] suggested that PBPs can improve work quality, while Shaw et al. [102] found no improvement and Yin et al. [116] found no difference in quality when varying bonus size.

Our results explain these disparities in prior work. Furthermore, we show how to generalize previous findings beyond the particular tasks that were studied. We design and run experiments with the goal of understanding not just whether PBPs improve work quality for a specific task or bonus size, but *when*, *why*, and *where* they improve work quality. We identify properties of the payment, payment structure, and the task itself that make PBPs effective.

In our experiments, we first identified a task (proofreading an article) for which PBPs improve workers' performance. We tested the robustness of this finding by varying the payment structure and amount. We found that using PBPs with a wide range of quality thresholds improved work

quality provided the bonus awarded for exceeding the threshold was sufficiently high. We also found that even when standard, unconditional payments are used and no explicit acceptance criteria is specified, workers may behave as if the payments are *implicitly* performance-based since they believe their work may be rejected if its quality is sufficiently low.

We examined potential reasons why PBPs improve quality. We found that simply increasing the amount of the base payment without offering any bonus significantly improved quality, contradicting several previous studies [6, 20, 54, 89, 97]. However, PBPs led to improved quality and lower cost compared to a guaranteed payment of the same amount. We also found that whether the opportunity to receive a bonus or higher base payment is revealed before or after the task is accepted does not make a difference in the quality of crowdwork, ruling out the possibility that the increased quality we observed was due (at least in part) to the reciprocity caused by workers' joy at receiving an unexpected bonus, as discussed by Gilchrist et al. [54].

Finally, we investigated which properties of a particular *task* allow PBPs to have an effect. We examined the conjecture that PBPs are more likely to improve quality on *effort-responsive* tasks, tasks for which workers can produce higher quality work by exerting additional effort. We ran experiments on four different tasks. By taking the amount of time workers spent on the task as the proxy measure for their effort, we found that additional effort was correlated with improved quality for the tasks for which PBPs helped, but not for the tasks for which PBPs did not help. This observation yields a simple method for requesters to determine whether or not a given task is likely to benefit from PBPs.

Based on our experimental results, we propose a simple theoretical model of worker behavior. The model is a variant of the standard principal-agent model from economics that additionally incorporates workers' subjective beliefs about the quality of work required to be paid. We show that this model can be used to explain our key empirical observations which cannot be explained using the principal-agent model alone. This model may be useful as a more realistic foundation for future theoretical work on crowdsourcing markets.

5.1.1 Related Work

Performance-based payments have been studied extensively outside of crowdsourcing markets. Lazear [83] conducted a highly influential analysis of observational data from an autoglass company that switched from paying workers a fixed hourly rate to paying workers based on the number of units installed, and showed that workers' performance significantly improved when payments were contingent on work done. As another example, Gneezy and Rustichini [55] gave college students fifty questions from IQ tests to answer and found that their performance increased when they received a bonus for answering questions correctly as long as the bonus was sufficiently high. On the other hand, when the bonus payment was very small, quality decreased compared with not offering a bonus at all. The psychology literature suggests that this is perhaps due to a decrease in workers' intrinsic motivation (enjoyment, responsibility, pride) for performing well, though this theory is not universally accepted [23, 41, 43].

Camerer and Hogarth [22] performed a meta-analysis of 74 papers examining the effect of using payments contingent on performance in lab experiments. They showed that using payments contingent on performance improved the average performance of subjects for tasks in which increased effort leads to improved performance, such as memory or recall tasks, clerical tasks such as coding words or building things, and problem-solving tasks. Similar meta-analyses were performed by Jenkins Jr. et al. [71], Bonner et al. [16], and Hertwig and Ortmann [58]. We build on and extend this work by showing that PBPs have a causal impact on work quality in a field setting, specifically, a crowdsourcing environment in which eliciting high quality work is a main concern for requesters. Moreover, we give evidence to support the conjecture of Camerer and Hogarth [22] that PBPs work in tasks that are effort-responsive.

The results of studies on performance-based payments in crowdsourcing markets have been mixed and sometimes discouraging. On the positive side, in an early workshop paper, Harris [57] showed that when asking workers to evaluate the relevance of resumes, PBPs increased both performance and the time workers spent on the task. In very recent work, Yin et al. [117] studied a setting in which workers switched back and forth between two types of tasks, and showed that PBPs improved performance, especially when used immediately after a task switch.

On the negative side, Shaw et al. [102] compared fourteen different incentive schemes, including four using PBPs,[1] and saw little variation in their effects on quality of work. However, the bonuses offered were extremely small, only 10% ($0.03) on base payments of $0.30. One might hypothesize that the lack of effect stems from decreased intrinsic motivation as in Gneezy and Rustichini [55], or that obtaining the small bonus is simply not worth the costly additional effort that would be required. More recently, Yin et al. [116] studied the effect of varying the bonus size of PBPs and found that the size of the bonus did not impact the quality of work. In their case, all bonuses offered were large compared with the base payment ($0.04, $0.08, $0.16, or $0.32 on a base payment of $0.01 per task), so it is possible that even their smallest bonus was large enough to elicit the workers' maximum effort; since Yin et al. did not include a treatment without PBPs, there is no way to know whether PBPs boosted quality compared with offering base payments alone. In Section 5.3.2, we give a unifying explanation for these results. We show small bonuses in a PBP result in little to no effect on work quality, and that it can be hard to detect the effect between two large PBP bonuses. However, the overall trend is that using PBPs with sufficiently high bonuses yields better quality work.

Several additional studies have examined financial incentives in crowdsourcing markets. Horton and Chilton [65] empirically estimated workers' reservation wages and found that many workers aim to hit payment targets (such as multiples of $0.05). Mason and Watts [89] found that paying workers more increased the number of tasks workers chose to complete, but did not increase performance on each task. That increased pay did not increase performance was observed by other authors as well [20, 54, 85, 97]. In this paper, we found that paying workers more can actually increase their performance for some types of tasks. This disparity can be explained by considering workers' subjective beliefs on how much work they must do to get their work accepted. In prior work, workers either already performed well even with low pay since the tasks were easy or were given additional instructions which could have primed their subjective beliefs. In our experiments, workers are uncertain about how much work they should do to get paid. Therefore, they are willing

[1]The differences in these four treatments were whether payments were described in terms of rewards for high quality or punishments for low quality, and whether quality was measured objectively or in terms of agreement with other workers' responses. Of these, the only treatment that resulted in performance statistically significantly different than the control was punishing workers when their responses did not agree with others.

to produce higher quality work to increase their chance of having their work accepted when the payments are higher.

Gilchrist et al. [54] showed that the manner in which the payment is presented can influence quality. In particular, they found that on oDesk, a crowdsourcing market for larger tasks, initially telling workers they would receive $3 per hour and increasing this payment to $4 after the job was accepted led to higher performance than paying either $3 or $4 per hour without the surprise. We show in Section 5.3.3 that this result did not translate to the Mechanical Turk setting.

Ho et al. [62] studied the algorithmic problem of adaptively optimizing PBPs in crowdsourcing markets, modeling the problem as a dynamic variant of the standard principal-agent model from contract theory. Their model assumed that each worker chooses how much effort to exert in order to maximize his expected utility, which is simply his expected payment minus the cost of his effort. In Section 5.4, we propose a variant of this worker model that is in line with our experimental observations. The results of Ho et al. [62] still apply when our worker model is used in place of theirs.

5.2 Preliminaries

Before describing our experiments, we briefly describe the setting in which they were conducted. All of our experiments were run on Amazon Mechanical Turk[2] (henceforth MTurk), one of the most popular crowdsourcing platforms. On MTurk, requesters post tasks (HITs) for workers to complete. When a worker browses a task, he sees a description of the work to be done along with the amount of money that the requester has offered as a base payment. A worker can then choose whether to accept the task. After the worker completes the chosen task, the requester may evaluate the worker's submission and choose to either approve or reject the work. Each worker has an approval rating which is simply the fraction of HITs he has submitted that have been accepted. If the work is rejected, the worker is not paid and his approval rating, which serves as a *de facto* reputation score, suffers. If the work is accepted, the worker receives the base payment for the task and his approval

[2]https://mturk.com

rating increases. Requesters also decide at this time whether or not to award a bonus payment on top of the base, and how much of a bonus to award. The possibility of such a bonus may or may not be included in the task description.

HITs can also have qualifications associated with them. Only those workers who have the appropriate qualifications can do a HIT. These qualifications are specified by the requester. For example, we used a geographic qualification to restrict our tasks to workers located in the United States, and used qualifications to disallow workers from completing the same type of task more than once. We also used qualifications for random assignment in one of our experiments; how and why we did this is described in Section 5.3.3.

Our experiments focus on the use of threshold-based PBPs. A threshold-based PBP is specified by a base payment, a bonus payment, and a threshold. The base payment specifies the amount of money a worker receives from completing the task; this is fixed at $\$0.50$ USD in all of our experiments.[3] The bonus payment specifies the amount the worker can potentially receive as a bonus. The threshold determines what the worker must do in order to obtain the bonus.

All of the experiments in this paper were approved by the Microsoft Research IRB.

5.3 Behavioral Experiments

5.3.1 Does PBP Work?

Our first experiment was designed with two goals in mind. The first was to verify that PBPs can lead to higher quality crowdwork and identify a task for which this happens. The second was to determine if there exists what we call an *implicit PBP effect*: even if the requester offers a guaranteed payment, MTurk workers have subjective beliefs on the quality of work they must produce in order to receive this payment, and therefore behave as if the payments were (implicitly) performance-based. We measure these subjective beliefs by the difference in the quality of crowdwork when base payments are explicitly guaranteed, effectively resetting the workers' subjective beliefs, compared to when

[3]The base payment of $\$0.50$ was chosen so that workers could obtain a \$6 hourly rate from the base payment alone with a reasonable amount of effort in each of our tasks.

base payments are not guaranteed.

Experiment Design

In this experiment, workers were asked to proofread an article of 500 to 700 words and correct spelling errors. For each article, we randomly inserted 20 typos from a list of common spelling errors. Workers were asked to input the line number of each typo, the misspelled word, and the correct spelling of the word.

This task has two key properties. First, we would expect that workers could produce better work by exerting more effort—the more carefully a worker reads or the more passes a worker takes over the text, the more typos he will find—and that this would open up the possibility of PBPs improving quality. (We study this conjecture in more detail in Section 5.3.4.) Second, since we injected the typos into the text, the quality of each worker's output could be measured objectively, though this was not known to the workers.

Before accepting the HIT, each worker saw a preview consisting of the instructions, an example article with typos, and the base payment of $\$0.50$ USD. The preview was the same for all workers. After workers accepted the HIT, they were randomly assigned to different treatments and then shown treatment-specific instructions, when applicable. Our experiment had a 2×3 design, with 2 treatments governing the base payment and 3 treatments governing the bonus payment (if any). We discuss the bonus treatments first:

- *No Bonus:* This is the control group. It had no bonus and no mention of a bonus.
- *Bonus for All:* All workers earned a $\$1$ bonus after submitting the HIT.
- *PBP:* Workers earned a $\$1$ bonus if they found 75% of the typos found by the other workers.

The purpose of the control group was to allow us to measure the baseline number of typos workers found. The purpose of the Bonus for All treatment was to test if simply paying more resulted in higher quality work. The purpose of the PBP bonus treatment was to test if specifically incentivizing for quality improved the number of typos found.

As mentioned, we are also interested in whether workers have subjective assumptions on how

much effort they must exert to get their work accepted. Workers may be afraid that if they do not find a sufficient number of typos their work will be rejected, resulting in no pay and a negatively affected MTurk reputation. To estimate this, we designed a treatment in which workers were explicitly guaranteed acceptance provided that they completed a very small amount of work. We had two treatments for the base payment:

- *Non-Guaranteed:* There were no extra instructions. This is the control and emulates most MTurk tasks.
- *Guaranteed:* Workers were told they would get paid if they found at least one typo.

The first typo appeared before line 3 in each article. Thus a worker would only have to do a trivial amount of work to ensure they got paid in the guaranteed base treatment.

Results

The HIT was completed by 1,000 unique workers, who were each assigned uniformly to one of the six treatments. We conducted a chi-squared test to check for significant differences in the number of participants who finished the six treatments and found none ($p = 0.38$). The primary dependent variable (or outcome measure) we were interested in is the number of true typos found. In the analysis we made six comparisons that we spell out below. We performed this analysis using an ANOVA with one-sided, planned comparisons [100] and report p-values that have been corrected for these multiple (six) comparisons. The results of this experiment are shown in Figure 5.1 and described below.

PBPs improve quality. To determine whether PBPs increase quality for this task, we focus on the non-guaranteed base treatments since almost all HITs on MTurk do not explicitly guarantee any kind of acceptance criteria. Workers in the PBP bonus treatment found on average 1.3 more typos than workers in the No Bonus treatment ($p = 0.042$), showing that PBPs did improve quality for this task.

All payment schemes may be implicitly performance-based. In the No Bonus treatment, the guaranteed base resulted in 1.5 fewer typos found on average compared with the non-guaranteed

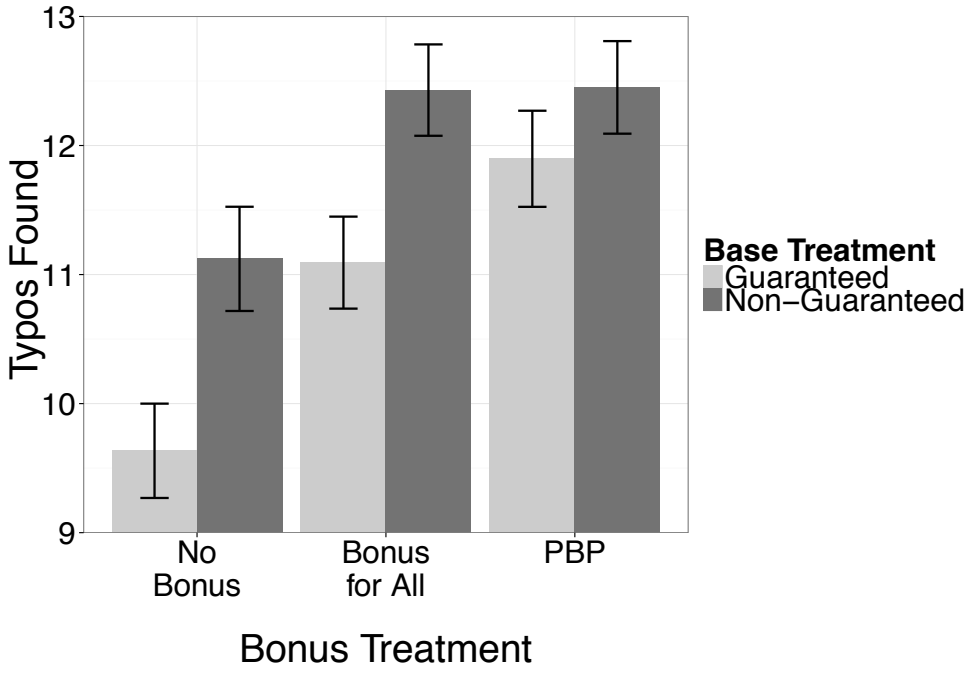


Figure 5.1: The effect of different payment schemes on work quality in the proofreading task. Error bars indicate the mean $\pm$ one standard error.

base ($p = 0.015$). Similarly, in the Bonus for All treatment, the guaranteed base resulted in 1.3 fewer typos found on average ($p = 0.024$). While there may be other explanations, this suggests that workers do have subjective beliefs on the amount of work that needs to be done for their work to be accepted, lending support to our conjecture that payments on MTurk are already implicitly performance-based. We discuss this further in Section 5.4.

In the PBP bonus treatment, we did not see a significantly different effect between the guaranteed base and non-guaranteed base treatments. We offer two related explanations of this finding. First, the only way to grant a bonus using the MTurk API is to first accept the work. This means that in the PBP bonus treatment, workers would likely believe that finding 75% of typos would almost certainly result in their work being accepted, already altering their subjective beliefs. Second, the treatment might have made this 75% threshold more salient to the workers. This gave a clear goal for the workers to strive for.

Simply paying more improves quality. Focusing again on the non-guaranteed base treatment, workers in the Bonus for All treatment found on average 1.3 more typos than workers in the No

Bonus treatment ($p = 0.036$). Thus offering an unconditional bonus—which is essentially just paying more—increased quality.

This finding is perhaps surprising since it appears to contradict the results of prior work [20, 89, 97]. We give two potential explanations. First, since the announcement of the bonus came after workers accepted the HIT, the workers may be exhibiting reciprocity by doing higher quality work [54], rewarding the requester for this pleasant surprise. We further test and refute this hypothesis in Section 5.3.3. Second, this could be explained by the implicit PBP effect described above. That is, workers might have subjective beliefs about the number of typos they must find to get paid. If we increase the bonus payment, workers might be willing to put in more effort to increase their probability of earning this higher amount.

This observation is not inconsistent with previous work. In most prior work, either easy tasks were chosen which might cause workers to perform well even for low pay [20, 97] or additional instructions or tutorials were provided which may have primed workers' subjective beliefs [89].

PBPs can save money compared with high unconditional payments. In the non-guaranteed base treatment, the difference in the number of typos found in the PBP and Bonus for All treatments is not significant. Both resulted in higher quality work than the control. However, we spent much less money on the PBP treatment. We paid each worker $\$1.50$ in the Bonus for All treatment, while we paid each worker only $\$0.97$ on average in the PBP treatment with non-guaranteed base and $\$0.96$ on average in the PBP treatment with guaranteed base. Therefore, it may still be advantageous for requesters to offer PBPs even if they could achieve the same quality work with unconditional payments.

5.3.2 When Does PBP Work?

Having established that PBPs can improve quality for the proofreading task, we investigated the effect of varying two parameters of the payment scheme, the bonus threshold and the bonus amount, to better understand when PBPs help.

Bonus Thresholds: Experiment Design

We first tested the effect of varying the threshold of quality that must be met in order for workers to receive the bonus. We used the same proofreading task described in Section 5.3.1, with the same base payment of $\$0.50$ and bonus of $\$1$. Workers were randomly assigned to treatments in which they were told they could earn the bonus if they found at least 5 typos or at least 25%, 75%, or 100% of the typos found by the other workers. In the control, workers did not receive any bonus or see any mention of a bonus.

Bonus Thresholds: Results

The results from 585 unique workers are presented in Figure 5.2. As before, we ran an ANOVA with one-sided, planned comparisons [100] and report p-values that have been corrected for the multiple (five) comparisons we describe below.

PBPs improve quality for a wide range of bonus thresholds. Improvements in quality over the control can be observed in both the 25% and 75% treatments. The 25% treatment resulted in workers finding, on average, 1.1 more typos than the control ($p = 0.082$). Similarly, the 75% treatment resulted in workers finding, on average 1.2 more typos, than the control ($p = 0.049$). We conjecture that setting a threshold anywhere between 25% and 75% would yield similar results, so the improvements from PBPs are not overly sensitive to the threshold. However, the 100% typo condition was neither significantly different than the 75% treatment or control, so our results suggest that if the bonus threshold is set too high, then workers' average performance slightly decreases. This could be due to some workers giving up because they do not believe the bonus is attainable.

Subjective beliefs on the quality thresholds can improve work. Since each article contains 20 typos in total, 25% of the typos found by other workers is at most 5 typos and will be exactly 5 typos if the number of workers is sufficiently large. In this sense, the two thresholds are roughly equivalent. However, workers in the 25% treatment performed much better than workers in the 5 typos treatment. Workers in the 25% treatment found, on average, 1.9 more typos than those in the 5 typo treatment ($p < .001$). An analysis of the line numbers in which typos were found showed that

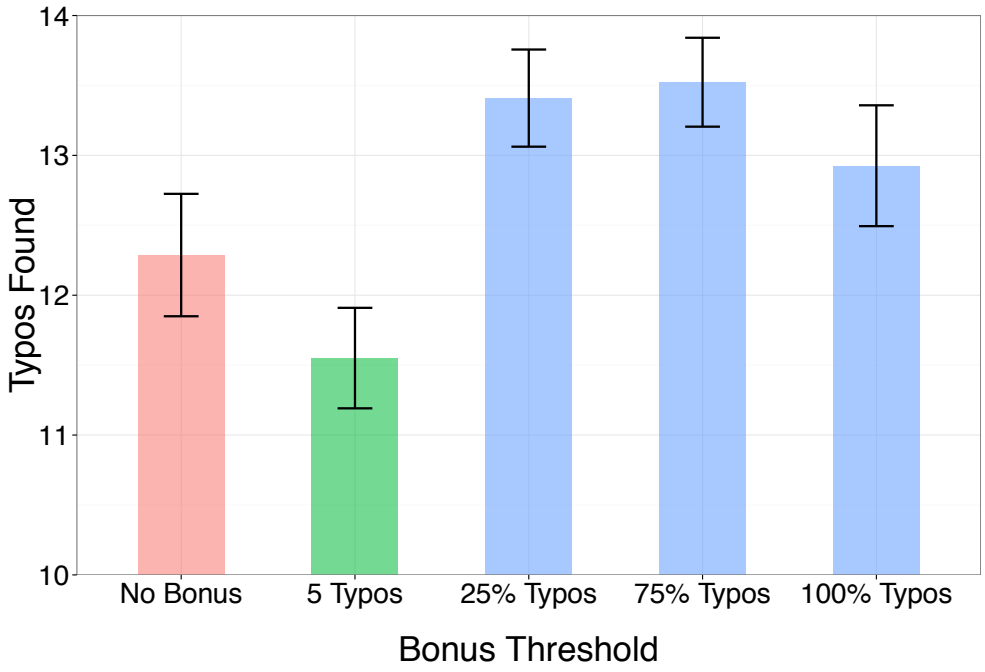


Figure 5.2: The effect of the bonus threshold on work quality in the proofreading task. Error bars indicate the mean $\pm$ one standard error.

workers in the 5 typo treatment did not stop reading before workers in the 25% treatment; however, there may be a variety of other explanations for this. For example, it is possible that workers in the 25% group had different subjective beliefs and thought that they would need to find more than 5 typos to receive the bonus. After all, workers did not know the total number of typos in the article.

Bonus Amounts: Experiment Design

We next examined the effect of varying the bonus amount. We used the same proofreading task with a base payment of \$0.50. Workers were assigned to treatments in which they could earn either \$0.05, \$0.50, or \$1 if they found 75% of the typos found by other workers. Once again, workers in the control did not receive or see any mention of a bonus.

As an implementation detail, we note that these experiments were run simultaneously with those described in Section 5.3.2, allowing us to share two treatments (the control and the \$1 for 75%) and run only seven treatments in total instead of nine. We excluded workers who had already participated in the experiment from Section 5.3.1. We collected results from 815 unique workers

assigned uniformly to the seven treatments. A chi-squared test showed no significant differences between the number of workers who completed the seven treatments ($p = 0.23$)

Bonus Amounts: Results

We collected data from 451 unique workers. Figure 5.3 shows the overall trend: PBPs lead to higher quality work only when the bonus is sufficiently large, but increasing the bonus amount has diminishing returns. Indeed, regressing the number of typos found on the bonus amount shows that an extra \$1 of bonus results in finding 1.4 more typos $(p = 0.002)$ on average.

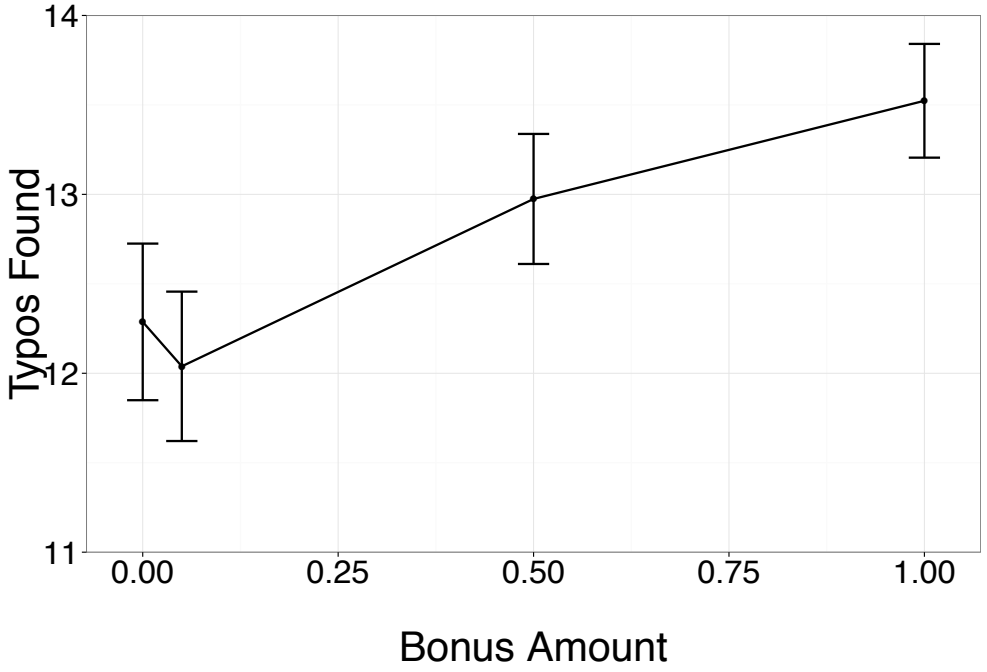


Figure 5.3: The effect of the bonus amount on work quality in the proofreading task. Error bars indicate the mean $\pm$ one standard error.

This may help to explain previous negative results on PBPs in crowdsourcing markets. Shaw et al. [102] reported little or no quality improvement using PBPs compared with fixed payments. However, they offered a bonus payment of only $\$0.03$, 10% of their $\$0.30$ base. As we observe from the leftmost two points in Figure 5.3, PBPs do not improve quality when the bonus is very small. Yin et al. [116] tested PBPs with several bonus sizes and reported that bonus size alone did not significantly impact quality. However, their study did not include a control with fixed payments,

and the bonus sizes that they tested were all significantly larger than their base payment ($0.04, $0.08, $0.16, and $0.32 on a base payment of $0.01). As we observe from the rightmost two points in Figure 5.3, increasing the bonus size only leads to minor improvements in quality once the bonus is already sufficiently large. It is only when we view the whole picture that we can see that PBPs do help.

Taken as a whole, the results in this section show that PBPs improve quality for a wide range of possible thresholds, provided that the requester offers a bonus that is high enough to make the extra reward salient.

5.3.3 Why Does PBP Work?

There are two primary motivations for our next experiment. First, we wanted to verify that PBPs are useful in other tasks beyond finding typos. Second, we wanted to explore potential reasons why PBPs work. In particular, as pointed out in Section 5.3.1, simply increasing the amount of the bonus payment led to almost as much of an improvement as using PBPs in the proofreading experiment. While it could be that workers are responding rationally to the provided incentives, it could also be the case that workers are increasing their effort due to a reciprocity effect; workers are pleasantly surprised to discover the opportunity to receive a (performance-based or unconditional) bonus after accepting the HIT, and reward the requester for this kind action by working harder. Indeed, Gilchrist et al. [54] found, in a different crowdsourcing context, that workers who accept a task and then receive an unexpected bonus do higher quality work than workers who are paid the same amount total but are told up front. This experiment is designed to test whether this "unexpected bonus effect," is the (partial) cause of the observed increases in performance using PBPs.

Experiment Design

In this task, workers were shown twenty pairs of images. Ten of the pairs were identical images, while the other ten pairs contained minor differences. Workers were asked to specify whether each pair was identical or not, and were not told how many pairs of images were identical in advance. Again, this task has two key properties we desire. First, we speculated that workers would be more

likely to spot the differences between images if they spent more time and effort looking. Second, we can objectively measure the quality of workers' output by the number of correctly answered pairs. A similar task was used in experiments by Yin et al. [116].

To test our research questions, we wished to vary the bonus amount and bonus rules as well as the amount of the base payment. Obviously, if we launched two HITs that differed only in the base payment amount, the majority of workers would choose the HIT with the higher base payment, resulting in selection bias. To avoid this, we used the following method for randomly assigning treatments. We first posted a qualification HIT in which workers were paid a small amount ($\$0.05$) if they agreed to receive notifications about our future tasks. We made it clear that they were under no obligation to do the future tasks. We then randomly assigned the workers who completed this recruitment task to different treatments. For each treatment, we posted a separate HIT. Workers were only qualified to see and do the HIT corresponding to their assigned treatment. Finally, we used the notifyWorkers API call to send notification emails to workers with a link to their assigned HITs (treatments). While others have suggested using qualifications to filter out workers for experiments [26] or recruiting a panel of workers in advance [88], we believe that the approach described here is novel and of independent interest.

We next describe the treatments:

- *Low Base:* The base payment was $\$0.50$. No opportunity for a bonus was given. This was our control.
- *High Base:* The base payment was $\$1.50$. No opportunity for a bonus was given.
- *Unexpected Bonus:* The base payment was $\$0.50$. After accepting the HIT, workers were told they would receive an additional bonus of $\$1$.
- *PBP:* The base payment was $\$0.50$. In addition to the base payment, workers could earn a bonus of $\$1$ if they correctly labeled 80% of the image pairs as identical or not. Workers were informed of the bonus and rules for receiving the bonus before accepting the HIT.

Note that the payment amounts in the High Base and Unexpected Bonus treatments are the same. The difference is only how and when the payments were described.

Results

We randomly chose 800 workers from the pool that completed the qualification HIT and randomly assigned them to the four treatments, 200 workers per treatment. After assigning qualifications corresponding to each treatment, we posted the HITs for each simultaneously and sent each worker a notification with a link to their treatment's HIT. We conducted a chi-squared test to check for significant differences in the number of participants who finished the four treatments and found none ($p = 0.90$). In the analysis we make six comparisons, described below. We did this analysis using an ANOVA with one-sided, planned comparisons [100] and report p-values that have been corrected for these multiple comparisons. The results are shown in Figure 5.4.

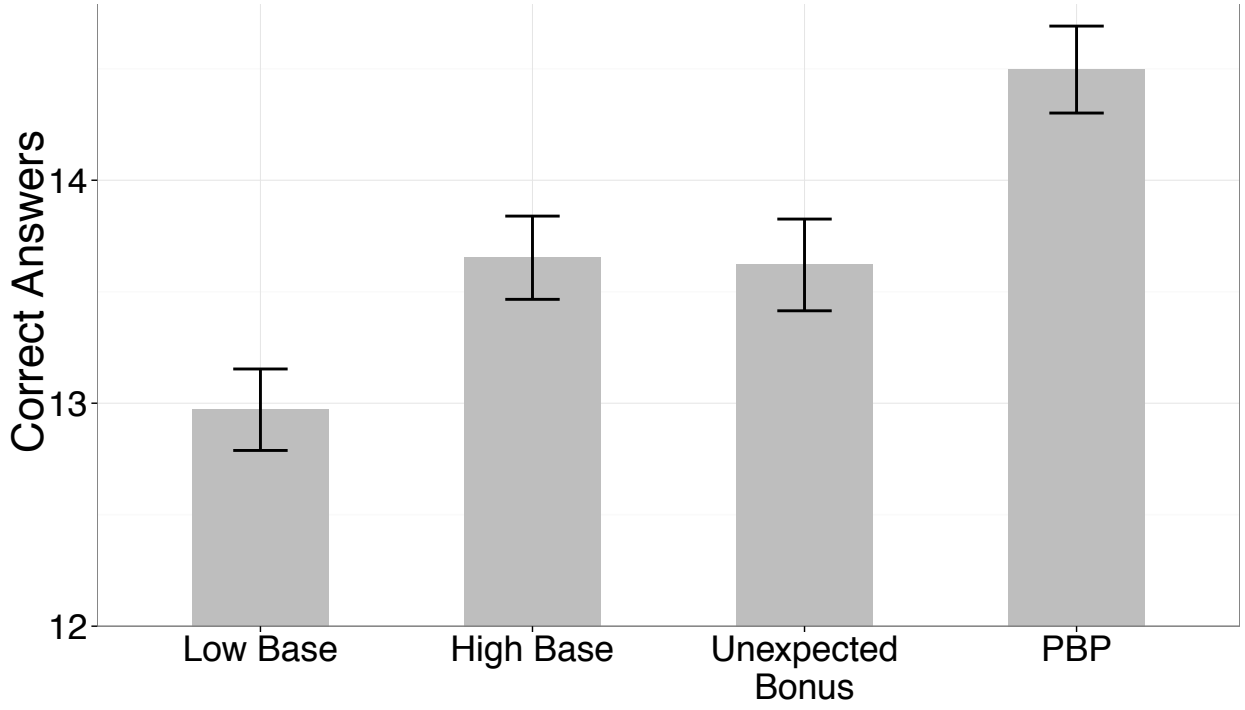


Figure 5.4: The effect of different payment schemes on work quality in the spot the differences task. Error bars indicate the mean $\pm$ one standard error.

Similar to the proofreading experiment described in Section 5.3.1, simply paying more resulted in higher quality work. The High Base treatment had a significantly higher number of correct answers than the Low Base treatment ($p = 0.030$). Similarly, the Unexpected Bonus treatment had a significantly higher number of correct answers than the Low Base treatment ($p = 0.047$). Figure 5.4

shows no significant difference between the High Base and the Unexpected Bonus treatments. This suggests that there was no "unexpected bonus effect" in contrast to Gilchrist et al. [54][4] The absence of any reciprocity effect due to the unexpected bonus suggests that workers were doing better work to increase the probability (according to their prior assumptions) that their work got accepted and thus earn the higher pay.

We also observe that workers in the PBP treatment outperformed workers in all other treatments ($p < 0.005$). This suggests that workers are rational to some degree and are willing to exert more effort to increase their chances of receiving higher payments. Note that in this experiment workers knew *before* they accepted the HIT that they could earn a bonus, in contrast to the experiment described in Section 5.3.1 in which workers were informed of the opportunity to earn a bonus only *after* they accepted the HIT. We have therefore shown that PBPs can work whether or not the opportunity for a bonus is unexpected.

5.3.4 Where Does PBP Work?

We have shown that PBPs incentivize higher quality crowdwork on two specific tasks, proofreading and spotting differences in images. It is natural to ask whether our results generalize, and in particular, what properties of a task open up the possibility of performance improvements with PBPs.

Camerer and Hogarth [22] note that in the context of economics lab experiments, performance-based incentives tend to improve quality for *effort-responsive* tasks, tasks for which it is possible to generate higher quality work by exerting additional effort (presumably without requiring *too much* effort). One might ask if the same is true in a crowdsourcing setting. More specifically, we investigate a hypothesis that whether, and to what extent, a task is effort-responsive is an important reason for whether or not PBPs work for this task. We find an empirical correlation between the two, which we interpret as a strong evidence in favor of this hypothesis. Since it is difficult to directly measure how much effort a worker has put into a task, we use the time a worker spent on a HIT as a

[4]Note that our experimental setting is not the same as theirs. They ran experiments on oDesk for tasks with much longer working hours (3 hours) while our experiment is on MTurk and our tasks only last for on average 8.8 minutes.

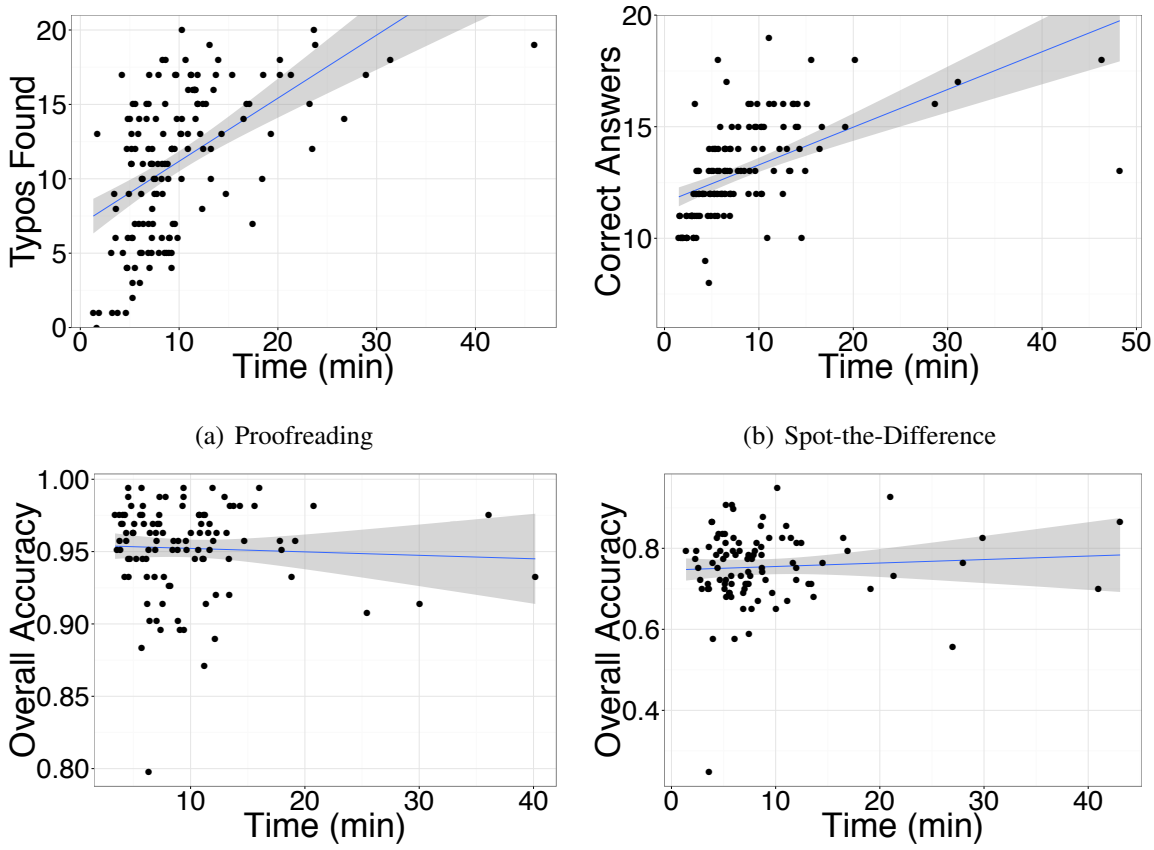
proxy measure for effort, and examine the relationship between time spent and quality of work.

Figures 5.5(a) and 5.5(b) illustrate this correlation for the proofreading and spot-the-difference tasks respectively. Each shows the amount of time that a worker spent on the HIT versus the quality of his work (measured as number of typos found or number of correctly labeled image pairs as before). For the proofreading task, regressing the number of correct typos found on the amount of time a worker spent shows that every minute is correlated with finding another 0.42 typos on average ($p < 0.001$). Similarly, for the spot-the-difference task, regressing the number of pairs correctly identified on the amount of time a worker spent shows that every minute is correlated with another 0.17 correct answers ($p < 0.001$). We see that, in general, workers who spent more time on our tasks generated better quality work. We observe similar trends in all treatments, but include only workers in the control groups in the plots since they are most comparable across tasks. This is evidence that the tasks on which we observed improvements from PBPs are effort-responsive.

To further explore this hypothesis and the generalizability of our results, we examined the effects of PBPs on two additional tasks, handwriting recognition and audio transcription. For consistency with our previous experiments, we maintained a base payment of $\$0.50$ (adjusting the task lengths to maintain an hourly rate of roughly $\$6$) and a bonus of $\$1$ when applicable. Workers were randomly assigned to treatments after accepting the HIT in the same way as in the experiments described in Sections 5.3.1 and 5.3.2.

Handwriting Recognition: Design

For the handwriting recognition task, workers were shown two images containing handwritten text and asked to transcribe the text. The images were collected from the IAM Handwriting Database [87]. One contained 89 words and the other 74. As in our other experiments, workers in the control treatment received only the base payment. We also used a PBP treatment in which workers were told that for one of the images, their transcription would be compared against a gold standard solution and they would receive the bonus if they correctly transcribed 90% of the words in the image. They were not told which image would be used to assess their accuracy. This payment rule was used to make the task appear more like a realistic MTurk task.



(a) Proofreading

(b) Spot-the-Difference

(c) Handwriting Rec.

(d) Audio Transcription

Figure 5.5: Time vs. quality for effort responsive tasks in panels 5.5(a) and 5.5(b), and non-effort responsive tasks in panels 5.5(c) and 5.5(d). The blue lines indicate the regression line and the shaded areas represent the 95% confidence interval around it. Results are similar when outliers are excluded from the analysis.

Handwriting Recognition: Results

Data was collected from 220 workers. As shown in Figure 5.5(c), the quality of work produced was not significantly correlated with the time a worker spent on the task. In other words, this task does not appear to be effort-responsive. Moreover, we did not find a significant difference between the accuracy of workers in the control group versus the PBP treatment via a one-sided t-test. We note that one of the two images given to workers was chosen because it was especially difficult for workers in a pilot study. Restricting our analysis to just this image did show a marginal effect, with workers in the control group averaging 92.3% accuracy and workers in the PBP treatment

averaging 93.5% ($p = 0.055$). Since PBPs only had a small, marginal effect on even the most difficult handwriting recognition image, we conclude that PBPs only have a small effect on this task in general.

One reason why PBPs did not have a strong effect on quality for this task may be that there was a ceiling effect, as discussed in the context of incentives in lab experiments by Camerer and Hogarth [22]. As Figure 5.5(c) shows, the average accuracy over the two articles of the workers in the control group was 95.2%, leaving little room for PBPs to have impact. A related explanation is that most of handwritten words in our data sets were trivial to recognize. Over 80% of words were correctly transcribed by over 90% of workers.

It is possible that PBPs would have improved performance if we had chosen a different threshold. However, since the average performance was already very high, there was little room for experimentation. If a better threshold exists, it would be difficult for a requester to identify.

Audio Transcription: Design

For the audio transcription task, workers were asked to transcribe 10 audio clips, each of which contained approximately 5 seconds of speech. The audio clips were collected from VoxForge[5] and we intentionally chose clips from speakers with heavy accents to increase the difficulty of the task and avoid the ceiling effect. Once again workers in the control treatment received only the base payment. We additionally included three PBP treatments with different thresholds of quality required to receive the bonus. In these treatments, workers were told that their answers for 5 of the 10 clips would be compared against gold standard answers, and that they would receive the bonus if they correctly transcribed 80%, 85%, or 90% of the words respectively. These threshold values were chosen based on quality observed in a pilot experiment on this task in order to cover the range of thresholds that we believed would be most likely to lead to quality improvements with PBPs. We note that audio transcription is one of the most common tasks on MTurk.

[5]http://www.voxforge.org/

Audio Transcription: Results

We collected data from 400 workers. As Figure 5.5(d) shows, quality and time are not significantly correlated. That is, audio transcription does not appear to be effort-responsive. Furthermore, we did not find a significant difference between any of the three PBP treatments and the control group. Since the average accuracy in the control group was only 75.4%, this cannot be fully explained by a ceiling effect as may have been the case for handwriting recognition. One might ask if there are certain hard words for which PBPs did improve performance. We took a closer look at the data and found that this does not appear to be the case. Of the 97 words included in the 10 clips, workers' combined accuracy was better in the 85% threshold PBP treatment compared with the control on 52 words, and better in the control than the 85% threshold PBP treatment on 44; essentially these differences appear to be mostly due to noise. We observe a similar pattern for the other PBP treatments. Additionally, for more than 80% of the words, the percentage of workers correctly transcribing the word when PBPs were offered is close (within $\pm 5\%$) to the percentage in the control group. This could suggest that workers' performance is limited by their abilities and cannot be improved through PBPs.

A Practical Recommendation

While the results in this section are not causal, they are in line with the hypothesis that the extent to which a task is effort-responsive is an important reason for whether or not PBPs help improve quality for this task. This suggests an approach that requesters can use when deciding whether or not to employ PBPs in their own HIT. A requester could run a pilot of their HIT with a small number of workers and a fixed (not performance-based) payment and plot the time that workers spend on the task versus the quality of their work to determine whether and to what extent the task is effort-responsive. A requester may be able to incentivize higher quality using PBPs only if the task is (sufficiently) effort-responsive. In this case, the requester must determine whether the boost in quality is worth the extra cost of PBPs.

5.4 A Theory of Worker Incentives

The worker model in the well-known *principal-agent* framework [82] assumes that a worker chooses a level of effort or quality to maximize his expected utility, which is simply the expected payment he receives minus the expected cost of doing the work at the chosen effort level. In this section, we show that incorporating the worker's subjective beliefs about how much quality is required to earn the base and bonus payments into the worker model allows us to explain the main observations from our experiments in a parsimonious way. Furthermore, it is not clear how one might capture our experimental results using the traditional worker model *without* incorporating the worker's subjective beliefs. Our model can be used to reason about the possible consequences of using performance-based payments.

We assume that a worker views a task and chooses to produce work at a particular quality level in order to maximize his expected utility, defined as his *perceived* expected payment minus the perceived cost of doing the task at a given quality level. In our model, when workers are offered performance-based payments with base payment p and bonus payment b, the worker's expected utility is

$$U_{\texttt{pbp}}(q) = p \Pr(\texttt{base}|q) + b \Pr(\texttt{bonus}|q) - c(q), \tag{5.1}$$

where q is the quality level, $\Pr(\texttt{base}|q)$ and $\Pr(\texttt{bonus}|q)$ denote the worker's perceived probabilities of receiving the base and bonus payments with work of quality q, and $c(q)$ is the worker's perceived cost of producing quality q. We posit that q comes from a totally ordered set, and assume that $\Pr(\texttt{base}|q)$ and $\Pr(\texttt{bonus}|q)$ are non-decreasing in q.

While we typically think of the cost of producing work as positive, capturing the effort the worker must exert to produce work of the chosen quality level, it could in some cases be negative, capturing the subjective intrinsic utility the worker receives from his enjoyment of the task or his satisfaction from a job well done. To allow for such effects, we make no assumptions on the monotonicity of $c(q)$.

We make a minor *consistent tie-breaking* assumption. If multiple quality levels maximize the expected utility for a pair of base and bonus payments, we assume the tie is broken consistently in

the sense that the worker chooses the same quality level for any payments leading to this particular tie.

Consequences of the Worker Model

Given the worker model, we are able to provide a coherent explanation of our key observations, including some that are not explained by the standard principal-agent model. To explain the key observations from our experiments we compare Equation 5.1, which describes the worker's utility when there are both a base payment and a bonus payment, with the utility of a worker with subjective beliefs under the "standard" payment scheme in which workers are offered a base payment only. The standard payment scheme was used in the control group in our experiments. In this case, the worker's expected utility is simply

$$U_{\texttt{std}}(q) = p \Pr(\texttt{base}|q) - c(q). \tag{5.2}$$

Let $q_{\texttt{std}}$ be the quality level chosen by the worker under this utility function, so $q_{\texttt{std}} \in \arg\max_q U_{\texttt{std}}(q)$, and $q_{\texttt{pbp}}$ be the quality chosen under Equation 5.1, so $q_{\texttt{pbp}} \in \arg\max_q U_{\texttt{pbp}}(q)$.

Subjective beliefs about acceptance criteria increase quality. In Section 5.3.1, we experimentally showed that the quality of work produced is higher when workers have subjective beliefs about acceptance criteria than when the base payment is explicitly guaranteed. We called this an *implicit PBP*. It is easy to explain in our model.

Consider the standard setting (no PBPs). With a guaranteed base payment, the worker's utility becomes $p - c(q)$. Let q^* be a maximizer of this expression. It follows that q^* is a minimizer of $c(q)$, thus $c(q^*) \leq c(q)$ for all q. For any $q < q^*$, we have $p \Pr(\texttt{base}|q) \leq p \Pr(\texttt{base}|q^*)$ given the monotonicity of $\Pr(\texttt{base}|q)$. Therefore, for any $q < q^*$, $U_{\texttt{std}}(q) \leq U_{\texttt{std}}(q^*)$. Given the consistent tie-breaking assumption, the worker will not choose to generate work of quality q if $q < q^*$. Therefore, if $q_{\texttt{std}}$ is the maximizer of Equation 5.2 then $q_{\texttt{std}} \geq q^*$. Thus $q_{\texttt{std}}$, the optimal quality level with no guaranteed base payment, is greater than or equal to q^*, the optimal quality level with guaranteed base payment, which we observed experimentally in Figure 5.1. It is possible

to strengthen this result to strict inequality ($q_{\texttt{std}} > q^*$) with a few additional assumptions.[6] Similar arguments can be made for the setting with PBPs.

Higher payments increase quality. The experiments in Sections 5.3.1 and 5.3.2 demonstrated that increasing the base payment (or the unconditional bonus) can increase quality. This is also easy to explain in our model. Consider increasing the base payment in the standard setting (no PBPs). Let q_δ be the maximizer of $U_\delta(q) = (p + \delta)\Pr(\texttt{base}|q) - c(q)$, the utility of the worker if the base payment were $p + \delta$ instead of p, where $\delta > 0$. By optimality of $q_{\texttt{std}}$, for all q,

$$p\Pr(\texttt{base}|q) - c(q) \leq p\Pr(\texttt{base}|q_{\texttt{std}}) - c(q_{\texttt{std}}).$$

Since $\Pr(\texttt{base}|q)$ is non-decreasing in q, $q < q_{\texttt{std}}$ implies $\delta\Pr(\texttt{base}|q) \leq \delta\Pr(\texttt{base}|q_{\texttt{std}})$. Combining the last two inequalities, $U_\delta(q) \leq U_\delta(q_{\texttt{std}})$ for all $q < q_{\texttt{std}}$. Therefore, given the consistent tie-breaking assumption, q_δ, the optimal quality level under higher pay, is greater than or equal to $q_{\texttt{std}}$, the optimal quality level under the standard setting. Again, strict inequality is achievable with the additional assumptions from Footnote 6. Similar arguments can be made for increasing the base payment with PBPs, and for increasing the bonus payment.

These conclusions hold only when uncertainty about receiving the payment is included. They would not hold in the standard principal-agent model, where increasing a guaranteed payment does not increase quality.

Performance-based payments (significantly) increase quality. A key result from our experiments in Sections 5.3.1-5.3.3 is that PBPs can, in fact, increase quality in a significant way. This can be explained from our model too.

We have that the quality improves ($q_{\texttt{pbp}} \geq q_{\texttt{std}}$, or $q_{\texttt{pbp}} > q_{\texttt{std}}$ with additional assumptions) as a special case of the argument above. However, this statement is relatively weak, as it does not say anything about the magnitude of the improvement, i.e., the difference $q_{\texttt{pbp}} - q_{\texttt{std}}$. We would

[6] In multiple places in this section, to make inequalities on quality strict, we could first assume that q takes values on some interval $[q_{\min}, q_{\max}]$ such that $c(q_{\max}) > b + p$, and that $\Pr(\texttt{base}|q)$, $\Pr(\texttt{bonus}|q)$, and $c(q)$ are differentiable on this interval. We must then also assume that the subjective probabilities $\Pr(\texttt{base}|q)$ and $\Pr(\texttt{bonus}|q)$ are strictly increasing in q. In this particular instance, under these assumptions, since $c'(q^*) = 0$, we have $U'_{\texttt{std}}(q^*) > 0$, which implies that q^* cannot be the maximizer of $U'_{\texttt{std}}(q^*)$, so $q_{\texttt{std}} > q^*$.

like this difference to be large; a requester might not want to take on the extra costs of PBPs if the increase in quality is small. In order to gain some intuition for when it is or is not possible to obtain a sufficiently large improvement in quality, we consider several special cases as examples.

As a first example, consider the case in which the worker does not have fine-grained control over the precise quality of his work, but can only choose between two options: high-quality, denoted q_{high}, or low quality, q_{low}. Then PBPs work if and only if the worker's optimal quality level is q_{low} without PBPs, and q_{high} with PBPs. In formulas, $U_{\texttt{std}}(q_{high}) < U_{\texttt{std}}(q_{low})$ and $U_{\texttt{pbp}}(q_{high}) > U_{\texttt{pbp}}(q_{low})$, or

$$\begin{aligned} p\left(\Pr(\texttt{base}|q_{high}) - \Pr(\texttt{base}|q_{low})\right) &< c(q_{high}) - c(q_{low}) \\ &< p\left(\Pr(\texttt{base}|q_{high}) - \Pr(\texttt{base}|q_{low})\right) + \\ &\quad b\left(\Pr(\texttt{bonus}|q_{high}) - \Pr(\texttt{bonus}|q_{low})\right). \end{aligned}$$

In words, the extra cost to produce high-quality work must be less than the extra benefit the worker would receive in terms of expected payments if the bonus is included, and bigger than the extra benefit he would receive with standard payments. Examining this expression gives us intuition about when we might expect PBPs to help. First, $c(q_{high}) - c(q_{low})$ cannot be too large. It must be possible for the worker to substantially increase his quality with additional effort (and at a reasonable cost)—essentially, the task must be effort-responsive, as conjectured in Section 5.3.4. Second, $c(q_{high}) - c(q_{low})$ cannot be too small (in particular, compared with $\Pr(\texttt{base}|q_{high}) - \Pr(\texttt{base}|q_{low})$ and p) or PBPs are unnecessary to achieve high quality. This is a partial explanation for why we did not see improvements from PBPs in the handwriting recognition task when the cost of producing high quality was already small. Third, the bonus b must be set large enough. This could partially explain our observation that PBPs did not help in the proofreading task with a very small bonus. Finally, the difference $\Pr(\texttt{bonus}|q_{high}) - \Pr(\texttt{bonus}|q_{low})$ must be high enough. This could explain why we observed that PBPs did not help when the threshold for receiving a bonus is set too low.

As a second tractable example, suppose that the worker has fine-grained control over quality,

but has no uncertainty over whether the bonus will be obtained, i.e.,

$$\Pr(\texttt{bonus}|q) = \begin{cases} 1 & \text{if } q \geq \bar{q} \\ 0 & \text{otherwise} \end{cases}$$

for some threshold value $\bar{q}$. It is easy to show that either $q_{\texttt{pbp}} = q_{\texttt{std}}$ (if the worker prefers to do less work and pass up the bonus) or $q_{\texttt{pbp}} \geq \bar{q}$ (if the worker prefers to do more work to receive the bonus). PBPs are useful if and only if $\bar{q}$ is sufficiently higher than $q_{\texttt{std}}$ (according to the needs of the requester), and for some $q \geq \bar{q}$, $U_{\texttt{pbp}}(q) > U_{\texttt{pbp}}(q_{\texttt{std}})$, or equivalently

$$c(q) - c(q_{\texttt{std}}) < p(\Pr(\texttt{base}|q) - \Pr(\texttt{base}|q_{\texttt{std}})) + b. \tag{5.3}$$

Again we see that for PBPs to help it must be possible for the worker to increase his quality with additional effort at a reasonable cost, and the bonus must be set sufficiently large.

Equation 5.3 provides a concrete and simple way to think about whether it would help to increase the bonus threshold $\bar{q}$. For fixed payments p and b, it is optimal to choose as $\bar{q}$ the largest q which satisfies Equation 5.3. However, $\bar{q}$ is a *perceived* threshold and cannot always be controlled directly. Instead, it may help to alter workers' perception, perhaps without even changing the objective bonus rule. For example, in the proofreading experiment from Section 5.3.2 the "5 typos" bonus rule is roughly equivalent to the "25%" bonus rule in terms of when bonus payments are awarded, yet workers react differently. One possible explanation for this general phenomenon is a difference in the perceived value $\bar{q}$.

Comparison with Principal-Agent Model

In the standard principal-agent model, a worker maximizes his payment minus the intrinsic cost of his effort. We deviate from the standard model in that we include the worker's subjective beliefs about how high quality his work must be to be paid. In particular, while the objective probability of receiving the base payment is often 1, the subjective probability may be much smaller, depending on the quality level. This feature allows us to capture some effects that are not captured by the standard model, e.g., that increasing the base payment may increase the quality of work that the worker chooses to produce, while removing uncertainty about the base payment may decrease quality.

Our model may be useful as a more realistic foundation for theoretical work. As an example, consider Ho et al. [62], a recent theoretical paper on the optimization of PBPs. While that paper posits the standard principal-agent model, all results carry over to our model.

5.5 Summary

We describe the results of a series of experiments studying the effect of performance-based payments on the quality of crowdwork on Amazon Mechanical Turk. Our goal is to identify properties of the payment, payment structure, and task that allow for quality improvements using PBPs.

We find that PBPs can improve the quality of submitted work for some tasks, but are not likely to for others. We identify the extent to which a task is *effort-responsive* as a potential important reason for whether or not PBPs work for this task. This leads to an actionable insight for requesters. When considering whether or not to use PBPs, one could first run a pilot experiment to determine whether a task is effort-responsive by examining the correlation between time spent and quality. If additional time spent leads to a sufficiently high boost in quality, we would expect PBPs to improve performance.

We find strong evidence for what we call the *implicit PBP effect*: workers may have their own subjective beliefs about the quality of work they must submit to have their work accepted, which makes them view fixed payments as implicitly performance-based. Workers may also have subjective beliefs about the likelihood of receiving the bonus when payments are explicitly performance-based. This should be taken into account when designing a payment scheme. For example, we find that in some cases a requester can incentivize higher quality work by defining the bonus threshold relative to other workers or with respect to gold standard data that the workers do not have access to.

We show that in order for PBPs to improve quality, the bonus payment offered must be sufficiently large, but that there are diminishing returns for further increasing this payment. This partially explains existing negative results on the effectiveness of PBPs in crowdsourcing markets. We provide evidence that when PBPs improve quality, they do so for a wide range of quality thresholds.

Finally, we suggest a theoretical model of workers' behavior that captures all of the above effects and explains several outcomes we observed in our experiments. We believe this model may be useful in further work on crowdsourcing markets, both as a concrete way to think about the consequences of using this or that payment scheme in practice, and as a more realistic foundation for theoretical work compared to the standard principal-agent model.

CHAPTER 6

Interactions Between Learning and Incentives

6.1 Overview

In crowdsourcing markets, the requester often tries to procure information from workers, who might have their own objectives and need to be incentivized to work. So far in this dissertation, we have separately considered learning problems and worker incentives. However, these two aspects often coincide and need to be considered jointly.

In this chapter, we consider the following question which involves the interaction of learning and incentives.

> In a world where data is offered to us at heterogeneous prices by self-interested agents, and in particular when these prices may be arbitrarily correlated with the underlying data, how can we design mechanisms that are incentive-compatible, have robust learning guarantees, and optimize the cost-efficiency of learning?

We will address this question for the classic problem of statistical learning which we now describe.

From sample complexity to budget efficiency

The classical problem in statistical learning theory is the following. We are given n datapoints (examples) $z_1, \ldots, z_n \in \mathbb{Z}$ sampled from some distribution $\mathcal{D}$. Our goal is to select a hypothesis $h \in \mathcal{H}$ which "performs well" on unseen data from $\mathcal{D}$. We can specify performance in terms of a *loss function* $\ell(h, z)$, and we write $\mathcal{L}(h)$, known as the *risk* of h, to be the expectation of $\ell(h, z)$ on a random draw z from $\mathcal{D}$. The goal is to produce a hypothesis $\bar{h}$ whose risk is not much more than

that of h^*, the optimal member of $\mathcal{H}$. For example, in *binary classification*, each data point consists of a pair $z = (x, y)$ where x encodes some "features" and $y \in \{-1, 1\}$ is the label; a hypothesis h is a function that predicts a label for a given set of features; and the loss $\ell(h, (x, y))$ is 0 if $h(x) = y$ and is 1 otherwise.

Research in statistical learning theory attempts to characterize how well such tasks can be performed in terms of the *resources* available and the inherent *difficulty* of the problem. The resource is usually the quantity of data n. In binary classification, for instance, the difficulty or richness of the problem is captured by the "VC-dimension" d, and a famous result is that there is an algorithm achieving the bound

$$\mathcal{L}(\bar{h}) \leq \mathcal{L}(h^*) + O\left(\sqrt{\frac{d \log n}{n}}\right). \tag{6.1}$$

In this chapter, we consider an alternative scenario: the learner has a fixed budget B and can use this budget to purchase examples. More precisely, on round t of a sequence of T rounds, agent t arrives with data point z_t, sampled i.i.d. from some $\mathcal{D}$, and a cost $c_t \in [0, 1]$. The learning mechanism may offer a menu of *take-it-or-leave-it* prices $\pi_t(z)$, with a possibly different price $\pi_t(z)$ for each data point z. The arriving agent observes the price $\pi_t(z_t)$ offered for her data and accepts as long as $\pi_t(z_t) \geq c_t$, in which case the mechanism pays the agent $\pi_t(z_t)$ and learns (c_t, z_t).[1] Our goal is to actively select data to purchase, subject to a budget B, in order to minimize the risk of our final output $\bar{h}$. At a high level, our main result is as follows:

Main Result 1 (Informal). For a large class of problems, there is an active data purchasing algorithm that spends at most B in expectation and outputs a hypothesis $\bar{h}$ satisfying,

$$\mathbb{E}\mathcal{L}(\bar{h}) \;\leq\; \mathcal{L}(h^*) + O\left(\sqrt{\frac{\gamma}{B}}\right),$$

where $\gamma \in [0, 1]$ is a parameter of the (cost, data) sequence capturing the *monetary difficulty of learning* and the expectation is over the algorithm's internal randomness. The only prior knowledge required is a rough estimate of γ.[2]

[1] We will discuss the interaction model further in Sections 6.2.

[2] Knowledge of γ can be weakened with gracefully degrading guarantees; for instance, a rough estimate of the mean of the arriving costs suffices.

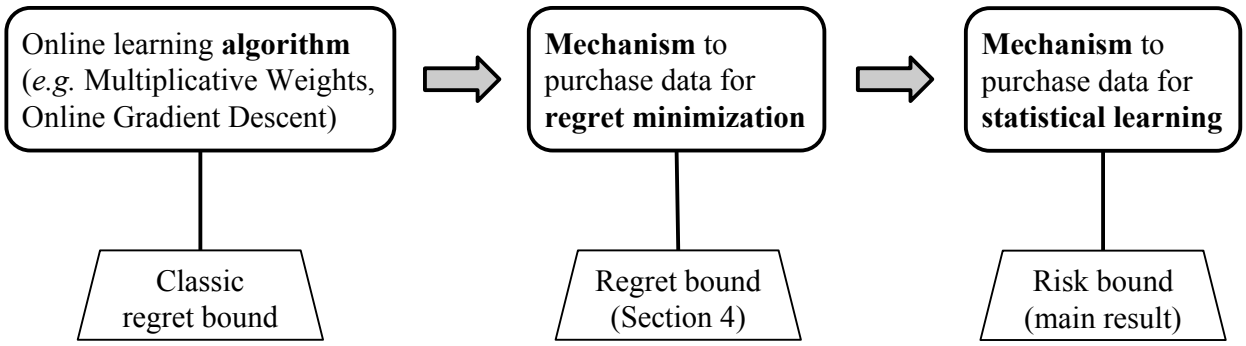


Figure 6.1: **Algorithmic and analytic approach.** First, we convert "Follow-the-Regularized-Leader" online no-regret algorithms (a broad class) into mechanisms that purchase data for a regret-minimization setting that we introduce for purposes of analysis. Then, we convert these into mechanisms to solve our main problem, statistical learning. The mechanisms interact with the online learning algorithms as black boxes, but the analysis relies on "opening the box".

Our results highlight parallels to classical bounds such as Equation 6.1. The key resource constraint of the problem is now the *budget* B.[3] We also identify a parameter $\gamma \in [0, 1]$ that captures the *monetary difficulty* of the setting. γ is roughly defined as *the average product of an arriving datapoint's cost and its "difficulty score"*. An important illustration of the role of γ is the following. If the average cost of the data goes to 0, then $\gamma \to 0$ and Main Result 1 guarantees good learning for cheap. The same holds if the average "difficulty score" of the data goes to 0.

But in some settings, the average cost of data is large and the average "difficulty score" is also large. The purchasing problem appears difficult, requiring a large budget; and the learning-theoretic problem also appears difficult relative to the traditional resource of quantity of data. However, due to beneficial correlations between difficulty and cost, it may be that the parameter γ remains relatively small. In this case, Main Result 1 implies that we can learn at a much more efficient rate relative to the resource we care about, the budget. And crucially, this fact would not have been apparent by considering either the economic or the learning-theoretic aspects of the problem in isolation.

Overview of Techniques

Our general idea for attacking this problem is to utilize online learning algorithms (OLAs) for regret minimization [24]. These algorithms output a hypothesis or prediction at each step $t = 1, \ldots, T$, and their performance is measured by the summed loss of these predictions over all the steps. The idea is that the hypotheses produced by the OLA at each step can be used both to determine the value of data during the procurement process and to generate a final prediction.

In Section 6.3.1, we lay out the tools we need for a pricing and learning mechanism to interact with OLAs. The first high-level problem is that, because of the budget constraint, our OLA will only see a small subset of the data sequence. We use the tool of *importance-weighting* to give good regret-minimization guarantees even when we do not see the entire data sequence. The second problem is how to aggregate the hypotheses of the OLA and convert its regret guarantee into a risk guarantee for our statistical learning setting. This is achieved with the standard "online-to-batch" conversion [25].

Given the tools of Section 6.3.1, the key remaining challenge is to develop a pricing and learning strategy that achieves low regret. We address this question in Section 6.3.2. We formally define a model of online learning for regret minimization with purchased data, in which the mechanism must output a hypothesis at each time step and perform well in hindsight against the entire data sequence, but only has enough budget to purchase and observe a fraction of the arriving data. We defer until later our detailed analysis of this setting, derivation of a pricing strategy, and lower bounds. At this point, we present our pricing strategy and prove regret guarantees for this setting.

In Section 6.4, we prove our main results: risk guarantees for a learner with budget B and access to T arriving agents. These bounds follow directly by using the tools in Section 6.3.1 and regret-minimization results in Section 6.3.2.

In Section 6.5, we develop a deeper understanding of the regret minimization setting. We derive our pricing strategy from an in-depth analysis of a more analytically tractable variant of the problem, the "at-cost" setting, where the mechanism is only required to pay the cost of the arriving data

[3] Assuming that budget is scarce relative to data; see formal theorems for details.

point rather than the price posted. For this setting, we are able to derive the optimal pricing strategy for minimizing the regret bound of our class of learning algorithms subject to an expected budget constraint.

We also complement our upper bounds by proving lower bounds for data-purchasing regret minimization. These show that our mechanisms for the easier at-cost setting have an order-optimal regret guarantee of $\frac{T}{\sqrt{B}}\gamma$. There is a small gap to our mechanisms for the main regret minimization setting, in which our guarantee is on the order of $\frac{T}{\sqrt{B}}\sqrt{\gamma}$ (recall that $\gamma \in [0, 1]$, so this is a weaker guarantee). The dependence $T/\sqrt{B}$ approaches the classic $\sqrt{T}$ regret bound when B is large (approaching T). When B is small but still superconstant, we observe the perhaps counterintuitive fact that we can achieve $o(1)$ average regret per arrival while only observing an $o(1)$ fraction of the arriving data; in other words, we have "no data, no regret."

All proofs appear in the appendix.

6.1.1 Related work

Our motivations overlap with much prior work, but the problem of generic learning with purchased data does not seem to admit active solutions from the literature. A naïve approach would be to offer the maximum price of 1 to every arriving agent until the budget is exhausted, then run the learning algorithm. To improve on this is already highly nontrivial, because as soon as one posts a price below 1, the obtained data becomes biased toward lower costs, making it unclear how to guarantee good learning properties on the entire distribution.

Improvements on this naïve solution for "batch" settings have appeared in recent work, especially Roth and Schoenebeck [98], which considered the design of mechanisms for efficient estimation of a statistic with incentive-compatible payment schemes. However, this work and others in related settings [30, 51, 84] consider "passive" solutions that do not treat different types of data points differently, *e.g.* drawing a posted price independently regardless of the type of datapoint. We focus on an *active* approach in which the marginal value of individual examples is estimated according to the current learning progress and budget.

Horel et al. [64] does take a data-dependent approach to purchasing data for learning, but focuses

on a quite different learning setting, focusing on a model of regression with noisy samples and using a budget-feasible mechanism design approach.

Other works such as Dekel et al. [33], Ghosh et al. [53], Meir et al. [91] focus on a setting in which agents may misreport their data (see also the peer-prediction literature). We suppose that agents may misreport their costs but not their data.

Many of the ideas in the present work draw from recent advances in using importance weighting for the active learning problem [15]. There is a wealth of theoretical research into active learning, including Balcan et al. [11], Beygelzimer et al. [14], Hanneke [56] and many others.

"Budgeted Learning" is a somewhat related area of machine learning, but there the budget is not *monetary*. The idea is that we do not see all of the features of the data points in our set, but rather have a "budget" (for instance, we may choose to observe any two of the three features height, weight, age).

6.2 Problem Formulation: Actively Purchasing Data for Learning

In this section, we formally define the problem setting. The body of the paper will then consist of a series of steps for deriving mechanisms for this setting with provable guarantees, which will finally appear in Section 6.4.

We consider a statistical learning problem described as follows. Our data points are objects $z \in \mathbb{Z}$. We are given a hypothesis class $\mathcal{H}$ which we will assume is parameterized by vectors $\mathbb{R}^d$ but more broadly can be any Hilbert space endowed with a norm $\|\cdot\|$; for convenience we will treat elements $h \in \mathcal{H}$ as vectors which can be added, scaled, etc. We are also given a loss function $\ell : \mathcal{H} \times \mathbb{Z} \to \mathbb{R}$ that is convex in $\mathcal{H}$. We assume throughout the paper that the loss function is *1-Lipschitz* in h; that is, for any $z \in \mathbb{Z}$ and any $h, h' \in \mathcal{H}$ we have $|\ell(h, z) - \ell(h', z)| \leq \|h - h'\|$.

In many common scenarios, $\mathbb{Z}$ is the space of pairs (x, y) from the cross product $\mathcal{X} \times \mathcal{Y}$, with x the feature input and y the label, though in our setting $\mathbb{Z}$ can be a more generic object. For example, in the canonical problem of *linear regression*, we have that $\mathbb{Z} = \mathcal{X} \times \mathcal{Y} = \mathbb{R}^d \times \mathbb{R}$, the hypothesis class is vectors $\mathcal{H} = \mathbb{R}^d$, and the loss function is defined according to squared error

$\ell(h, (x, y)) := (h^\top x - y)^2$.

The **data-purchasing statistical learning problem** is parameterized by the data space $\mathbb{Z}$, hypothesis space $\mathcal{H}$, loss function ℓ, number of arriving data points T, and expected budget constraint B. A *problem instance* consists of a distribution $\mathcal{D}$ on the set $\mathbb{Z}$ and a sequence of pairs $(c_1, z_1), \ldots, (c_T, z_T)$ where each z_t is a data point drawn i.i.d. according to $\mathcal{D}$ and each $c_t \in [0, 1]$ is the cost associated with that data point. The costs may be arbitrarily chosen, *i.e.* we consider a worst-case model of costs. (For instance, if costs and data are drawn together from a joint, correlated distribution, then this is a special case of our setting.)

In this problem, the task is to design a *mechanism* implementing the operations "post", "receive", and "predict" and interacting with the problem instance as follows.

- For each time step $t = 1, \ldots, T$:
    1. The mechanism *posts* a pricing function $\pi_t : \mathbb{Z} \to \mathbb{R}$, where $\pi_t(z)$ is the price posted for data point z.
    2. Agent t arrives, possessing (c_t, z_t).
    3. If the posted price $\pi_t(z_t) \geq c_t$, then agent t accepts the transaction: The mechanism pays $\pi_t(z_t)$ to the agent and *receives* (c_t, z_t). If $\pi_t(z_t) < c_t$, agent t rejects the transaction and the mechanism *receives* a null signal.
- The mechanism outputs a *prediction* $\bar{h} \in \mathcal{H}$.

The design problem of the mechanism is how to choose the pricing function π_t to *post* at each time, how to update based on *receiving* data, and how to choose the final *prediction*. The *risk* or predictive error of a hypothesis is

$$\mathcal{L}(h) = \mathbb{E}_{z \sim \mathcal{D}} \ell(h, z)$$

and the goal of the mechanism is to minimize the risk $\mathcal{L}(\bar{h})$ of its final hypothesis $\bar{h}$. The benchmark is the optimal hypotehsis in the class, $h^* = \arg\min_{h \in \mathcal{H}} \mathcal{L}(h)$.

The mechanism must guarantee that, for every input sequence $(c_1, z_1), \ldots, (c_T, z_T)$, it spends at most B in expectation over its own internal randomness.

Agent-mechanism interaction. The model of agent arrival and posted prices contains several assumptions. First, agents cannot fabricate data; they can only report data they actually have to the mechanism. Second, agents are rational in that they accept a posted price when it is higher than their cost and reject otherwise. Third, the mechanism can truthfully obtain the agent's costs c_t.

We emphasize that the purpose of this paper is not focused on the implementation of such a setting, but instead on developing active learning and pricing techniques and guarantees. This is also intended as a simple and clean model in which to begin developing such techniques. However, we briefly note some possible implementations.

In the most straightforward one, the mechanism posts prices directly to the agent who responds directly. This would be a weakly truthful implementation, as agents have no incentive to misreport costs after they choose to accept the transaction.

One strictly truthful implementation uses a *trusted third party* (TTP) that can facilitate the transactions (and guarantee the validity of the data if necessary). For example, we could imagine attempting to learn to classify a disease, and we could rely on a hospital to act as the broker allowing us to negotiate with patients for their data. Then the TTP/agent interaction could proceed as follows:

1. Learning mechanism submits the pricing function π_t to the TTP;
2. Agent provides his data point z_t and cost c_t to the TTP;
3. TTP determines whether $\pi_t(z_t) \geq c_t$ and, if so, instructs the learner to pay $\pi_t(z_t)$ to the agent and then provides the pair (z_t, c_t) to the learner.

Other possibilities for strictly truthful implementation include using a bit of cryptography.

6.3 Our Approach

6.3.1 Tools for Converting Regret-Minimizing Algorithms

In this section, we begin with the classic regret-minimization problem and a broad class of algorithms for this problem. We then show how to apply techniques that convert these algorithms into a form

that will be useful for solving the statistical learning problem with purchased data. The only missing ingredient will then be a price-posting strategy, which will be presented in Section 6.3.2.

Recap of classic regret-minimization setting and algorithms

In the classic regret-minimization problem, we have a hypothesis class $\mathcal{H}$ with the same assumptions as stated in Section 6.2. At each time $t = 1, \ldots, T$, the algorithm posts a hypothesis $h_t \in \mathcal{H}$. A convex loss function $f_t : \mathcal{H} \to \mathbb{R}$ arrives.[4] We assume the loss functions are 1-Lipschitz. The algorithm receives f_t, updates, and posts a hypothesis h_{t+1}.

The *loss* of the algorithm on this particular input sequence is

$$Loss = \sum_{t=1}^{T} f_t(h_t). \tag{6.2}$$

We define the *regret* as compared to the best fixed hypothesis h^* in hindsight:

$$Regret = Loss - \min_{h^* \in \mathcal{H}} \sum_{t=1}^{T} f_t(h^*). \tag{6.3}$$

We generally consider *expected* loss and regret, where the expectation is over any randomness in the algorithm (but is still for a fixed, non-random input data sequence). An algorithm is said to guarantee regret $R(T)$ if for every sequence of loss functions $f_1, \ldots, f_T$, its expected regret is at most $R(T)$.

We utilize the broad class of *Follow-the-Regularized-Leader* (FTRL) online algorithms (Algorithm 4). This class includes most popular no-regret algorithms such as Online Gradient Descent, Multiplicative Weights, and so on for particular choices of the regularizer G (these special cases usually have very computationally efficient updates). For example, Multiplicative Weights follows by using the negative entropy function as a regularizer, which is strongly-convex with respect to ℓ_1 norm [24]. Online Gradient Descent follows by using the regularizer $G(h) = \frac{1}{2}\|h\|_2^2$, which is strongly-convex with respect to ℓ_2 norm.

It is well-known (and indeed follows as a special case of Lemma 6.3.1) that, under the assumptions on our setting, FTRL algorithms guarantee an expected regret bound of $O(\sqrt{T})$, and this is

[4] This definition of "loss function" is a departure from our main setting which involved $\ell(\cdot, \cdot)$. But we will use this somewhat more general setup by choosing $f_t(h) \propto \ell(h, z_t)$ for the datapoint z_t.

ALGORITHM 4: Follow-the-Regularized-Leader (FTRL). Note we assume in the definition a strongly-convex regularizer G.

Input: parameter η,

regularizer $G : \mathcal{H} \to \mathbb{R}$ which is strongly-convex with respect to a norm $\|\cdot\|$ over $\mathcal{H}$

for $t = 1, \ldots, T$ **do**

post hypothesis h_t;

receive loss function f_t;

update $h_{t+1} = \inf_{h \in \mathcal{H}} \left\{ \sum_{t' \le t} f_{t'}(h) \; + \; \eta G(h) \right\}$;

tight in that no algorithm has a better guarantee. We will later prove generalizations of these bounds for a regret minimization setting with purchased data.

Importance-Weighting Technique for Less Data

As a starting point, suppose we wish to design an online learning algorithm that does not observe all of the arriving loss functions, but still performs well against the entire arrival sequence.

Because the arrival sequence may be adversarially chosen, a good algorithm should randomly choose to sample some of the arrivals. In this section, we abstract away the decision of how to randomly sample. (This will be the focus of Section 6.3.2.) In this section, we suppose that at each time t, after posting a hypothesis h_t, a probability $q_t > 0$ is specified by some external means as a (possibly random) function of the preceding time steps. With probability q_t independently, we observe f_t; with probability $1 - q_t$, we do not observe f_t.

Our goal is to modify our online learning algorithms for this setting and obtain a modified regret guarantee. Notice crucially that the definition of loss (6.2) and regret (6.3) are unchanged: We still suffer the loss $f_t(h_t)$ regardless of whether we observe f_t.

The key technique we use is *importance weighting*. The idea is that, if we only observe each of a sequence of values x_i with probability p_i, then we can get an unbiased estimate of their sum by taking the sum of $\frac{x_i}{p_i}$ for those we do observe. To check this fact, let $\mathbb{1}_i$ be the indicator variable

for the event that we observe i and note that the expectation of our sum is $\mathbb{E}\left[\sum_i \mathbb{1}_i \frac{x_i}{p_i}\right] = \sum_i x_i$. This is called importance-weighting the observations (and is a specific instance of a more general machine learning technique). Furthermore, if each $\frac{x_i}{p_i}$ is bounded and independent, we can expect the estimate to be quite good via tail bounds.

The importance-weighted modification to an online learning algorithm is outlined in Algorithm 5. The importance-weighted regret guarantee we obtain is given in Lemma 6.3.1. It depends upon a key parameter at each time step, for a given hypothesis h of the online algorithm and arriving loss function f:

Definition 1. For a norm $\|\cdot\|_\star$, a hypothesis $h \in H$, and a convex loss function $f : \mathcal{H} \to \mathbb{R}$, define

$$\Delta_{h,f} = \|\nabla f(h)\|_\star.$$

We will informally refer to $\Delta_{h,f}$ as the "benefit" or "value" of f when the current hypothesis is h.

In this paper, we will always use a FTRL algorithm, which is assumed to have a regularizer G that is strongly-convex according to some norm $\|\cdot\|$. In this case, $\|\cdot\|_\star$ is the dual norm to $\|\cdot\|$. We assume implicitly from now on that this $\|\cdot\|_\star$ is the norm used in the definition of $\Delta_{h,f}$.

ALGORITHM 5: Importance-Weighted Online Learning Algorithm.

Input: access to Online Learning Algorithm (OLA)

for $t = 1, \ldots, T$ **do**

- post hypothesis $h_t \leftarrow$ OLA;
- observe q_t (chosen by some external process);
- **if** *receive loss function f_t (which occurs with probability q_t)* **then**
  - let *importance-weighted loss function* $\hat{f}_t(\cdot) = \frac{f_t(\cdot)}{q_t}$;
  - send $\hat{f}_t \to$ OLA;
- **else**
  - send 0 function $\to$ OLA;

Lemma 6.3.1. *If Algorithm 5 is run with (nonzero) probabilities $q_1, \ldots, q_T$ and FTRL as the*

learning algorithm, then the expected regret is bounded by

$$\frac{\beta}{\eta} + 2\eta\mathbb{E}\left[\sum_{t=1}^{T}\frac{\Delta_{h_t,f_t}^2}{q_t}\right],$$

where β is a constant depending on $\mathcal{H}$, η is a parameter of the algorithm, and the expectation is over any randomness in the choices of h_t and q_t.

We can recover the classic regret bound as follows: Take each $q_t = 1$, and note by the Lipschitz assumption that each $\Delta_{h_t,f_t} \leq 1$. Then by setting $\eta = \Theta\left(1/\sqrt{T}\right)$, we get an expected regret bounded by $O\left(\sqrt{T}\right)$.

The "Online-to-Batch" Conversion

So far so good: We can convert an online regret-minimization algorithm to use smaller amounts of data (leaving the question of how and when to purchase that data till Section 6.3.2).

The next question is: Given that we have such an algorithm, how can we use it to solve our original task of generating an accurate *prediction* based on the data?

We answer that question with a standard tool known as the "online-to-batch conversion" (so-named for converting an online learning algorithm into a "batch" or statistical learner). The standard argument (*e.g.* [101]) goes as follows: Given a "batch" of i.i.d. data points, feed them to the no-regret algorithm. Because the algorithm has low regret, on average over the data points, its hypotheses predict well. But since each data point was drawn i.i.d., this means that the hypotheses on average predict well on an i.i.d. draw from the distribution. Thus it suffices to take the mean of the hypotheses or to pick one of them uniformly at random.

Lemma 6.3.2 (Online-to-Batch [25])**.** *Suppose the sequence of convex loss functions $f_1, \ldots, f_T$ are drawn i.i.d. from a distribution $\mathcal{F}$ and that an online learning algorithm with hypotheses $h_1, \ldots, h_T$ achieves expected regret $R(T)$. Let $\mathcal{L}(h) = \mathbb{E}_{f\sim\mathcal{F}} f(h)$ and $h^* = \arg\min_{h\in\mathcal{H}} \mathcal{L}(h)$. Then by choosing either $\bar{h} = \frac{1}{T}\sum_{t=1}^{T} h_t$ or $\bar{h}$ a uniformly randomly selected h_t, we have*

$$\mathbb{E}_{\substack{f_1,\ldots,f_T,\\ alg}}\mathcal{L}(\bar{h}) \leq \mathcal{L}(h^*) + \frac{1}{T}R(T).$$

We note that this conversion will continue to hold in the data-purchasing no-regret setting we define next, since all that is required is that the algorithm output a hypothesis h_t at each step and that there is a regret bound on these hypotheses.

6.3.2 Regret Minimization with Purchased Data

In this setting, we define the problem of regret minimization with purchased data. We will design mechanisms with good regret guarantees for this problem, which will translate via the aforementioned online-to-batch conversion (Lemma 6.3.2) into guarantees for our original problem of statistical prediction.

The essence of the data-purchasing no-regret learning setting is that an online algorithm ("mechanism") is asked to perform well against a sequence of data, but by default, the mechanism does not have the ability to see the data. Rather, the mechanism may purchase the right to observe data points using a limited budget. The mechanism is still expected to have low regret compared to the optimal hypothesis in hindsight on the entire data sequence (even though it only observes a portion of the sequence).

Problem Definition

The **data-purchasing regret minimization problem** is parameterized by the hypothesis space $\mathcal{H}$, number of arriving data points T, and expected budget constraint B. A *problem instance* is a sequence of pairs $(c_1, f_1), \ldots, (c_T, f_T)$ where each $f_t : \mathcal{H} \to \mathbb{R}$ is a convex loss function and each $c_t \in [0, 1]$ is the cost associated with that data point. We assume that the f_t are 1-Lipschitz, and suppose that the f_t come from a set $\mathcal{F}$.

In this problem, we design a *mechanism* implementing the operations "post" and "receive" and interacting with the problem instance as follows.

- For each time step $t = 1, \ldots, T$:

    1. The mechanism *posts* a hypothesis h_t and a pricing function $\pi_t : \mathcal{F} \to \mathbb{R}$, where $\pi_t(f)$ is the price posted for loss function f.

2. Agent t arrives, possessing (c_t, f_t).

3. If the posted price $\pi_t(f_t) \geq c_t$, then agent t accepts the transaction: The mechanism pays $\pi_t(f_t)$ to the agent and *receives* (c_t, f_t). If $\pi_t(f_t) < c_t$, agent t rejects the transaction and the mechanism *receives* a null signal.

Note the key differences from the statistical learning setting: We must post a hypothesis h_t at each time step (and we do not output a final prediction), and data is not assumed to come from a distribution.

The goal of the mechanism is to minimize the loss (Equation 6.2), namely $\sum_t f_t(h_t)$. The definition of regret is also the same as in the classical setting (Equation 6.3). Note that we suffer a loss $f_t(h_t)$ at time t regardless of whether we purchase f_t or not. The mechanism must also guarantee that, for every problem instance $(c_1, f_1), \ldots, (c_T, f_T)$, it spends at most B in expectation over its own internal randomness.

The Importance-Weighting Framework

Recall that, in Section 6.3.1, we introduced the *importance-weighting* technique for online learning. This gave regret guarantees for a learning algorithm when each arrival f_t is observed with some probability q_t.

Our general approach will be to develop a strategy for randomly drawing posted prices π_t. This will induce a probability q_t of obtaining each arrival f_t.

Therefore, the entire problem has been reduced to choosing a posted-price strategy at each time step. This posted-price strategy should attempt to minimize the regret bound while satisfying the expected budget constraint.

A brief sketch of the proof arguments is as follows. After we choose a posted price strategy, each q_t will be determined as a function of h_t, c_t, and f_t. (q_t is just equal to the probability that our randomly drawn price exceeds the agent's cost c_t.) Thus, we can apply Lemma 6.3.1, which (recall) stated that for these induced probabilities q_t, the expected regret of the learning algorithm is

$$\frac{\beta}{\eta} + 2\eta\mathbb{E}\sum_t \frac{\Delta^2_{h_t,f_t}}{q_t},$$

where β is a constant and η is a parameter of the learning algorithm to be chosen later.

After we choose and apply such a strategy, the general approach to proving our regret bounds is to find an *a priori* bound M such that $2\mathbb{E}\sum_t \frac{\Delta^2_{h_t,f_t}}{q_t} \leq M$. Then the regret bound becomes $\frac{\beta}{\eta} + \eta M$. If we know this upper-bound M in advance using some prior knowledge, then we can choose $\eta = \Theta\left(1/\sqrt{M}\right)$ as the parameter for our learning algorithms. This gives a regret guarantee of $O(\sqrt{M})$.

A First Step to Pricing: The "At-Cost" Variant

The bulk of our analysis of the no-regret data-purchasing problem actually focuses on a slightly easier variant of the setting: If the arriving agent accepts the transaction, then the mechanism only has to pay the cost c_t rather than the posted price $\pi_t(f_t)$. We call this the "at-cost" variant of the problem. This setting turns out to be much more analytically tractable: We derive optimal regret bounds for our mechanisms and matching lower bounds. We then take the key approach and insights derived from this variant and apply them to produce a solution to the main no-regret data purchasing problem. In order to keep the story moving forward, we summarize our results for the "at-cost" setting here and explore how they are obtained in Section 6.5.

In the at-cost setting, we are able to solve directly for the pricing strategy that minimizes the importance-weighted regret bound of Lemma 6.3.1. We first state the strategy and result in Theorem 6.3.3, then explain and discuss.

Theorem 6.3.3. *There is a mechanism for the "at-cost" problem of data purchasing for regret minimization that interfaces with FTRL and guarantees to meet the expected budget constraint, where for a parameter $\gamma \in [0, 1]$ (Definition 2),*

1. *The expected regret is bounded by $O\left(\max\left\{\frac{T}{\sqrt{B}}\gamma\ ,\ \sqrt{T}\right\}\right)$.*
2. *This is optimal in that no mechanism can have a better guarantee (up to constant factors).*
3. *The pricing strategy is to set $K = O\left(\frac{T}{B}\gamma\right)$ and draw $\pi_t(f)$ randomly according to a distribution such that*

$$\Pr[\pi_t(f) \geq c] = \min\left\{1\ ,\ \frac{\Delta_{h_t,f}}{K\sqrt{c}}\right\}.$$

The only prior knowledge required is an estimate of γ up to a constant factor.

To understand this bound, we will answer the following questions about the parameter γ:

1. How is γ formally defined?
2. When should we expect our mechanism to have approximate prior knowledge of γ?
3. What guarantees can we obtain if we do not have this knowledge?
4. What are the implications of γ's dependence on the choice of learning algorithm?

(1) How is γ defined?

Definition 2. For a fixed input sequence $(c_1, f_1), \ldots, (c_T, f_T)$, $\Delta_{h,f}$ in Definition 1, and a mechanism outputting (possibly random) hypotheses $h_1, \ldots, h_T$, define

$$\gamma = \mathbb{E}\frac{1}{T}\sum_t \Delta_{h_t,f_t}\sqrt{c_t}$$

where the expectation is over the randomness of the algorithm. Note that γ lies in $[0, 1]$ by our assumptions on bounded cost and Lipschitz loss.

In each term in the γ summation, the value Δ_{h_t,f_t} intuitively captures both the "difficulty" of the data f_t and also the "value" or "benefit" of f_t. To explain the difficulty aspect, by examining the regret bound for FTRL learning algorithms (*e.g.* the importance-weighted regret bound of Lemma 6.3.1 with all $q_t = 1$), one observes that if each Δ_{h_t,f_t} is small, then we have an excellent regret bound for our learning algorithm; the problem is "easy". To explain the "value aspect", one can for concreteness take the Online Gradient Descent algorithm; the larger the gradient, the larger the update at this step, and Δ_{h_t,f_t} is the norm of the gradient.

Thus, γ captures the correlations between the *value* or *benefit* of the arriving data and the cost of that data. If either the mean of the costs or the average benefit Δ_{h_t,f_t} of the data is converging to 0, then $\gamma \to 0$ and in these cases we can learn with high accuracy very cheaply, as may be expected. More interestingly, it is possible to have both high average costs, and high average data-values, and yet still have $\gamma \to 0$ due to beneficial correlations. In these cases we can learn much more cheaply than might be expected based on either the economic side or the learning side alone.

(2) When should we expect to have good prior knowledge of γ? Although in general γ will be domain-specific, there are several reasons for optimism. First, γ compresses all information about the data and costs into a single scalar parameter (compare to the common mechanism-design assumption that the prior distribution of agents' values is fully known). Second, we do not need very exact estimates of γ (*e.g.* we do not need to know $\gamma \pm \epsilon$): For order-optimal regret bounds, we only need an estimate within a constant factor of γ. Third, γ is directly proportional to K, which is a normalization constant: If we increase K, the probability of obtaining a given data point only decreases, and vice versa. In fact, the best choice of K is the normalization constant so that, if we draw prices according to the proposed distribution of Theorem 6.3.3, we run out of budget precisely when the last arrival leaves. Thus, K (equivalently, γ) can be estimated and adjusted online by tracking the "burn rate" (spending per unit time) of the algorithm. In simulations, we have observed success with a simple approach of estimating K based on the average correlation so far along with the burn rate, *i.e.* if the current estimated γ is $\hat{\gamma}$ and there are $\hat{T}$ steps remaining with $\hat{B}$ budget remaining to spend, set $K = \hat{\gamma}\hat{T}/\hat{B}$.

(3) What can we prove without prior knowledge of γ? It turns out that, if we only have an estimate of $\bar{c} = \frac{1}{T}\sum_t \sqrt{c_t}$, respectively $\mu = \frac{1}{T}\sum_t c_t$, then this suffices for regret guarantees on the order of $T\bar{c}/\sqrt{B}$, respectively $T\sqrt{\mu}/\sqrt{B}$. This "graceful degradation" will continue to be true in the main setting. The idea is that we can follow the optimal form of the pricing strategy while choosing any normalization constant $K \geq \frac{T}{B}\gamma$. It may no longer be optimal, but it will ensure that we satisfy the budget and give guarantees depending on the magnitude of K. So all we need is an approximate estimate of some value larger than γ. Both $\bar{c}$ and μ are guaranteed to upper-bound on γ, so both can be used to pick K while satisfying the budget.

(4) γ depends on the algorithm—what are the implications? If the original OLA chosen for the learning problem was a poor fit or has bad guarantees, then after being converted with our reduction, it is probably still a poor fit and γ may not be an appropriate measure of difficulty of the problem in general. However, the positive result is that if one chooses a good algorithm for the problem, the dependence on γ shows that the mechanism can take advantage of correlations when they exist. If we do have a good algorithm, a natural question is:

(4a) Can we remove the algorithm-dependence of γ? One might hope to achieve a bound depending on an algorithm-independent quantity that captures correlations between data and cost. A natural candidate is $\gamma^* := \frac{1}{T}\sum_t \|\nabla\ell(h^*, z_t)\|\sqrt{c_t}$. In general, there are difficult cases where one can not achieve a bound in terms of γ^*. However, in nicer scenarios we may expect γ to approximate γ^*. For instance, suppose $\ell(h, z) = \phi(h^\top z)$ where ϕ is a differentiable 1-Lipschitz convex function (commonly-used examples include the *squared hinge loss* and the *log loss*). Under this condition, we can show that

$$\begin{aligned}
\|\nabla\ell(h_t, z_t)\|\sqrt{c_t} - \|\nabla\ell(h^*, z_t)\|\sqrt{c_t} &\leq \|(\phi'(h_t^\top z_t) - \phi'(h^{*\top} z_t))z_t\| \\
&\leq |\phi(h_t^\top z_t) - \phi(h^{*\top} z_t)| \\
&= |\ell(h_t, z_t) - \ell(h^*, z_t)|.
\end{aligned}$$

By the regret guarantee of our mechanism when run with a good algorithm, even initialized with very weak knowledge, this difference in losses per time step is $o(1)$, implying that $\gamma \to \gamma^*$. A deeper investigation of this phenomenon is a good candidate for future work.

Mechanisms and Results for Data-Purchasing Regret Minimization

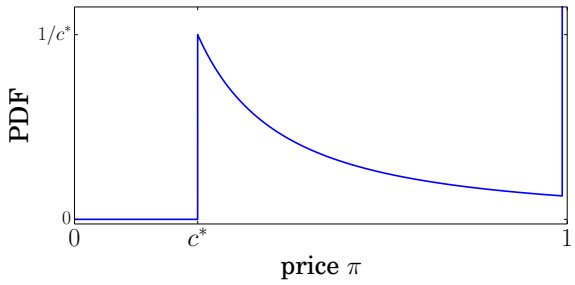
Unlike in the "at-cost" setting, we cannot in general solve for the form of the optimal pricing strategy. This is intuitively because, when we must pay the price we post, the optimal strategy depends on c_t. But the algorithm cannot condition the purchasing decision directly on c_t, as this is private information of the arriving agent.

We propose simply drawing posted prices according to the optimal strategy derived for the at-cost setting, namely,

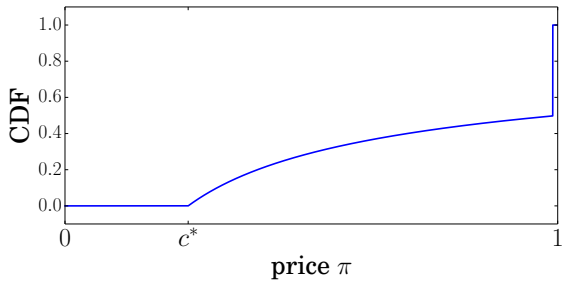
$$\Pr[\pi_t(f) \geq c] = \min\left\{1\ ,\ \frac{\Delta_{h_t,f}}{K\sqrt{c}}\right\},$$

but with a different choice of normalization constant K. More details about the imposed pricing distribution are given in Figure 6.2. This strategy gives Mechanism 6.

As in the known-costs case, our regret bounds depend upon the prior knowledge of the algorithm. It will turn out to be helpful to have prior knowledge about both γ and the following parameter,



(a) Probability density function of the pricing distribution. The distribution has a point mass at the maximum cost 1, setting price $\pi(f) = 1$ with probability $\min\{1, \Delta_{h_t,f}/K\}$. Otherwise, it draws the price $\pi(f)$ from a continuous distribution on the interval $(c^*, 1)$ with probability density function $x \mapsto \Delta_{h_t,f}/2Kx^{3/2}$.



(b) Cumulative distribution function of the pricing distribution. Equal to zero for $\pi \leq c^*$, then equal to $1 - \Delta_{h_t,f}/K\sqrt{\pi}$ on $(c^*, 1)$, then equal to 1 at cost 1.

Figure 6.2: **The pricing distribution.** Illustrates the distribution from which we draw our posted prices at time t, for a fixed arrival f. The quantity $\Delta_{h_t,f}$ captures the "benefit" from obtaining f. K is a normalization parameter. The distribution's support has a lowest price c^*, which has the form $c^* = \Delta^2_{h_t,f_t}/K^2$.

which can be interpreted as γ with all costs $c_t = 1$:

$$\gamma_{max} = \mathbb{E}\frac{1}{T}\sum_t \Delta_{h_t,f_t}.$$

Theorem 6.3.4. *If Mechanism 6 is run with prior knowledge of γ and of γ_{max} (up to a constant factor), then it can choose K and η to satisfy the expected budget constraint and obtain a regret bound of*

$$O\left(\max\left\{\frac{T}{\sqrt{B}}g\,,\ \sqrt{T}\right\}\right),$$

where $g = \sqrt{\gamma \cdot \gamma_{max}}$ (by setting $K = \frac{T}{B}\gamma_{max}$). Similarly, knowledge only of γ, respectively $\bar{c} = \frac{1}{T}\sum_t \sqrt{c_t}$, respectively $\mu = \frac{1}{T}\sum_t c_t$ suffices for the regret bound with $g = \sqrt{\gamma}$, respectively $g = \sqrt{\bar{c}}$, respectively $g = \mu^{1/4}$.

We can observe a quantifiable "price of strategic behavior" in the difference between Theorems 6.3.4 (this setting) and Theorem 6.3.3 (the "at-cost") setting:

$$\frac{T}{\sqrt{B}}\sqrt{\gamma \cdot \gamma_{max}} \quad \text{vs} \quad \frac{T}{\sqrt{B}}\gamma.$$

Mechanism 6: Mechanism for no-regret data-purchasing problem.

Input: parameters K, η, access to online learning algorithm (OLA)

set OLA parameter η;

for $t = 1, \ldots, T$ **do**

post hypothesis $h_t \leftarrow$ OLA;

for each f, post price $\pi_t(f)$ drawn independently from the distribution satisfying

$$\Pr[\pi_t(f) \geq c] = \min\left\{1\ ,\ \frac{\Delta_{h_t,f}}{K\sqrt{c}}\right\}.$$

if *we receive* (c_t, f_t) **then**

let $q_t = \Pr_{\pi_t}[\pi_t(f_t) \geq c_t]$;

let *importance-weighted loss function* $\hat{f}_t(\cdot) = \frac{f_t(\cdot)}{q_t}$;

send $\hat{f}_t \rightarrow$ OLA;

else

send 0 function $\rightarrow$ OLA;

Note that $\gamma_{max} \geq \gamma$, and they approach equality as all costs approach the upper bound 1, but become very different as the average cost $\mu \rightarrow 0$ while the maximum cost remains fixed at 1.

Comparison to lower bound. Our lower-bound for the data purchasing regret minimization problem is $\Omega\left(\frac{T}{\sqrt{B}}\gamma\right)$ (because the lower bound for the at-cost setting applies to the general setting as well), so the difference in bounds discussed above, γ to γ_{max}, is the only gap between our upper and lower bounds for the general data purchasing no regret problem.

The most immediate open problem in this paper is close this gap. Intuitively, the lower bound does not take advantage of "strategic behavior" in that a posted-price mechanism may often have to pay significantly more than the data actually costs, meaning that it obtains less data in the long run. Meanwhile, it may be possible to improve on our upper-bound strategy by drawing prices from a different distribution.

6.4 Main Results

In this section, we give the final mechanism, Mechanism 7, for the data purchasing statistical learning problem. The idea is to simply run the regret-minimization Mechanism 6 on the arriving agents. At each stage, Mechanism 6 posts a hypothesis h_t. We then aggregate these hypothesis by averaging to obtain our final prediction.

Mechanism 7: Mechanism for statistical learning data-purchasing problem.

Input: parameters K, η, access to OLA

identify each data point z with the loss function $f(\cdot) = \ell(\cdot, z)$;

run Mechanism 6 with parameters η, K and access to OLA;

let $h_1, \ldots, h_T$ be the resulting hypotheses;

output $\bar{h} = \frac{1}{T}\sum_t h_t$ (alternatively, $\bar{h}$ = a randomly chosen h_t);

Theorem 6.4.1. *Mechanism 7 guarantees spending at most B in expectation and*

$$\mathbb{E}\mathcal{L}(\bar{h}) \leq \mathcal{L}(h^*) + O\left(\max\left\{\frac{g}{\sqrt{B}}\,,\,\sqrt{\frac{1}{T}}\right\}\right),$$

where $g = \sqrt{\gamma \cdot \gamma_{max}}$, assuming that γ and γ_{max} are known in advance up to a constant factor.

If instead one assumes approximate knowledge respectively of γ, of $\bar{c} = \frac{1}{T}\sum_t \sqrt{c_t}$, or of $\mu = \frac{1}{T}\sum_t c_t$, then the guarantee holds with respectively $g = \sqrt{\gamma}$, $g = \sqrt{\bar{c}}$, or $g = \mu^{1/4}$.

Proof. By Theorem 6.3.4, Mechanism 6 guarantees an expected regret of $O\left(\max\left\{\frac{T}{\sqrt{B}}g\,,\,\sqrt{T}\right\}\right)$ when run with the specified prior knowledge for the specified values of g. Therefore, the online-to-batch conversion of Lemma 6.3.2 proves the theorem. □

The statement of Main Result 1 is the special case where only γ is known and $g = \sqrt{\gamma}$.

6.5 Deriving Pricing and the "at-cost" Variant

In this section, we dig a bit deeper into the structure of the no-regret data purchasing problem. To do so, we consider a simpler "at-cost" variant of the no-regret data purchasing problem defined in

Section 6.3.2: Rather than paying the price it posts, the mechanism is only required to pay the cost c_t of the data it is acquiring. Otherwise, the setting is exactly the same.

We first show how our posted-price strategy is derived as the optimal solution to the at-cost problem of minimizing regret subject to the budget constraint. The resulting upper bounds for the "at-cost" variant were given in Theorem 6.3.3. Then, we give some fundamental lower-bounds on regret, showing that in general our upper bounds cannot be improved upon here. These lower bounds also hold for the main no-regret data purchasing problem.

Deriving an Optimal Pricing Strategy

We begin by asking what seems to be an even easier question. Suppose that for every pair (c_t, f_t) that arrives, we could first "see" (c_t, f_t), then choose a probability with which to obtain (c_t, f_t) and pay c_t. What would be the optimal probability with which to take this data?

Lemma 6.5.1. *To minimize the regret bound of Lemma 6.3.1, the optimal choice of sampling probability is of the form*

$$q_t = \min\left\{1\ ,\ \frac{\Delta_{h_t,f_t}}{K^*\sqrt{c_t}}\right\}.$$

The normalization factor $K^* \approx \frac{T}{B}\gamma$.

The proof follows by formulating the convex programming problem of minimizing the regret bound of Lemma 6.3.1 subject to an expected budget constraint. It also gives the form of the normalization constant K^*, which depends on the input data sequence and the hypothesis sequence.

The key insight is now that we can actually achieve the sampling probabilities dictated by Lemma 6.3.1 using a randomized posted-price mechanism. First, we utilize the same pricing strategy as a function of $\Delta_{h_t,f}$ for each possible loss function f. Then whichever f_t actually arrives, we are drawing prices from the "right" distribution for that f_t. Second, once we fix f_t, notice that the optimal sampling probability is decreasing in the associated c_t. This can be achieved by drawing a posted price from the appropriate distribution. Namely, the distribution we have previously defined satisfies the dictates of Lemma 6.5.1 for *all* c_t *simultaneously*.

Thus, our final mechanism for the at-cost variant is to simply apply Mechanism 6, but only pay

the cost of the arrival rather than the price we posted. We set $K = \frac{T}{B}\gamma$. Note that this choice of normalization constant K is different from the main setting because we on average pay less in the at-cost setting; this leads to the difference in the regret bounds. Our main bound for the at-cost variant was given in Theorem 6.3.3. An open problem for this setting is to attempt to obtain the same regret bounds, but without being told anything at all about the arriving costs and data.

Lower bounds for data purchasing regret minimization

Here, we prove lower bounds analogous to the classic regret lower bound, which states that no algorithm can guarantee to do better than $O\left(\sqrt{T}\right)$. These lower bounds will hold even in the "at-cost" setting, where they match our upper bounds. An open problem is to obtain a larger-order lower bound for the main setting where the mechanism pays its posted price. This would show a separation between the problems.

First, we give what might be considered a "sample complexity" lower bound for no-regret learning: It specializes our setting to the case where all costs are equal to one (and this is known to the algorithm in advance), so the question is what regret is achievable by an algorithm that observes B of the T arrivals.

Theorem 6.5.2. *Suppose all costs $c_t = 1$. No algorithm for the at-cost online data-purchasing problem has regret better than $O\left(\frac{T}{\sqrt{B}}\right)$; that is, for every algorithm, there exists an input sequence on which its regret is $\Omega\left(\frac{T}{\sqrt{B}}\right)$.*

Proof Idea: We will have two coins, with probabilities $\frac{1}{2} \pm \epsilon$ of coming up heads. We will take one of the coins and provide T i.i.d. flips as the input sequence. The possible hypotheses for the algorithm are $\{\text{heads}, \text{tails}\}$ and the loss is zero if the hypothesis matches the flip and zero otherwise. The cost of every data point will be one.

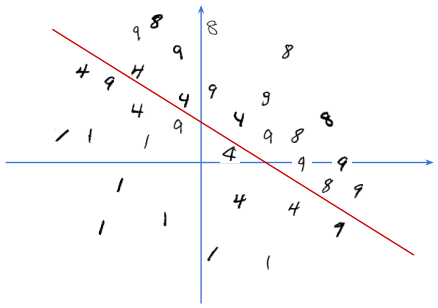
The idea is that an algorithm with regret much smaller than $T\epsilon$ must usually predict heads if it is the heads-biased coin and usually predict tails if it is the tails-biased coin. Thus, it can be used to distinguish these cases. However, there is a lower bound of $\Omega\left(\frac{1}{\epsilon^2}\right)$ samples required to distinguish the coins, and the algorithm only has enough budget to gain information about $O(B)$ of the samples. Setting $\epsilon = 1/\sqrt{B}$ gives the regret bound. □

We next extend this idea to the case with heterogeneous costs. The idea is very simple: Begin with the problem from the label-complexity lower bound, and introduce "useless" data points and heterogeneous costs. The worst or "hardest" case for a given average cost is when cost is perfectly correlated with benefit, so all and only the "useful" data points are expensive.

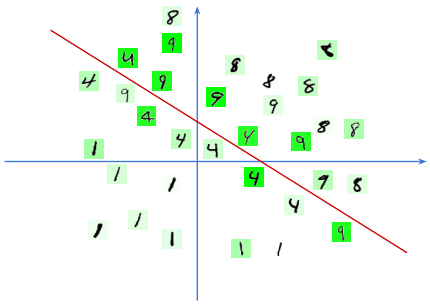
Theorem 6.5.3. *No algorithm for the non-strategic online data-purchasing problem has expected regret better than* $O\left(\gamma T/\sqrt{B}\right)$*; that is, for every* γ*, for every algorithm, there exists a sequence with parameter* γ *on which its regret is* $\Omega\left(\gamma\frac{T}{\sqrt{B}}\right)$*. Similarly, for* $\bar{c} = \frac{1}{T}\sum_t \sqrt{c}$ *and* $\mu = \frac{1}{T}\sum_t c_t$*, we have the lower bounds* $\Omega\left(T\bar{c}/\sqrt{B}\right)$ *and* $\Omega\left(T\mu/\sqrt{B}\right)$*.*

6.6 Examples and Experiments

In this section, we give some examples of the performance of our mechanisms on data. We use a binary classification problem with feature vector $x \in \mathbb{R}^d$ and label $y \in \{-1, 1\}$. The dataset is described in Figure 6.3.



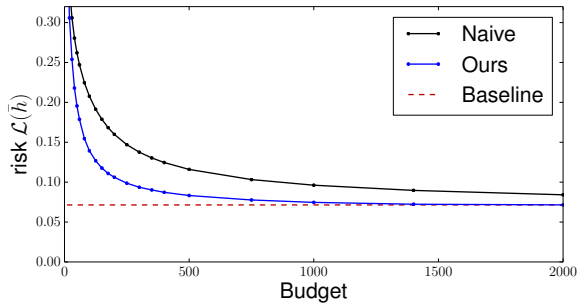
(a) Visualizing the classification problem without costs.



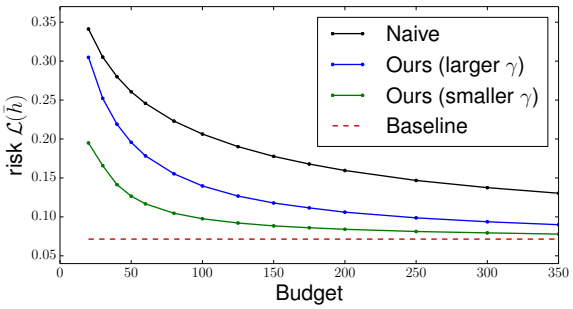
(b) A brighter green background corresponds to a higher-cost data point.

Figure 6.3: **Dataset.** Data points are images of handwritten digits, each data point consisting of a feature vector x of grayscale pixels and a label y, the digit it depicts. We use the MNIST handwritten digit dataset (http://yann.lecun.com/exdb/mnist/). The algorithm is asked to distinguish between two "categories" of digits, where "positive" examples are digits 9 and 8 and "negative" examples are 1 and 4 (all other digits are not used), giving $T = 8503$. This task allows us to adjust the correlations by drawing costs differently for different digits.

The hypothesis is a hyperplane classifier, *i.e.* vector w where the example is classified as



(a) A comparison of mechanisms. "Naive" offers a maximum price of 1 to every arrival until out of budget. "Ours" is Mechanism 7, with K initialized to 0 and then adjusted online according to the estimated average γ on the data so far. "Baseline" obtains every data point (has no budget constraint). Costs are distributed uniform $(0, 1)$ independently. Each datapoint is an average of 4000 trials, with standard error of at most 0.0002.



(b) An illustration of the role of cost-data correlations. The marginal distribution of costs is 1 with probability 0.2 and free otherwise, but the correlation of cost and data changes. The performance of Naive and the Baseline do not change with correlations. The larger-γ case has high-cost points consisting of only 4s and 9s, while γ is smaller when costs and data are independent. Each datapoint is an average of 2000 trials, with standard error of at most 0.0004.

Figure 6.4: **Examples of mechanism performance.**

positive if $w \cdot x \geq 0$ and negative otherwise; the risk is therefore the error rate (fraction of examples misclassified). For the implementation of the online gradient descent algorithm, we use a "convexified" loss function, the well-known hinge loss: $\ell(w, (x, y)) = \max\{0, 1 - y(w \cdot x)\}$ where $y \in \{-1, 1\}$.

In our simulations, we give each mechanism access to the exact same implementation of the Online Gradient Descent algorithm, including the same parameter η chosen to be $0.1/c$ where c is the average norm of the data feature vectors. We train on a randomly chosen half of the dataset and test on the other half.

The "baseline" mechanism has no budget cap and purchases every data point. The "naive" mechanism offers a maximum price of 1 for every data point until out of budget. "Ours" is an implementation of Mechanism 7. We do not use any prior knowledge of the costs at all: We initialize $K = 0$ and then adjust K online by estimating γ from the data purchased so far. (For a symmetric comparison, we do not adjust η accordingly; instead we leave it at the same value as used with the

other mechanisms.) The examples are shown in Figure 6.4.

6.7 Summary

In this chapter, we propose an *active* scheme for learning and pricing data as it arrives online, held by strategic agents. The active approach allows learning from past data and selectively pricing future data. Our mechanisms interface with existing no-regret algorithms in an essentially black-box fashion (although the proof depends on the specific class of algorithms). The analysis relies on showing that they have good guarantees in a model of no-regret learning with purchased data. This no-regret setting may be of interest in future work, to either achieve good guarantees with no foreknowledge at all other than the maximum cost, or to propose variants on the model.

The no-regret analysis means our mechanisms are robust to adversarial input. But in nicer settings, one might hope to improve on the guarantees. One direction is to assume that costs are drawn according to a known marginal distribution (although the correlation with the data is unknown). A combination of our approach and the posted-price distributions of Roth and Schoenebeck [98] may be fruitful here.

Broadly, the problem of purchasing data for learning has many potential models and directions for study. One motivating setting, closer to crowdsourcing, is an active problem where data points consist of pairs (example, label) and the mechanism can offer a price for anyone who obtains the label of a given example. In an online arrival scheme, such a mechanism could build on the importance-weighted active learning paradigm [15].

CHAPTER 7

Conclusion and Future Directions

In this dissertation, we have explored the design and analysis of crowdsourcing mechanisms. We address the challenges that crowdworkers might not always give trustworthy information and developed algorithms and techniques to obtain high-quality contributions from crowdworkers.

We first study how to assign tasks to suitable workers with initially unknown skills in Chapter 2 using techniques from machine learning and online optimization. We show that our algorithm can learn to assign tasks to workers near-optimally even when the workers' arriving sequence and workers' skill levels are initially unknown. In Chapter 3 and 4, we explore how to incentive workers to exert high effort using game theoretic approaches. We demonstrate the use of reputations and performance-based payments as incentives to encourage high quality contribution. Next, in Chapter 5, we examine whether our assumptions on workers' incentives hold in practice. We conduct online behavioral experiments to understand how humans behave in the real systems and develop a more realistic worker model with the support of empirical evidence. Finally, in Chapter 6, we explore the setting in which we need to jointly consider the learning/optimization problems and workers' incentives.

There are still abundant of interesting open questions in better utilizing the wisdom of crowd. We conclude this dissertation by outlining a selection of future research directions.

7.1 Learning with strategic agents

The growth of Web technology has raised many interesting machine learning challenges. For example, Facebook might want to learn users' preferences and interests to provide targeted advertising. LinkedIn might want to learn the best matching between employers and employees to facilitate the

job-seeking process. However, these problems all involve learning from data generated or possessed by strategic users. To enable learning, we need to apply different mechanisms and observe agents' behavioral changes. However, changing mechanisms also changes agents' incentives, and agents will take different actions. Therefore, it is necessary to consider agents' incentives while learning from their behavior.

In Chapter 6, we have addressed this problem in a special case where strategic agents posses data that they cannot fabricate. However, there are more interesting cases when agents can make more sophisticated choices, such as the amount of effort to exert. These problems present new challenges both in theory and in practice. In order to make progress in this domain, it is important to incorporate the techniques from game theory into machine learning. For example, one natural direction would be to introduce solution concepts in game theory as either the constraints or objectives in the learning problem we are interested in.

7.2 Better understanding human behavior

Most of the work on the design of crowdsourcing mechanisms assumes simple user models. For example, users' skills are represented by a probability distribution, and users' utility is a quasi-linear function of payments minus costs. However, these assumptions fail in many scenarios. Users might improve their skills over time and might have different objectives. It is important to develop more realistic user behavior models. In particular, it is important to conduct extensive experiments on various crowdsourcing platforms, such as Amazon Mechanical Turk, to develop and validate models using empirical observations.

This problem is especially interesting since some of the recent works [40, 49] have demonstrated that a small deviation on the worker model could hugely influence the optimal design of mechanisms. In Chapter 5, we have empirically studied how online users react to different performance-based payments and developed theoretical models based on our empirical observations. However, we only focus on one single aspect of how workers behave. In order to develop practical theories and algorithms to utilize human intelligence, it is important to identify the key features of human behavioral models, empirically examine them, and develop realistic theories.

7.3 Crowdsourcing complex tasks to groups of workers

Most work in crowdsourcing assumes we only interact with one worker at a time. This dissertation has also been mainly focusing on either eliciting high-quality responses from individual workers or aggregating multiple noisy responses into a high-quality one. However, real-world tasks are often complex and cannot be completed by individual workers. For example, crisis mapping, which aims at utilizing online users to gather and analyze data during a crisis (such as a natural disaster), requires the collaboration of a group of users. It is therefore important to design methods to coordinate human effort for complex tasks.

There are several interesting research directions. For example, can we design incentive mechanisms to encourage a group of workers to collaborate with each other while rewarding workers fairly? Can we automatically decompose a complex task into small microtasks? Given the decomposition, how do we design efficient workflows to solve the decomposed microtasks and aggregate the results? It would be important to tackle these problems using techniques from Artificial Intelligence, Economics, and Social Sciences.

APPENDIX A

Appendix

A.1 Additional Proofs from Chapter 2

A.1.1 Proof of Lemma 2.3.1

Without loss of generality, assume that $\ell_i = 1$. Let $X_{i,j}$ be a random variable which represents the weighted label, so

$$X_{i,j} = \begin{cases} w_{i,j}, & \text{with probability } p_{i,j}, \\ -w_{i,j}, & \text{with probability } 1 - p_{i,j}, \end{cases}$$

and let $X_i = \sum_{j \in J_i} X_{i,j}$.

Recall that we predict that task i has label 1 if $X_i \geq 0$ and has label -1 otherwise. Since the true label is 1, bounding $\Pr(X_i \leq 0)$ would give us a bound on the probability of an error. We will need that

$$\begin{aligned} E[X_i] &= \sum_{j \in J_i} E[X_{i,j}] \\ &= \sum_{j \in J_i} (p_{i,j} w_{i,j} - (1 - p_{i,j}) w_{i,j}) \\ &= \sum_{j \in J_i} (w_{i,j}(2p_{i,j} - 1)). \end{aligned}$$

Note that $\Pr(X_i \leq 0) = \Pr(E[X_i] - X_i \geq E[X_i])$. By Hoeffding's inequality, we have

$$\begin{aligned} \Pr(X_i \leq 0) &\leq \exp\left(-\frac{2(E[X_i])^2}{\sum_{j \in J_i} (2w_{i,j})^2} \right) \\ &= \exp\left(-\frac{(\sum_{j \in J_i} w_{i,j}(2p_{i,j} - 1))^2}{2 \sum_{j \in J_i} w_{i,j}^2} \right). \end{aligned}$$

This shows the first part of this lemma. This error bound is maximized when the expression

$$\frac{(\sum_{j \in J_i} w_{i,j}(2p_{i,j} - 1))^2}{2\sum_{j \in J_i} w_{i,j}^2}$$

is minimized. Setting the gradient of this expression to $\mathbf{0}$, we see that this happens when for every k,

$$\frac{\sum_{j \in J_i} w_{i,j}(2p_{i,j} - 1)}{\sum_{j \in J_i} w_{i,j}^2} w_{i,k} = 2p_{i,k} - 1,$$

i.e., when $w_{i,k} \propto 2p_{i,k} - 1$. (Note that scaling the weights will not change the bound since $\hat{\ell}_i$ will not change.) Plugging $w_{i,j} = 2p_{i,j} - 1$ into the bound from the first half of the lemma, we get that

$$\begin{aligned}\Pr(X_i \leq 0) &\leq \exp\left(-\frac{(\sum_{j \in J_i} q_{i,j})^2}{2\sum_{j \in J_i} q_{i,j}}\right) \\ &= \exp\left(-\frac{1}{2}\sum_{j \in J_i} q_{i,j}\right).\end{aligned}$$

□

A.1.2 Proof of Theorem 2.3.2

The bulk of this proof involves showing that there exists a primal optimal $\bar{\mathbf{y}}^*$ for the relaxed linear program such that for at most $\min(m, n)$ pairs of (i, j), $\bar{y}^*_{i,j} \neq y_{i,j}$. Therefore, since $\bar{y}^*_{i,j}, y_{i,j} \in [0, 1]$ for all i and j, $\sum_{i=1}^n \sum_{j=1}^m (y_{i,j} - \bar{y}^*_{i,j}) \leq \min(m, n)$. To complete the proof, we will use the fact that $\mathbf{y}^*$ is feasible in the relaxed linear program to show that this implies the result.

We start with a helpful lemma that characterizes the dual optimal solution of the relaxed linear program.

Lemma A.1.1. *If $\mathbf{x}^*$ is the optimal value of $\mathbf{x}$ in any dual optimal solution, then there exists a dual optimal solution $(\mathbf{x}^*, \mathbf{z}^*, \mathbf{t}^*)$ such that for each $j \in \{1, \cdots, m\}$,*

$$z_j^* = \begin{cases} 0, & \text{if } n_j \leq M_j, \\ v_{c_j,j}, & \text{otherwise} \end{cases}$$

where $v_{i,j} = q_{i,j}x_i^* - 1$ is the task value for (i, j) and c_j is the task with the M_jth largest task value for worker j among all tasks.

Proof. We prove the lemma in two steps. In Step 1, we show that there cannot exist a dual optimal $(\mathbf{x}^*, \bar{\mathbf{z}}, \bar{\mathbf{t}})$ such that $\bar{z}_j \neq v_{c_j,j}$ for some j such that $n_j > M_j$. In Step 2, we show that if there exists a dual optimal $(\mathbf{x}^*, \bar{\mathbf{z}}, \bar{\mathbf{t}})$ such that $\bar{z}_j \neq 0$ for some j with $n_j \leq M_j$, then there exists another dual optimal solution $(\mathbf{x}^*, \bar{\mathbf{z}}', \bar{\mathbf{t}}')$ such that

$$\bar{z}'_k = \begin{cases} \bar{z}_k, & \text{if } k \neq j, \\ 0, & \text{if } k = j, \end{cases} \tag{A.1}$$

and

$$\bar{t}'_{i,k} = \begin{cases} \bar{t}_{i,k}, & \text{if } k \neq j \text{ or } k = j \text{ and } v_{i,j} < 0, \\ \bar{t}_{i,k} + \bar{z}_j, & \text{otherwise.} \end{cases} \tag{A.2}$$

Therefore, starting with any dual solution, we can transform it into a solution satisfying the condition that $z^*_j = 0$ if $n_j \leq M_j$ by repeating this transformation for any workers j for which the condition did not originally hold.

In the following proof, we define the function g to represent the dual objective:

$$g(\mathbf{x}, \mathbf{z}, \mathbf{t}) = C_\epsilon \sum_{i=1}^{n} x_i - \sum_{j=1}^{m} M_j z_j - \sum_{i=1}^{n} \sum_{j=1}^{m} t_{i,j}.$$

Recall that the dual constraints are

$$1 - q_{i,j} x_i + z_j + t_{i,j} \geq 0 \; \forall (i,j),$$
$$x_i, z_j, t_{i,j} \geq 0.$$

Step 1: Assume by contradiction that there exists a dual optimal $(\mathbf{x}^*, \bar{\mathbf{z}}, \bar{\mathbf{t}})$ and worker j such that $\bar{z}_j \neq v_{c_j,j}$ and $n_j > M_j$. Below we show that we can always generate a dual feasible solution $(\mathbf{x}^*, \bar{\mathbf{z}}', \bar{\mathbf{t}}')$ which leads to higher dual objective. Therefore, $(\mathbf{x}^*, \bar{\mathbf{z}}, \bar{\mathbf{t}})$ cannot be optimal.

Define

$$\bar{z}'_k = \begin{cases} \bar{z}_k, & \text{if } k \neq j, \\ v_{c_j,j}, & \text{if } k = j, \end{cases}$$

and

$$\bar{t}'_{i,k} = \begin{cases} \bar{t}_{i,k}, & \text{if } k \neq j \text{ or } k = j \text{ and } v_{i,j} < 0 \\ & \text{or } k = j \text{ and } v_{i,j} \geq 0 \text{ and } \bar{z}_j > v_{c_j,j}, \\ \bar{t}_{i,k} + v_{c_j,j} - \bar{z}_j, & \text{otherwise.} \end{cases}$$

1) Suppose that $\bar{z}_j < v_{c_j,j}$. We can observe that $\bar{z}'_j > \bar{z}_j$ and $\bar{t}'_{i,k} \geq \bar{t}_{i,k}$ for all k. Since $(\mathbf{x}^*, \bar{\mathbf{z}}, \bar{\mathbf{t}})$ is dual feasible, We can verify that $(\mathbf{x}^*, \bar{\mathbf{z}}', \bar{\mathbf{t}}')$ is also dual feasible.

Next we show that $(\mathbf{x}^*, \bar{\mathbf{z}}', \bar{\mathbf{t}}')$ leads to higher dual objective than $(\mathbf{x}^*, \bar{\mathbf{z}}, \bar{\mathbf{t}})$.

$$\begin{aligned} & g(\mathbf{x}^*, \bar{\mathbf{z}}, \bar{\mathbf{t}}) - g(\mathbf{x}^*, \bar{\mathbf{z}}', \bar{\mathbf{t}}') \\ &= -M_j \bar{z}_j + M_j v_{c_j,j} - \sum_{i: v_{i,j} \geq 0} v_{c_j,j} + \sum_{i: v_{i,j} \geq 0} \bar{z}_j \\ &= (\bar{z}_j - v_{c_j,j})(n_j - M_j) < 0. \end{aligned}$$

Therefore, $(\mathbf{x}^*, \bar{\mathbf{z}}, \bar{\mathbf{t}})$ couldn't have been optimal.

2) Suppose that $\bar{z}_j > v_{c_j,j}$. Note that $(\mathbf{x}^*, \bar{\mathbf{z}}, \bar{\mathbf{t}})$ is dual feasible, and the only change in $(\mathbf{x}^*, \bar{\mathbf{z}}', \bar{\mathbf{t}}')$ is that $\bar{z}'_j = v_{c_j,j}$. We need only examine if $-v_{c_j,j} + \bar{z}'_j + \bar{t}_{i,j} \geq 0$ to check the dual feasibility. Since $\bar{t}'_{i,j} \geq 0$, we know $(\mathbf{x}^*, \bar{\mathbf{z}}', \bar{\mathbf{t}}')$ is dual feasible.

Furthermore, $(\mathbf{x}^*, \bar{\mathbf{z}}', \bar{\mathbf{t}}')$ also leads to higher dual objective:

$$\begin{aligned} g(\mathbf{x}^*, \bar{\mathbf{z}}, \bar{\mathbf{t}}) - g(\mathbf{x}^*, \bar{\mathbf{z}}', \bar{\mathbf{t}}') &= -M_j \bar{z}_j + M_j v_{c_j,j} \\ &= M_j (v_{c_j,j} - \bar{z}_j) < 0. \end{aligned}$$

Therefore, $(\mathbf{x}^*, \bar{\mathbf{z}}, \bar{\mathbf{t}})$ couldn't have been optimal. This shows that there cannot be a dual optimal solution with $z_j^* \neq v_{c_j,j}$ for some j with $n_j > M_j$.

Step 2: Assume there exists a dual optimal solution $(\mathbf{x}^*, \bar{\mathbf{z}}, \bar{\mathbf{t}})$ and worker j such that $\bar{z}_j > 0$ and $n_j \leq M_j$. Below we show that $(\mathbf{x}^*, \bar{\mathbf{z}}', \bar{\mathbf{t}}')$, as defined in Equations A.1 and A.2, is dual feasible, and the dual objective $g(\mathbf{x}^*, \bar{\mathbf{z}}', \bar{\mathbf{t}}')$ is no less than $g(\mathbf{x}^*, \bar{\mathbf{z}}, \bar{\mathbf{t}})$.

To show feasibility, first observe that for all $k \neq j$, the dual variables $\bar{z}'_k$ and $\bar{t}'_{i,k}$ are not changed. Therefore, we need only check if the constraint $-v_{i,k} + \bar{z}'_k + \bar{t}'_{i,k} \geq 0$ holds for all i and $k = j$. Observing Equations A.1 and A.2, when $v_{i,j} < 0$, the constraint trivially holds since $\bar{z}'_k$ and $\bar{t}'_{i,k}$ are

nonnegative. When $v_{i,j} \geq 0$, the left-hand side of the constraint can be written as $v_{i,j} + \bar{t}_{i,k} + \bar{z}_j$, which is larger then or equal to 0 since $(\mathbf{x}^*, \bar{\mathbf{z}}, \bar{\mathbf{t}})$ is dual feasible. Therefore, $(\mathbf{x}^*, \bar{\mathbf{z}}', \bar{\mathbf{t}}')$ is also dual feasible.

Furthermore, we have

$$\begin{aligned} g(\mathbf{x}^*, \bar{\mathbf{z}}, \bar{\mathbf{t}}) - g(\mathbf{x}^*, \bar{\mathbf{z}}', \bar{\mathbf{t}}') &= -M_j \bar{z}_j + \sum_{i: v_{i,j} \geq 0} \bar{z}_j \\ &= (-M_j + n_j)\bar{z}_j \leq 0. \end{aligned}$$

Therefore, $(\mathbf{x}^*, \bar{\mathbf{z}}', \bar{\mathbf{t}}')$ must be an optimal dual solution.

Combining this with Step 1, we can always transform any dual solution $(\mathbf{x}^*, \mathbf{z}^*, \mathbf{t}^*)$ into another that satisfies the properties in the lemma without changing $\mathbf{x}^*$. □

Given the dual optimal above, using complementary slackness, we can characterizes the primal optimal solution $\bar{\mathbf{y}}^*$. Below we list the cases in which $\bar{y}^*_{i,j}$ takes on integer values. Looking ahead, we will see that in all of these cases, $\bar{y}^*_{i,j}$ and $y_{i,j}$ do not differ. We can then show there are at most $\min(m, n)$ pairs of $y_{i,j}$ such that $y_{i,j} \neq \bar{y}^*_{i,j}$.

Lemma A.1.2. *Consider a dual optimal solution $(\mathbf{x}^*, \mathbf{z}^*, \mathbf{t}^*)$ satisfying the criteria in Lemma A.1.1, and let $\bar{\mathbf{y}}^*$ be the corresponding primal optimal solution to the relaxed linear programming problem. Let $v_{i,j} = q_{i,j}x^*_i - 1$ be the task value for (i, j) and c_j be the task with the M_jth largest task value for worker j among all tasks. For all (i, j),*

1. *If $v_{i,j} < 0$ then $\bar{y}^*_{i,j} = 0$.*
2. *If $v_{i,j} > 0$ & $n_j \leq M_j$ then $\bar{y}^*_{i,j} = 1$.*
3. *If $v_{i,j} > 0$, $n_j > M_j$, & $v_{i,j} > v_{c_j,j}$, then $\bar{y}^*_{i,j} = 1$.*
4. *If $v_{i,j} > 0$, $n_j > M_j$, & $v_{i,j} < v_{c_j,j}$, then $\bar{y}^*_{i,j} = 0$.*

Proof. We prove the four parts in turn. First, it is useful to note that by complementary slackness,

we have

$$\bar{y}^*_{i,j}(1 - q_{i,j}x^*_i + z^*_j + t^*_{i,j}) = 0 \ \ \forall (i,j), \tag{A.3}$$

$$t^*_{i,j}(\bar{y}^*_{i,j} - 1) = 0 \ \ \forall (i,j). \tag{A.4}$$

Part 1: The first part is simple. Since $z^*_j, t^*_{ij} \geq 0$, if $v_{i,j} < 0$, it must be the case that $1 - q_{i,j}x^*_i + z^*_j + t^*_{i,j} > 0$. From Equation A.3, we have $\bar{y}^*_{i,j} = 0$.

Part 2: From Lemma A.1.1, if $n_j \leq M_j$ then $z^*_j = 0$. When $z^*_j = 0$ and $v_{i,j} > 0$, we must have that $t^*_{i,j} \geq v_{i,j}$ in order to satisfy the first dual constraint. Equation A.4 then implies that $\bar{y}^*_{i,j} = 1$.

Part 3: From Lemma A.1.1, if $n_j > M_j$ then $z^*_j = v_{c_j,j}$. In this case, when $v_{i,j} > v_{c_j,j}$, by the dual constraints we must have $t^*_{i,j} \geq v_{i,j} - v_{c_j,j} > 0$, and by Equation A.4, $\bar{y}^*_{i,j} = 1$.

Part 4: Again, from Lemma A.1.1, since $n_j > M_j$ we know that $z^*_j = v_{c_j,j}$. In this case, if $v_{i,j} < v_{i',j}$, then any $t_{i,j} \geq 0$ is feasible. Since $t_{i,j}$ appears negated in the objective and does not appear in other constraints, we must have $t^*_{i,j} = 0$. This implies that $1 - q_{i,j}x^*_i + z^*_j + t^*_{i,j} > 0$, and by Equation A.3, $\bar{y}^*_{i,j} = 0$. □

One can verify that in all of the cases covered by Lemma A.1.2, $\bar{y}^*_{i,j} = y_{i,j}$. Notice that the only case not covered by this lemma is the case in which $v_{i,j} > 0$, $n_j > M_j$, and $v_{i,j} = v_{c_j,j}$. Since we have assumed that $v_{i,j} \neq v_{i',j}$ unless $i = i'$, this can happen at most once for each worker j, and, therefore, at most m times in total. In cases where $y_{i,j}$ and $\bar{y}^*_{i,j}$ differ, $y_{i,j} = 1$ and $\bar{y}^*_{i,j} \in [0,1]$. Hence,

$$\sum_{i=1}^{n}\sum_{j=1}^{m} y_{i,j} - \sum_{i=1}^{n}\sum_{j=1}^{m} \bar{y}^*_{i,j} \leq m. \tag{A.5}$$

Since $y_{i,j} \in \{0,1\}$, we know that for each worker j, there exists at most one task i such that $\bar{y}^*_{i,j} \notin \{0,1\}$. Furthermore, we know that if such a task exists, the Primal Approximation Algorithm assigns the task to worker j, i.e., $y_{i,j} = 1$. Below we further argue that for each task i, there exists at most one worker j such that $\bar{y}^*_{i,j} \notin \{0,1\}$, and this worker is the one with minimum skill level $q_{i,j}$ among all the workers with $y_{i,j} = 1$. Hence, the difference in Equation A.5 can be upper bounded by n as well.

Let j_i be the worker with the minimum skill level among all the workers with $y_{i,j} = 1$. As we will describe in later in this section, we can add random noise to the $q_{i,j}$ values such that $q_{i,j} \neq q_{i',j'}$

unless $i = i'$ and $j = j'$. Assume for contradiction that there exists a worker $j \neq j_i$ such that $\bar{y}^*_{i,j} \notin \{0, 1\}$. We know that the Primal Approximation Algorithm sets $y_{i,j}$ to 1 when $\bar{y}^*_{i,j} \notin \{0, 1\}$. Therefore, we have $q_{i,j} > q_{i,j_i}$ by definition of j_i. Let $\Delta = \min(q_{i,j_i}\bar{y}^*_{i,j_i}/q_{i,j}, 1 - \bar{y}^*_{i,j})$. Suppose that we increased the value of $\bar{y}^*_{i,j}$ by Δ and decreased the value of $\bar{y}^*_{i,j_i}$ by $q_{i,j}\Delta/q_{i,j_i}$. It is easy to verify that no constraints of the relaxed offline formulation would be violated, but the objective would decrease since $q_{i,j} > q_{i,j_i}$. This is a contradiction. Therefore, for each task i, there exists at most one worker j such that $\bar{y}^*_{i,j} \notin \{0, 1\}$, and this worker is the one with minimum skill level $q_{i,j}$ among all the workers with $y_{i,j} = 1$. Hence,

$$\sum_{i=1}^{n}\sum_{j=1}^{m} y_{i,j} - \sum_{i=1}^{n}\sum_{j=1}^{m} \bar{y}^*_{i,j} \leq n. \tag{A.6}$$

Combining Equations A.5 and A.6

$$\sum_{i=1}^{n}\sum_{j=1}^{m} y_{i,j} - \sum_{i=1}^{n}\sum_{j=1}^{m} \bar{y}^*_{i,j} \leq \min(m, n).$$

Finally, since $\mathbf{y}^*$ (the optimal solution of the IP) is feasible in the relaxed linear program where $\bar{\mathbf{y}}^*$ is optimal, we know that

$$\sum_{i=1}^{n}\sum_{j=1}^{m} \bar{y}^*_{i,j} \leq \sum_{i=1}^{n}\sum_{j=1}^{m} y^*_{i,j}.$$

Putting these together, we have

$$\sum_{i=1}^{n}\sum_{j=1}^{m} y_{i,j} - \sum_{i=1}^{n}\sum_{j=1}^{m} y^*_{i,j} \leq \min(m, n),$$

which completes the proof of Theorem 2.3.2. □

A.1.3 Discussion of Perturbation Assumption

In Section 2.3.3 where we introduced the perturbation assumption, we claimed that we can add small random perturbations (noise) to the $q_{i,j}$ values as in Devanur and Hartline [36] to satisfy the condition that $q_{i,j}x_i \neq q_{i',j}x_{i'}$ for $i \neq i'$. In this section, we explain the details of the claim.

First of all, note that if all the $q_{i,j}$ values *(i)* are determined independently, and, *(ii)* are in general position in $\mathbb{R}$, then the probability that any two $q_{i,j}$ values are the same is 0 [36]. However, in our

solution, the $q_{i,j}$ values are estimated by our exploration phase. Since the number of exploratory assignments are small, there is only a limited number of possible $q_{i,j}$ values. Therefore, the probability of the $q_{i,j}$ values being all different is clearly *non-zero*. To remedy this, we can add a small amount of noise to the $q_{i,j}$ values after estimating them empirically. If the noise is drawn from a uniform distribution, the new noisy $q_{i,j}$ values will be in general position. Therefore, the probability that two noisy $q_{i,j}$ values are the same is 0 after adding the noise. Note that we can set the random noise arbitrarily small to reduce the effects of it in our analysis. For example, if we know the error in estimating the $q_{i,j}$ values is bounded by t, then the amount of random noise can be set to something much smaller than t.

Second, we show that when $q_{i,j}$ values are all different, at most $n-1$ ties in the $q_{i,j}x_i^*$ values can happen. Note that for some worker j and two tasks i and i', if $q_{i,j}x_i^* = q_{i',j}x_{i'}^*$, then $x_i^* = x_{i'}^*(q_{i',j}/q_{i,j})$. Hence, in the worst case, an adversary could create at most $n-1$ ties by adjusting the $\mathbf{x}^*$ values since the noisy $q_{i,j}$ values are in general position.

A.1.4 Proof of Theorem 2.4.1

We first characterize some properties of the Primal Approximation Algorithm. In order to do this, we consider the expanded version of the algorithm in Algorithm 8. Its behavior is identical to the algorithm stated in Section 2.3.3 in terms of the way that the assignments $y_{i,j}$ are made, but it also includes settings for the dual variables $\mathbf{z}$ and $\mathbf{t}$ as well.

This expanded version of the algorithm can be used to prove the following lemma.

Lemma A.1.3. *Given any inputs* $\mathbf{x}$ *and* $\mathbf{q}$*, let* $\mathbf{y}$ *and* $(\mathbf{z}, \mathbf{t})$ *be the primal assignments and dual variables set by the Expanded Primal Approximation Algorithm. Let* $P(\mathbf{y})$ *and* $D(\mathbf{x}, \mathbf{z}, \mathbf{t})$ *denote the primal and dual objective. Then under the perturbation assumption,*

$$P(\mathbf{y}) = D(\mathbf{x}, \mathbf{z}, \mathbf{t}) + \sum_{i=1}^{n} \left(\sum_{j=1}^{m} q_{i,j} y_{i,j} - C_\epsilon \right) x_i.$$

Proof. By examining the Expanded Primal Approximation Algorithm, one can easily verify the following properties.

ALGORITHM 8: The Expanded Primal Approximation Algorithm, which explicitly sets dual variables.

Input: Values x_i and $q_{i,j}$ for all (i, j)

Output: Values $y_{i,j}$, $t_{i,j}$, z_j for all (i, j)

Calculate task values $v_{i,j} = q_{i,j}x_i - 1$ for all i.

if there are no more than M_j tasks with $v_{i,j} \geq 0$ **then**

Aet z_j to 0.

for every task i with $v_{i,j} \geq 0$ **do**

Set $t_{i,j} = v_{i,j} = q_{i,j}x_i - 1$.

Set $y_{i,j} = 1$.

end for

for every task i with $v_{i,j} < 0$ **do**

Set $t_{i,j} = 0$.

Set $y_{i,j} = 0$.

end for

end if

if there are more than M_j tasks with $v_{i,j} \geq 0$ **then**

Set z_j to a value such that there are exactly M_j tasks with $v_{i,j} - z_j \geq 0$.

for every task i with $v_{i,j} - z_j \geq 0$ **do**

Set $t_{i,j} = v_{i,j} - z_j = q_{i,j}x_i - 1 - z_j$.

Set $y_{i,j} = 1$.

end for

for every task i with $v_{i,j} - z_j < 0$ **do**

Set $t_{i,j} = 0$.

Set $y_{i,j} = 0$.

end for

end if

1. If $t_{i,j} > 0$, then $y_{i,j} = 1$.

2. $\sum_{i=1}^{n} y_{i,j} \leq M_j$.

3. If $\sum_{i=1}^{n} y_{i,j} < M_j$, then $z_j = 0$.

4. If $y_{i,j} = 1$, then $z_j + t_{i,j} = q_{i,j}x_i - 1$.

By definition, the dual objective can be written as

$$D(\mathbf{x}, \mathbf{z}, \mathbf{t}) = C_\epsilon \sum_{i=1}^{n} x_i - \sum_{j=1}^{m} M_j z_j - \sum_{i=1}^{n}\sum_{j=1}^{m} t_{i,j}.$$

Note that the first property implies $\Sigma_{i=1}^{n}\Sigma_{j=1}^{m} t_{i,j} = \Sigma_{i=1}^{n}\Sigma_{j=1}^{m} t_{i,j}y_{i,j}$. The second and third properties together imply $M_j z_j = (\Sigma_{i=1}^{n} y_{i,j})z_j$. Hence,

$$\begin{aligned} D(\mathbf{x}, \mathbf{z}, \mathbf{t}) &= C_\epsilon \sum_{i=1}^{n} x_i - \sum_{j=1}^{m}(\sum_{i=1}^{n} y_{i,j})z_j - \sum_{i=1}^{n}\sum_{j=1}^{m} t_{i,j}y_{i,j} \\ &= C_\epsilon \sum_{i=1}^{n} x_i - \sum_{i=1}^{n}\sum_{j=1}^{m}(z_j + t_{i,j})y_{i,j}. \end{aligned}$$

Finally, since $y_{i,j} \in \{0, 1\}$, the fourth property implies that $(z_j + t_{i,j})y_{i,j} = (q_{i,j}x_i - 1)y_{i,j}$. Therefore,

$$\begin{aligned} D(\mathbf{x}, \mathbf{z}, \mathbf{t}) &= C_\epsilon \sum_{i=1}^{n} x_i - \sum_{i=1}^{n}\sum_{j=1}^{m}(q_{i,j}x_i - 1)y_{i,j} \\ &= C_\epsilon \sum_{i=1}^{n} x_i - \sum_{i=1}^{n}\sum_{j=1}^{m}(q_{i,j}y_{i,j})x_i + \sum_{i=1}^{n}\sum_{j=1}^{m} y_{i,j} \\ &= P(\mathbf{y}) - \sum_{i=1}^{n}\left(\sum_{j=1}^{m} q_{i,j}y_{i,j} - C_\epsilon\right)x_i. \end{aligned}$$

□

In our algorithm, we hire an extra γm workers and observe their skill levels $q_{i,j}$. We then solve the sampled LP, which consists of these γm workers. Let $\hat{\mathbf{x}}^*$ be the optimal task weights of the sampled LP. We then apply the Primal Approximation Algorithm with inputs $\hat{\mathbf{x}}^*$ and $q_{i,j}$ values of the primary m workers. We denote $\hat{\mathbf{y}}^*$ as the assignments generated by the Primal Approximation Algorithm using these two inputs.

By observing Algorithm 8, we can see that as long as the input x_i is non-negative for all i, z_j and $t_{i,j}$ will be non-negative and $1 - q_{i,j}x_i + z_j + t_{i,j} \geq 0$. Therefore, the Primal Approximation Algorithm always generates feasible dual solutions, and the dual objective $D(\mathbf{x}, \mathbf{z}, \mathbf{t})$ is no bigger than primal optimal by weak duality for any inputs $\mathbf{x}$ and $\mathbf{q}$, subject to $x_i \geq 0$ for all i.

Next we bound the values of $\hat{x}_i^*$ (in Lemma A.1.4) and $\Sigma_{j=1}^m q_{i,j}\hat{y}_{i,j}^* - C_\epsilon$ (in Lemma A.1.6). And then we can show the primal objective is close to optimal.

Lemma A.1.4. *For every $i \in \{1, \cdots, n\}$ and $j \in \{1, \cdots, \gamma m\}$, let $q_{i,j}^s$ be a value in $[0, 1]$ perturbed by random noise so that $q_{i,j}^s \neq q_{i',j'}^s$ unless $i = i'$ and $j = j'$. Let $\hat{\mathbf{x}}^*$ be the optimal task weights of the sampled LP with these perturbed values. Let $\hat{\mathbf{y}}^{s,*}$ denote the assignment generated by applying the Primal Approximation Algorithm with inputs $\hat{\mathbf{x}}^*$ and $\mathbf{q}^s$. Then under the perturbation assumption, for all i, $\hat{x}_i^* = 1/\min_{\{j:\hat{y}_{i,j}^{s,*}=1\}} q_{i,j}^s$.*

Proof. In this proof, our discussion is restricted to the sampled LP. Therefore, we omit all the superscript s in the notations.

Let j_i be the worker with minimum skill level for task i among the set of workers assigned to task i, i.e., $j_i = \text{argmin}_{j:\hat{y}_{i,j}^*=1} q_{i,j}$. Note that j_i is uniquely specified since all $q_{i,j}$ values are different.

Let $\bar{\mathbf{y}}^*$ be the optimal solution of the sampled LP. By the same argument as the one in the proof of Theorem 2.3.2, we know that for each worker j, there exists at most one task i such that $\bar{y}_{i,j}^* \neq \hat{y}_{i,j}^*$, which implies that for each worker j, there exists at most one task i such that $\bar{y}_{i,j}^* \notin \{0, 1\}$. Furthermore, we know that if such a task exists, the Primal Assignment Algorithm assigns the task to worker j, i.e., $\hat{y}_{i,j}^* = 1$.

We now show that if $\bar{y}_{i,j}^* \notin \{0, 1\}$, then $j = j_i$. This means for each task i, there exists at most one worker j such that $\bar{y}_{i,j}^* \notin \{0, 1\}$, and this worker is the one with minimum skill level among all the workers who are assigned to that task. Assume for contradiction that there exists a worker $j \neq j_i$ such that $\bar{y}_{i,j}^* \in (0, 1)$. We know that this worker is assigned to task i by the Primal Approximation Algorithm and so $q_{i,j} > q_{i,j_i}$. Let $\Delta = \min(q_{i,j_i}\bar{y}_{i,j_i}^*/q_{i,j}, 1 - \bar{y}_{i,j}^*)$. Suppose that we increased the value of $\bar{y}_{i,j}^*$ by Δ and decreased the value of $\bar{y}_{i,j_i}^*$ by $q_{i,j}\Delta/q_{i,j_i}$. It is easy to verify

that no constraints of the sampled LP would be violated, but the objective would increase since $q_{i,j} > q_{i,j_i}$. This is a contradiction.

Since the $q_{i,j}$ values have been randomly perturbed, there cannot exist any set of workers J_i for any task i such that $\Sigma_{j \in J_i} q_{i,j} = C_\epsilon$. Together with the last paragraph and the fact that $\Sigma_j q_{i,j} \bar{y}^*_{i,j} = C_\epsilon$, this implies that for each task i, $\bar{y}^*_{i,j_i} \notin \{0, 1\}$.

With this fact in place, we are ready to show that $\hat{x}^*_i = 1/q_{i,j_i}$ for all i. Let $(\hat{\mathbf{x}}^*, \hat{\mathbf{z}}^*, \hat{\mathbf{t}}^*)$ denote the dual optimal solution of the sampled LP.

First, assume there exists a task i such that $\hat{x}^*_i > 1/q_{i,j_i}$. Then we know that $1 - q_{i,j_i} \hat{x}^*_i < 0$. Recall that by the dual constraints, $1 - q_{i,j} \hat{x}^*_i + \hat{z}^*_j + \hat{t}^*_{i,j} \geq 0$ for all (i, j). Therefore, we have either $\hat{z}^*_{j_i} > 0$ or $\hat{t}^*_{i,j_i} > 0$. However, by complementary slackness, if $\hat{t}^*_{i,j_i} > 0$, we have $\bar{y}^*_{i,j_i} = 1$, which is a contradiction to the property above. On the other hand, if $\hat{z}^*_{j_i} > 0$, then $\Sigma_{i=1}^n \bar{y}^*_{i,j_i} = M_j$. Since M_j is an integer and there exist at most one task i such that $\bar{y}^*_{i,j_i} \notin \{0, 1\}$, we have $\bar{y}^*_{i,j_i} \in \{0, 1\}$. This is also a contradiction. Therefore, we must have that $\hat{x}^*_i \leq 1/q_{i,j_i}$.

Now assume there exists a task i such that $\hat{x}^*_i < 1/q_{i,j_i}$. Then $1 - q_{i,j_i} \hat{x}^*_i > 0$. Therefore $1 - q_{i,j_i} \hat{x}^*_i + \hat{z}^*_{j_i} + \hat{t}^*_{i,j_i} > 0$, which, by complementary slackness, means $\bar{y}^*_{i,j_i} = 0$. This is again a contradiction.

From the previous two paragraphs, we know that $\hat{x}^*_i = 1/q_{i,j_i}$ for all i. □

Next, we bound the value of $\Sigma_{j=1}^m q_{i,j} \hat{y}^*_{i,j} - C_\epsilon$. The proof relies on Lemma 3 of Devanur and Hayes [37], which we restate here for completeness.

Lemma A.1.5. *[37] Let $Y = (Y_1, ..., Y_m)$ be a vector of random variables. For any $\gamma \in (0, 1)$ such that γm is an integer, let S be a random subset of $\{1, \ldots, m\}$ of size γm, and $Y_S = \sum_{j \in S} Y_j$. Then under the perturbation assumption, for any $\delta \in (0, 1)$,*

$$Pr(|Y_S - E[Y_S]| \geq \frac{2}{3} \|Y\|_\infty \ln \frac{2}{\delta} + \|Y\|_2 \sqrt{2\gamma \ln \frac{2}{\delta}}) \leq \delta.$$

Lemma A.1.6. *For any given task i, with probability of at least $1 - \delta$,*

$$\sum_{j=1}^m q_{i,j} \hat{y}^*_{i,j} - C_\epsilon \leq \frac{d_{m_i}}{\gamma} + \frac{20}{\gamma} \ln \frac{2}{\delta} + 10 \sqrt{\frac{1}{\gamma} \ln \frac{2}{\delta}} \sqrt{C_\epsilon + \frac{1}{\gamma}}.$$

where $\Sigma_{i=1}^{n} d_{m_i} = \gamma m$. *Since* $\gamma \geq 1/C_\epsilon$,

$$\sum_{j=1}^{m} q_{i,j}\hat{y}_{i,j}^{*} - C_\epsilon \leq \frac{d_{m_i}}{\gamma} + 35 \ln \frac{2}{\delta}\sqrt{\frac{C_\epsilon}{\gamma}}.$$

Proof. We are considering the random permutation model. The adversary chooses values of capacities M_j and skills $p_{i,j}$ for $(1+\gamma)m$ workers, but the workers' arriving sequence is randomly permuted. In our algorithm, we hire the first γm workers and estimate the task weights. We then try to minimize the number of assignments for the latter m workers.

To simplify notations, in the proof, we use $\mathbf{q}^s$ to denote the skills of the first γm workers, and use $\mathbf{q}$ to denote the skill levels of primary m workers. We will use the following two facts to prove the lemma.

1. If $\hat{\mathbf{x}}^*$ is the optimal task weights of the sampled LP, let $\hat{\mathbf{y}}^{s,*}$ be the assignments generated by the Primal Approximation Algorithm with input $\hat{\mathbf{x}}^*$ and $\mathbf{q}^s$. We have $\gamma C_\epsilon \leq \Sigma_{j=1}^{\gamma m} q_{i,j}^s \hat{y}_{i,j}^{s,*} \leq \gamma C_\epsilon + 1$. This property follows directly from the proof of Lemma A.1.4. For each task i, there is at most one value $\hat{y}_{i,j}^{s,*}$ which is not equal to the optimal value $y_{i,j}^{s,*}$ of the sampled LP, and $\Sigma_{j=1}^{\gamma m} q_{i,j}^s y_{i,j}^{s,*} = C_\epsilon$.

   We also have $\Sigma_{j=1}^{\gamma m} q_{i,j}^s \hat{y}_{i,j}^{s,*} \leq \gamma C_\epsilon + d_{m_i}$, where $\Sigma_{i=1}^{n} d_{m_i} = \gamma m$. This property follows directly from the proof of Theorem 2.3.2. For worker j, there is at most one $\hat{y}_{i,j}^{s,*}$ which is not equal to the optimal solution of the sampled LP.

2. Given any task weight $\mathbf{x}$, let $\mathbf{y}^s$ be the assignments generated by the Primal Approximation Algorithm with inputs $\mathbf{x}$ and $\mathbf{q}^s$, and $\mathbf{y}$ be the assignments generated by the Primal Approximation Algorithm with input $\mathbf{x}$ and $\mathbf{q}$. Note that in our setting, given task weights $\mathbf{x}$ and worker skill levels, the assignments $y_{i,j}$ made by the Primal Approximation Algorithm are determined and remain the same for a given task, whether it appears in the initial sample or primary set of workers. Below we use this fact to show that, for task i, with high probability, $\Sigma_{j=1}^{\gamma m} q_{i,j}^s y_{i,j}^s$ is close to $\gamma \Sigma_{j=1}^{m} q_{i,j} y_{i,j}$.

   Consider a single task i. Define $Y_j = q_{i,j}^s y_{i,j}^s$ for $j \in \{1, ..., \gamma m\}$ and $Y_j = q_{i,j-\gamma m} y_{i,j-\gamma m}$ for $j \in \{\gamma m + 1, ..., (1+\gamma)m\}$, and let $Y = \{Y_1, ...Y_{(1+\gamma)m}\}$. Since we assume the worker

arriving sequence is randomly permuted, we can think of $\Sigma_{j=1}^{\gamma m} q_{i,j}^s y_{i,j}^s$ as the sum of a random subset of size γm from Y. We can then obtain a bound by applying Lemma A.1.5.

Before applying the lemma, we first bound the values of $\|Y\|_\infty$, $\|Y\|_2$, and $E[Y_S]$. Below we use the notation $Q_i = \Sigma_{j=1}^{m} q_{i,j} y_{i,j}$ and $Q_i^s = \Sigma_{j=1}^{\gamma m} q_{i,j}^s y_{i,j}^s$.

$$\begin{aligned}
\|Y\|_\infty &= \max_j \{Y_j\} \leq 1. \\
\|Y\|_2 &= \sqrt{\sum_{j=1}^{m} (q_{i,j} y_{i,j})^2 + \sum_{j=1}^{\gamma m} (q_{i,j}^s y_{i,j}^s)^2} \\
&\leq \sqrt{Q_i + Q_i^s}. \\
E[Y_S] &= \frac{\gamma}{(1+\gamma)} (Q_i + Q_i^s).
\end{aligned}$$

Applying Lemma A.1.5, with probability at least $1 - \delta$,

$$\begin{aligned}
&|Q_i^s - \frac{\gamma}{1+\gamma}(Q_i + Q_i^s)| \\
\leq &\frac{2}{3} \ln \frac{2}{\delta} + \sqrt{Q_i + Q_i^s} \sqrt{2 \frac{\gamma}{1+\gamma} \ln \frac{2}{\delta}}.
\end{aligned}$$

Finally, multiplying both sides by $(1+\gamma)$ gives us

$$\begin{aligned}
&|\gamma Q_i - Q_i^s| \\
\leq &(1+\gamma) \frac{2}{3} \ln \frac{2}{\delta} + \sqrt{Q_i + Q_i^s} \sqrt{2\gamma(1+\gamma) \ln \frac{2}{\delta}}.
\end{aligned}$$

Let $Q_i^* = \Sigma_{j=1}^{m} q_{i,j} \hat{y}_{i,j}^*$ and $Q_i^{s,*} = \Sigma_{j=1}^{\gamma m} q_{i,j}^s \hat{y}_{i,j}^{s,*}$. Using the two facts above, we have

$$Q_i^* \leq \frac{Q_i^{s,*}}{\gamma} + \frac{1+\gamma}{\gamma} \frac{2}{3} \ln \frac{2}{\delta} + \sqrt{\frac{2(1+\gamma)}{\gamma} \ln \frac{2}{\delta}} \sqrt{Q_i^* + Q_i^{s,*}}.$$

To simplify the notation, let $C = Q_i^{s,*}/\gamma$ and $K = 2\ln(2/\delta)(1+\gamma)/\gamma$, we can get

$$\begin{aligned}
Q_i^* &\leq C + K/3 + \sqrt{K} \sqrt{\gamma C + Q_i^*} \\
&\leq C + K/3 + \sqrt{K} \sqrt{\gamma C} + \sqrt{K} \sqrt{Q_i^*}.
\end{aligned}$$

By rearranging the variables, we can get

$$Q_i^* - \sqrt{K}\sqrt{Q_i^*} - (C + K/3 + \sqrt{K}\sqrt{\gamma C}) \leq 0.$$

Applying the quadratic equation, we know that

$$\begin{aligned}&\frac{\sqrt{K} - \sqrt{K + 4(C + K/3 + \sqrt{K}\sqrt{\gamma C})}}{2}\\ \leq&\sqrt{Q_i^*} \leq \frac{\sqrt{K} + \sqrt{K + 4(C + K/3 + \sqrt{K}\sqrt{\gamma C})}}{2}.\end{aligned}$$

Therefore,

$$\begin{aligned}\sqrt{Q_i^*} &\leq \frac{\sqrt{K} + \sqrt{K} + 2\sqrt{C + K/3 + \sqrt{K}\sqrt{\gamma C}}}{2}\\ &= \sqrt{K} + \sqrt{C + K/3 + \sqrt{K}\sqrt{\gamma C}}.\end{aligned}$$

Since $\gamma \leq 1$, taking the square on both sides,

$$\begin{aligned}Q_i^* &\leq K+C+K/3+\sqrt{K\gamma C}+2\sqrt{K}\sqrt{C+K/3+\sqrt{K\gamma C}}\\ &\leq C+4K/3+\sqrt{K\gamma C}+2\sqrt{K}\left(\sqrt{C}+\sqrt{K/3}+\sqrt{\sqrt{K\gamma C}}\right)\\ &\leq C + (\frac{4+2\sqrt{3}}{3})K + (\gamma^{\frac{1}{2}} + 2)K^{\frac{1}{2}}C^{\frac{1}{2}} + 2\gamma^{\frac{1}{4}}K^{\frac{3}{4}}C^{\frac{1}{4}}\\ &\leq C + 3K + 3K^{\frac{1}{2}}C^{\frac{1}{2}} + 2K^{\frac{3}{4}}C^{\frac{1}{4}}.\end{aligned}$$

Since $K^{3/4}C^{1/4} \leq \max\{K, K^{1/2}C^{1/2}\}$, we have $K^{3/4}C^{1/4} \leq K + K^{1/2}C^{1/2}$. Therefore,

$$Q_i^* \leq C + 5K + 5K^{\frac{1}{2}}C^{\frac{1}{2}}.$$

Since $Q_i^{s,*} \leq \gamma C_\epsilon + \min(1, d_{m_i})$, we have $C \leq C_\epsilon + \min(1/\gamma, d_{m_i}/\gamma)$. Also, since $1 + \gamma \leq 2$, we have

$$\begin{aligned}Q_i^* - C_\epsilon &\leq \frac{\min(1, d_{m_i})}{\gamma} + 10\frac{1+\gamma}{\gamma}\ln\frac{2}{\delta} + 5\sqrt{2\frac{1+\gamma}{\gamma}\ln\frac{2}{\delta}}\sqrt{C_\epsilon + \frac{1}{\gamma}}\\ &\leq \frac{\min(1, d_{m_i})}{\gamma} + \frac{20}{\gamma}\ln\frac{2}{\delta} + 10\sqrt{\frac{1}{\gamma}\ln\frac{2}{\delta}}\sqrt{C_\epsilon + \frac{1}{\gamma}}.\end{aligned}$$

Since $\delta < 1/2$, $\ln(2/\delta) \geq 1$. Since $\gamma \geq 1/C_\epsilon$, the above bound can be written as

$$\begin{aligned} Q_i^* - C_\epsilon &\leq \frac{\min(1, d_{m_i})}{\gamma} + (20 + 10\sqrt{2}) \ln \frac{2}{\delta} \sqrt{\frac{C_\epsilon}{\gamma}} \\ &\leq \frac{\min(1, d_{m_i})}{\gamma} + 35 \ln \frac{2}{\delta} \sqrt{\frac{C_\epsilon}{\gamma}}. \end{aligned}$$

□

Note that the Primal Approximation Algorithm always generates feasible dual solutions, therefore, the dual objective is always no bigger than the optimal primal solution by weak duality. We use OPT to denote the primal optimal solution of the relaxed offline formulation. Combing Lemmas A.1.3, A.1.6, and A.1.4. Recall that $q_{i,min} = \min_{\{j:\hat{y}_{i,j}^{s,*}=1\}} q_{i,j}^s$ and $q_{min} = \min_i q_{i,min}$. Since we required $\ln(2/\delta) \geq 1$ and $\gamma \geq 1/C_\epsilon$, we have that

$$\begin{aligned} P(\hat{\mathbf{y}}^*) &= D(\hat{\mathbf{x}}^*, \hat{\mathbf{z}}^*, \hat{\mathbf{t}}^*) + \sum_{i=1}^{n} \left(\sum_{j=1}^{m} q_{i,j} \hat{y}_{i,j}^* - C_\epsilon \right) \hat{x}_i \\ &\leq OPT + \sum_{i=1}^{n} \left(\frac{\min(1, d_{m_i})}{\gamma} + 35 \ln \frac{2}{\delta} \sqrt{\frac{C_\epsilon}{\gamma}} \right) \frac{1}{q_{i,min}} \\ &\leq OPT + \frac{1}{q_{min}} \left(\min(m, n) + 35n \ln \frac{2}{\delta} \sqrt{\frac{C_\epsilon}{\gamma}} \right). \end{aligned}$$

Since $OPT \geq nC_\epsilon$,

$$P(\hat{\mathbf{y}}^*) \leq OPT \left(1 + \frac{\min(m, n)}{q_{min} n C_\epsilon} + \frac{35 \ln(2/\delta)}{q_{min} \sqrt{\gamma C_\epsilon}} \right).$$

Finally, we need to bound the prediction error. Recall that we know that

$$C_\epsilon \leq \sum_{j=1}^{\gamma m} q_{i,j}^s \hat{y}_{i,j}^{s,*} \leq \gamma C_\epsilon + 1,$$

and

$$|\gamma Q_i^* - Q_i^s| \leq (1 + \gamma) \frac{2}{3} \ln \frac{2}{\delta} + \sqrt{Q_i^* + Q_i^s} \sqrt{2\gamma(1 + \gamma) \ln \frac{2}{\delta}}.$$

Hence,

$$\begin{aligned} Q_i^* &\geq C_\epsilon - \left(\frac{1+\gamma}{\gamma} \frac{2}{3} \ln \frac{2}{\delta} + \sqrt{\frac{2(1+\gamma)}{\gamma} \ln \frac{2}{\delta}} \sqrt{Q_i^* + \gamma(C_\epsilon + \frac{1}{\gamma})} \right) \\ &= C_\epsilon - K/3 - \sqrt{K} \sqrt{Q_i^*} - \sqrt{K} \sqrt{\gamma C} \end{aligned}$$

Again, by simple manipulation,

$$Q_i^* + \sqrt{K}\sqrt{Q_i^*} - C_\epsilon + K/3 + \sqrt{K}\sqrt{\gamma C} \geq 0.$$

Using the quadratic formula again,

$$\sqrt{Q_i^*} \leq \frac{-\sqrt{K} - \sqrt{K + 4C_\epsilon - 4K/3 - 4\sqrt{K}\sqrt{\gamma C}}}{2}$$

or

$$\sqrt{Q_i^*} \geq \frac{-\sqrt{K} + \sqrt{K + 4C_\epsilon - 4K/3 - 4\sqrt{K}\sqrt{\gamma C}}}{2}.$$

Note that the first inequality never holds since $\sqrt{Q_i^*} \geq 0$. Squaring both sides of the second inequality,

$$\begin{aligned}
Q_i^* &\geq \frac{K}{4} + \frac{K}{4} + C_\epsilon - \frac{K}{3} - \sqrt{K}\sqrt{\gamma C} \\
&\quad - \frac{1}{2}\sqrt{K}\sqrt{K + 4C_\epsilon - 4K/3 - 4\sqrt{K}\sqrt{\gamma C}} \\
&\geq C_\epsilon + \frac{K}{6} - \sqrt{\gamma C K} \\
&\quad - \frac{1}{2}\sqrt{K}(\sqrt{K} + 2\sqrt{C_\epsilon} - 2\sqrt{K/3} - 2\sqrt{\sqrt{K}\sqrt{\gamma C}}) \\
&\geq C_\epsilon + 0.24K - (\gamma^{1/2} + 1)(CK)^{1/2} + (\gamma C)^{1/4} K^{3/4} \\
&\geq C_\epsilon - 2K^{\frac{1}{2}} C^{\frac{1}{2}} \\
&= C_\epsilon - 2\sqrt{2\frac{1+\gamma}{\gamma}\ln\frac{2}{\delta}}\sqrt{C_\epsilon + \frac{1}{\gamma}} \\
&\geq C_\epsilon - 4\sqrt{\frac{1}{\gamma}\ln\frac{2}{\delta}}\sqrt{C_\epsilon + \frac{1}{\gamma}}.
\end{aligned}$$

Again, since $\delta < 1/2$ and $\gamma \geq 1/C_\epsilon$,

$$\begin{aligned}
Q_i^* &\geq C_\epsilon - 4\sqrt{2}\sqrt{\frac{1}{\gamma}\ln\frac{2}{\delta}}\sqrt{C_\epsilon} \\
&= C_\epsilon(1 - 4\sqrt{2}\sqrt{\frac{1}{\gamma}\ln\frac{2}{\delta}}\frac{1}{\sqrt{C_\epsilon}}) \\
&\geq C_\epsilon(1 - 6\ln\frac{2}{\delta}\frac{1}{\sqrt{\gamma C_\epsilon}}).
\end{aligned} \tag{A.7}$$

Therefore, if labels are aggregated using weighted majority voting, the prediction accuracy is bounded by ϵ', where

$$\epsilon' = \epsilon^{1-6\ln(2/\delta)/\sqrt{\gamma C_\epsilon}}$$

because $C_{\epsilon'} = C_\epsilon(1 - 6\ln(2/\delta)/\sqrt{\gamma C_\epsilon})$. □

A.1.5 Proof of Lemma 2.4.3

We have assumed we have an estimate $\hat{p}_{i,j}$ such that $|p_{i,j} - \hat{p}_{i,j}| = \alpha$ for some $\alpha \in [0, t]$. Define $\hat{q}_{i,j} = (2\hat{p}_{i,j} - 1)^2$. We have

$$\begin{aligned} |q_{i,j} - \hat{q}_{i,j}| &= \left|(2p_{i,j} - 1)^2 - (2\hat{p}_{i,j} - 1)^2\right| \\ &= 4\left|p_{i,j}^2 - p_{i,j} - \hat{p}_{i,j}^2 + \hat{p}_{i,j}\right| \\ &= 4\left|(p_{i,j}^2 - \hat{p}_{i,j}^2) - (p_{i,j} - \hat{p}_{i,j})\right| \ . \end{aligned} \tag{A.8}$$

Consider the two terms in the absolute value in Equation A.8. First note that these two terms always have the same sign; both are positive if $p_{i,j} > \hat{p}_{i,j}$, negative if $p_{i,j} < \hat{p}_{i,j}$, and 0 if $p_{i,j} = \hat{p}_{i,j}$. Therefore, we can write

$$\begin{aligned} |q_{i,j} - \hat{q}_{i,j}| &= 4\left|\left|p_{i,j}^2 - \hat{p}_{i,j}^2\right| - |p_{i,j} - \hat{p}_{i,j}|\right| \\ &= 4\left|\left|p_{i,j}^2 - \hat{p}_{i,j}^2\right| - \alpha\right| . \end{aligned}$$

Consider the case in which $p_{i,j} \geq \hat{p}_{i,j}$. Then

$$\begin{aligned} \left|p_{i,j}^2 - \hat{p}_{i,j}^2\right| &= p_{i,j}^2 - \hat{p}_{i,j}^2 \leq p_{i,j}^2 - (p_{i,j} - \alpha)^2 \\ &= 2\alpha p_{i,j} - \alpha^2 \leq 2\alpha \ . \end{aligned}$$

A symmetric argument can be made for the case in which $p_{i,j} < \hat{p}_{i,j}$. So $\left|p_{i,j}^2 - \hat{p}_{i,j}^2\right| \in [0, 2\alpha]$ and therefore

$$|q_{i,j} - \hat{q}_{i,j}| \leq 4\alpha \leq 4t.$$

□

A.1.6 Proof of Theorem 2.4.4

We first show that the optimal solution of the relaxed offline formulation with the true $q_{i,j}$ values and parameter ϵ is feasible in the approximated LP using the values $\hat{q}_{i,j} = (2\hat{p}_{i,j} - 1)^2$ and parameter ϵ'. Let $\bar{\mathbf{y}}^*$ be the optimal solution of the relaxed offline formulation. To show that $\bar{\mathbf{y}}^*$ is feasible in the approximated LP, we need only show that for every task i, $\bar{\mathbf{y}}^*$ satisfies

$$\sum_{j=1}^{m} \hat{q}_{i,j}\bar{y}^*_{i,j} \geq C_{\epsilon'} = 2\ln\left(\frac{1}{\epsilon^{1-4t/\bar{q}^*_{min}}}\right), \tag{A.9}$$

because $\hat{\mathbf{q}}$ and $C_{\epsilon'}$ only appear in this constraint.

Fix a task i. Since we assumed that for every j, $|p_{i,j} - \hat{p}_{i,j}| \leq t$, we have from Lemma 2.4.3 that $\hat{q}_{i,j} \geq q_{i,j} - 4t$ for all j, and

$$\sum_{j=1}^{m} \hat{q}_{i,j}\bar{y}^*_{i,j} \geq \sum_{j=1}^{m} q_{i,j}\bar{y}^*_{i,j} - 4t\sum_{j=1}^{m} \bar{y}^*_{i,j}.$$

Since $\bar{\mathbf{y}}^*$ is optimal in the relaxed offline formulation we can easily show that $\Sigma_{j=1}^{m} q_{i,j}\bar{y}^*_{i,j} = C_\epsilon$ for all i. Assume for contradiction that this was not the case. We cannot have $\Sigma_{j=1}^{m} q_{i,j}\bar{y}^*_{i,j} < C_\epsilon$ for any i since this would violate the constraints of the relaxed offline formulation. Assume that $\Sigma_{j=1}^{m} q_{i,j}\bar{y}^*_{i,j} > C_\epsilon$ for some i. Then we could always decrease some $y^*_{i,j}$ of task i to make the equality hold, which would decrease the objective value without violating any other constraint, meaning that the solution could not be optimal.

By definition of $\bar{q}^*_i$, i.e., $\bar{q}^*_i = \Sigma_{j=1}^{m} q_{i,j}\bar{y}^*_{i,j} / \Sigma_{j=1}^{m} \bar{y}^*_{i,j}$, we also can get that $\Sigma_{j=1}^{m} \bar{y}^*_{i,j} = C_\epsilon / \bar{q}^*_i$.

Therefore,

$$\sum_{j=1}^{m} \hat{q}_{i,j}\bar{y}^*_{i,j} \geq C_\epsilon - 4tC_\epsilon/\bar{q}^*_i = C_\epsilon(1 - 4t/\bar{q}^*_i).$$

Since $C_\epsilon = 2\ln(1/\epsilon)$,

$$\begin{aligned} C_\epsilon(1 - 4t/\bar{q}^*_i) &= 2(1 - 4t/\bar{q}^*_i)\ln(1/\epsilon) \\ &= 2\ln\left(\frac{1}{\epsilon^{1-4t/\bar{q}^*_i}}\right). \end{aligned}$$

Hence,

$$\sum_{j=1}^{m} \hat{q}_{i,j}\bar{y}^*_{i,j} \geq 2\ln\left(\frac{1}{\epsilon^{1-4t/\bar{q}^*_i}}\right) \geq 2\ln\left(\frac{1}{\epsilon^{1-4t/\bar{q}^*_{min}}}\right).$$

which shows that $\bar{\mathbf{y}}^*$ satisfies Equation A.9 for all i.

Since the optimal solution of the relaxed offline formulation is feasible in the approximated LP, the optimal objective of the approximated LP is no bigger than the optimal objective of the relaxed offline formulation. Furthermore, the optimal objective of the relaxed offline formulation is always no bigger than the optimal objective of the offline integer program. Therefore, the optimal objective of the approximated LP would be no bigger than the offline optimal of integer program. □

A.1.7 Proof of Theorem 2.4.5

We begin by restating the main definitions from the theorem statement for easy reference. Recall that $\bar{q}^*_{min}$ is a value such that $\bar{q}^*_{min} \leq \bar{q}^*_i$ for all i, and our estimates $\hat{p}_{i,j}$ are guaranteed to satisfy $|p_{i,j} - \hat{p}_{i,j}| \leq t$ for all (i,j) pairs for some $t < \bar{q}^*_{min}/4$. Recall that $\mathbf{y}$ is any feasible integer assignment of the approximated LP with parameter $\epsilon' = \epsilon^{1-4t/\bar{q}^*_{min}}$ and skill levels $\hat{q}_{i,j} = (2\hat{p}_{i,j}-1)^2$, $J_i = \{j : y_{i,j} = 1\}$ denotes the set of workers that are assigned to task i according to the feasible integer assignment, and $\hat{q}_i = \Sigma_{j \in J_i} \hat{q}_{i,j} / |J_i|$. Tasks are assigned according to $\mathbf{y}$ and the results are aggregated using weighted majority voting with weights $w_{i,j} = 2\hat{p}_{i,j} - 1$, so

$$\hat{\ell}_i = \text{sign}\left(\sum_{j \in J_i} (2\hat{p}_{i,j} - 1)\ell_{i,j}\right).$$

Below we show that $\hat{\ell}_i = \ell_i$ with probability at least $1 - \epsilon^{(1-4t/\bar{q}^*_{min})(1-4t/\hat{q}_{min})}$.

Without loss of generality, assume that $\ell_i = 1$. Note that although we weight each label by $2\hat{p}_{i,j} - 1$, the probability of label $\ell_{i,j}$ being correct (i.e., the probability that $\ell_{i,j} = \ell_i = 1$) is still $p_{i,j}$. Similar to the proof of Lemma 2.3.1, let

$$X_{i,j} = \begin{cases} 2\hat{p}_{i,j} - 1 & \text{with probability } p_{i,j} \\ -2\hat{p}_{i,j} + 1 & \text{with probability } 1 - p_{i,j}, \end{cases}$$

and

$$X_i = \sum_{j \in J_i} X_{i,j}.$$

Let $t_{i,j} = p_{i,j} - \hat{p}_{i,j}$. Then

$$\begin{aligned}
E[X_i] &= \sum_{j \in J_i} E[X_{i,j}] \\
&= \sum_{j \in J_i} (p_{i,j}(2\hat{p}_{i,j} - 1) + (1 - p_{i,j})(-2\hat{p}_{i,j} + 1)) \\
&= \sum_{j \in J_i} (4p_{i,j}\hat{p}_{i,j} - 2p_{i,j} - 2\hat{p}_{i,j} + 1) \\
&= \sum_{j \in J_i} (4(\hat{p}_{i,j} + t_{i,j})\hat{p}_{i,j} - 2(\hat{p}_{i,j} + t_{i,j}) - 2\hat{p}_{i,j} + 1) \\
&= \sum_{j \in J_i} (2\hat{p}_{i,j} - 1)^2 + \sum_{j \in J_i} 2t_{i,j}(2\hat{p}_{i,j} - 1) \\
&\geq \sum_{j \in J_i} (2\hat{p}_{i,j} - 1)^2 - \sum_{j \in J_i} 2\left|t_{i,j}\right| \left|2\hat{p}_{i,j} - 1\right|.
\end{aligned}$$

The first term is simply $|J_i|\,\hat{q}_i$. Since, by assumption, $|t_{i,j}| \leq t$ for all (i,j), and since $|2\hat{p}_{i,j} - 1| \leq 1$ for all (i,j), we have

$$E[X_i] \geq |J_i|\,\hat{q}_i - 2t\,|J_i| = |J_i|\,(\hat{q}_i - 2t).$$

Similar to the proof of Lemma 2.3.1, the probability of predicting the wrong label for task i is bounded by $\Pr(X_i \leq 0) = \Pr(E[X_i] - X_i \geq E[X_i])$. Using Hoeffding's inequality and the bound

above, we have

$$\begin{aligned}
Pr(X_i \leq 0) &\leq \exp\left(-\frac{2(E[X_i])^2}{\sum_{j\in J_i}(4\hat{p}_{i,j}-2)^2}\right) \\
&\leq \exp\left(-\frac{2|J_i|^2(\hat{q}_i-2t)^2}{4|J_i|\hat{q}_i}\right) \\
&= \exp\left(-\frac{|J_i|(\hat{q}_i-2t)^2}{2\hat{q}_i}\right) \\
&= \exp\left(-\frac{|J_i|\hat{q}_i{}^2(1-2t/\hat{q}_i)^2}{2\hat{q}_i}\right) \\
&= \exp\left(-\frac{1}{2}|J_i|\hat{q}_i(1-2t/\hat{q}_i)^2\right) \\
&\leq \left(\epsilon^{1-4t/\bar{q}^*_{min}}\right)^{(1-2t/\hat{q}_i)^2} \\
&= \epsilon^{(1-4t/\bar{q}^*_{min})(1-2t/\hat{q}_i)^2} \\
&\leq \epsilon^{(1-4t/\bar{q}^*_{min})(1-4t/\hat{q}_i)}.
\end{aligned}$$

The sixth line follows from the constraints of the approximated LP which require that

$$|J_i|\hat{q}_i = \sum_{j\in J_i}\hat{q}_{i,j} \geq 2\ln(1/\epsilon'),$$

and so $\exp(-|J_i|\hat{q}_i/2) \leq \epsilon'$. □

A.1.8 Proof of Theorem 2.4.6

The proof is a direct application of Lemma 2.4.2 and Theorems 2.4.1, 2.4.4, and 2.4.5.

We assign each worker s gold standard tasks for each task type. If we use the empirical skill level $\hat{p}_{i,j}$ as an estimate of $p_{i,j}$ as in Lemma 2.4.2, we know that for any δ', with probability at least $1-\delta'$, $|p_{i,j}-\hat{p}_{i,j}| \leq \sqrt{\ln(2/\delta')/(2s)}$.

Note that we have T types of tasks and $(1+\gamma)m$ workers. Hence, we need to assign $T(1+\gamma)ms$ gold standard tasks in total to workers. Replacing δ' with $\delta/2$ and applying the union bound for all types of tasks and all workers, we know that with probability at least $1-\delta/2$,

$$|p_{i,j}-\hat{p}_{i,j}| \leq \sqrt{\frac{\ln(4T(1+\gamma)m/\delta)}{2s}} \text{ for all } (i,j). \tag{A.10}$$

Let $t = \sqrt{\ln(4T(1+\gamma)m/\delta)/(2s)}$. If we set $\hat{q}_{i,j} = (2\hat{p}_{i,j} - 1)^2$, from Lemma 2.4.3, we have an estimate $\hat{\mathbf{q}}$ such that with probability at least $1 - \delta/2$, $|q_{i,j} - \hat{q}_{i,j}| \leq 4t$ for all i and j. Given $\hat{\mathbf{q}}$ and access to $\bar{q}^*_{min}$ (as assumed in statement of the theorem), we can create the approximated LP as described in Section 2.4.2 with $\hat{\mathbf{q}}$ and $C_{\epsilon'}$ where $\epsilon' = \epsilon^{1-4t/\bar{q}^*_{min}}$.

Bounding Competitive Ratio

Below we bound the number of non-gold standard tasks assignment of the the full version. In Section 5.1, we assumed that the $q_{i,j}$ values of worker j are revealed when worker j arrives. However, now we have only estimates of these values. We can still apply Theorem 2.4.1, but now with the estimated values $\hat{q}_{i,j}$ in place of the $q_{i,j}$, and ϵ' in place of ϵ. The relaxed online formulation with these replacements is exactly the approximated LP of Section 2.4.2. Applying Theorem 2.4.1 with δ set to $\delta/2$, with probability at least $1 - \delta/2$, the Primal Approximation Algorithm with inputs $\hat{\mathbf{x}}$ and $\hat{\mathbf{q}}$ yields an assignment $\mathbf{y}$ such that the number of tasks assigned using $\mathbf{y}$ is no more than

$$\left(1 + \frac{\min(m,n)}{\hat{q}_{min} n C_{\epsilon'}} + \frac{35\ln(4/\delta)}{\hat{q}_{min}\sqrt{\gamma C_{\epsilon'}}}\right)$$

times the optimal value of the approximated LP, where $\hat{q}_{min} = \min_{(i,j):y^s_{i,j}=1} \hat{q}^s_{i,j}$, and $\mathbf{y}^s$ is the primal optimal solution of the sampled LP with the replacements described above.

From Theorem 2.4.4, we know that the optimal of the approximated LP yields no more assignment than the optimal of the relaxed offline formulation. Combining the above results, with probability of at least $1 - \delta/2$, the the full version yields no more assignments than

$$\left(1 + \frac{\min(m,n)}{\hat{q}_{min} n C_{\epsilon'}} + \frac{35\ln(4/\delta)}{\hat{q}_{min}\sqrt{\gamma C_{\epsilon'}}}\right)$$

times the optimal of the relaxed offline formulation, which is less than the optimal of the IP.

Since the analysis above holds only if $|p_{i,j} - \hat{p}_{i,j}| \leq t$ for all (i,j) and we know this happens with probability at least $1 - \delta/2$, we can apply union bound to show that the competitive ratio above can be achieved with probability of at least $1 - \delta$.

Bounding Prediction Error

We now bound the prediction error. By a similar argument to the one used above, we can use the same argument made to achieve the error bound in Theorem 2.4.1 (Equation A.7) to guarantee that

if the high probability events above hold, then

$$\sum_{j=1}^{m} \hat{q}_{i,j} y_{i,j} \geq C_{\epsilon'} \left(1 - 6\frac{\ln(4/\delta)}{\sqrt{\gamma C'_{\epsilon}}}\right)$$

where $\epsilon'' = \epsilon'^{(1-6\ln(4/\delta)/\sqrt{\gamma C_{\epsilon'}})}$.

Assume that the high probability event from above holds and so $|p_{i,j} - \hat{p}_{i,j}| \leq t$ for all (i, j). Let $\hat{q}_i = \sum_{j:y_{i,j}=1} q_{i,j}/|\{j : y_{i,j} = 1\}|$. Applying Theorem 2.4.5 with the value ϵ'' in place of ϵ' (since $\mathbf{y}$ is a feasible assignment to the relaxed LP with ϵ'' used instead of ϵ'), we have that the prediction error of each task i is bounded by

$$\begin{aligned}
\epsilon''^{1-4t/\hat{q}_i} &\leq \epsilon'^{(1-6\ln(4/\delta)/\sqrt{\gamma C'_{\epsilon}})(1-4t/\hat{q}_i)} \\
&\leq \epsilon^{(1-4t/\bar{q}^*_{min})(1-6\ln(4/\delta)/\sqrt{\gamma C'_{\epsilon}})(1-4t/\hat{q}_i)}.
\end{aligned}$$

Replacing s in Equation A.10, we have that $|p_{i,j} - \hat{p}_{i,j}| \leq t$ for all i and j where $t \leq \sqrt{\ln(4T(1+\gamma)m/\delta)/(2s)}$. Finally, replacing t in the error bound above completes the proof. □

A.2 Additional Proofs from Chapter 3

A.2.1 Proof of Lemma 3.3.2

A.2.1.1 Notations

Before the proof, we first define and describe some notations. We consider the case when *all users follow the social norm.* Below we show the derivation on the worker side, the notations and derivations are identical for the requesters.

- $u_w(\theta_w, \theta_r)$ is the worker stage payoff when a worker with reputation θ_w is matched with a requester with reputation θ_r. The values can be looked up in the payoff matrix in Table 3.1. For example, $u_w(\theta_w, \theta_r) = P - C$ if $\sigma_w(\theta_w, \theta_r) =$"High Effort" and $\sigma_r(\theta_w, \theta_r) =$"Pay".
- $v_w(\theta)$ is the expected stage payoff of a worker with reputation θ before she is matched. Since we are considering random matching, the expected stage payoff can be defined using

stationary reputation distribution $\{\eta_w(\theta)\}$

$$v_w(\theta) = \sum_{\tilde{\theta}} u_w(\theta, \tilde{\theta}) \eta_w(\tilde{\theta})$$

- $v_w^\infty(\theta)$ is the expected long-term payoff of a worker with reputation θ before she is matched. Let $p_w(\theta'|\theta)$ be the transition probability for a worker with reputation θ to become θ'. The long-term payoff $v_w^\infty(\theta)$ of a worker with reputation θ can be written as (δ is the discounting factor)

$$v_w^\infty(\theta) = v_w(\theta) + \delta \sum_{\theta' \in \Theta} p_w(\theta'|\theta) v_w^\infty(\theta')$$

Given the above definitions, we can show that $v_w^\infty(\theta)$ and $v_r^\infty(\theta)$ are non-decreasing in θ.

Proposition A.2.1. *$v_w^\infty(\theta)$ and $v_r^\infty(\theta)$ are non-decreasing in θ.*

Proof. Since we are considering the threshold-based social norm with threshold values (k_w, k_r), we know that a worker with reputation $\theta \geq k_w$ will always get paid, and she only exerts high effort to requesters with reputation more than k_r. Using this argument for $\theta < k_w$, we can get the expected stage payoff as follows,

$$v_w(\theta) = \begin{cases} P - C \cdot \sum_{\tilde{\theta}=k_r}^{L} \eta(\tilde{\theta}) & \text{if } \theta \geq k_w \\ -C \cdot \sum_{\tilde{\theta}=k_r}^{L} \eta(\tilde{\theta}) & \text{otherwise} \end{cases}$$

By observing the formulations, it is trivial to see that $v_w(\theta)$ is non-decreasing in θ. Now we write down the definition of v_w^∞ as follows:

$$v_w^\infty(L) = v_w(L) + \delta\epsilon_w v_w^\infty(0) + \delta(1-\epsilon_w) v_w^\infty(L)$$
$$v_w^\infty(\theta) = v_w(\theta) + \delta\epsilon_w v_w^\infty(0) + \delta(1-\epsilon_w) v_w^\infty(\theta+1) \; \forall \theta < L$$

According to the definition, we know that

$$v_w^\infty(L) - v_w^\infty(L-1) = v_w(L) - v_w(L-1) \geq 0$$

Therefore, $v_w^\infty(L) \geq v_w^\infty(L-1)$. For $\theta \neq L$, we know that

$$v_w^\infty(\theta) - v_w^\infty(\theta-1) = v_w(\theta) - v_w(\theta-1) + \delta(1-\epsilon)(v_w^\infty(\theta+1) - v_w^\infty(\theta))$$

Since we know $v_w(\theta) \geq v_w(\theta-1)$ and $v_w^\infty(L) \geq v_w^\infty(L-1)$, we can show that $v_w^\infty(\theta) \geq v_w(\theta-1)$ is valid for all θ by simple induction. Therefore, $v_w^\infty(\theta)$ is non-decreasing in θ. Exactly the same proof can be used in requester side. □

A.2.1.2 Proof of the lemma

In this proof, we check if we can find any worker who can get better expected payoff by deviating from the social norm. According to one-shot deviation principle in game theory, we know that we need only check if a worker can get better payoff by deviating in one stage (and follows the social norm in the other stages.) Below we check all possible cases in which a worker has different reputation values. Therefore, we cover all possible situations in the future rounds, and we need only check if a worker can get better payoff by deviating in the current stage. Since we assume the user population is large, the deviation of single worker will not influence the overall distribution. Therefore, we can calculate the long-term expected payoff using the stationary reputation distribution. As before, we only focus on the proof of the worker side. The same proof can be used on the requester side.

In the discussion, we divide the worker payoff into two parts: the current payoff, i.e. the payoff in the current stage, and the expected future payoff. We calculate the payoff differences between the following two worker strategies: 1) always follow the social norm, and 2) deviate from the social norm in the current stage and follow the social norm afterwords. Consider the case when the worker with reputation θ_w is matched with the requester with reputation θ_r. Let $\Delta V_{w,CUR}(\theta_w, \theta_r)$ be the difference of the current payoff changing from the first strategy to the second strategy, and $\Delta V_{w,FUT}(\theta_w, \theta_r)$ be the difference of the expected future payoff.

If the social norm is sustainable, it means no worker has incentive to change from the first strategy to second strategy. Therefore, we can write the sustainable conditions as $\Delta V_{w,CUR}(\theta_w, \theta_r) + \Delta V_{w,FUT}(\theta_w, \theta_r) \leq 0$ for all (θ_w, θ_r). In the following discussion, we find the necessary and sufficient conditions where the above inequalities are satisfied. Below we show how to calculate $\Delta V_{w,CUR}(\theta_w, \theta_r)$ and $\Delta V_{w,FUT}(\theta_w, \theta_r)$.

- $\Delta V_{w,FUT}(\theta_w, \theta_r)$

If a worker with reputation θ_w follows the social norm in the current stage, since we consider reporting errors, her reputation would become $\min\{\theta_w + 1, L\}$ with probability $1 - \epsilon$ and become 0 with probability ϵ. Therefore, her expected future payoff would be

$$\delta[(1 - \epsilon_w)v_w^\infty(\min\{\theta_w + 1, L\}) + \epsilon_w v_w^\infty(0)]$$

If a worker chooses to deviate from the social norm in the current stage, her expected future payoff would be

$$\delta[(1 - \epsilon_w)v_w^\infty(0) + \epsilon_w v_w^\infty(\min\{\theta_w + 1, L\})]$$

Let $\theta^+ = \min\{\theta + 1, L\}$, we can get the results as follows

$$\Delta V_{w,FUT}(\theta_w, \theta_r) = -\delta(1 - 2\epsilon_w)[v_w^\infty(\theta_w^+) - v_w^\infty(0)]$$

- $\Delta V_{w,CUR}(\theta_w, \theta_r)$

  Since we consider the case when all users follow the social norm, the difference of the current payoff can be expressed as follows

$$\Delta V_{w,CUR}(\theta_w, \theta_r) = \begin{cases} C & \text{if } \theta_r \geq k_r \\ -C & \text{otherwise} \end{cases}$$

A social norm is sustainable for workers if and only if $\Delta V_{w,CUR}(\theta_w, \theta_r) + \Delta V_{w,FUT}(\theta_w, \theta_r) \leq 0$ for all (θ_w, θ_r). In the following analysis, we consider the worst set of (θ_w, θ_r) where both $\Delta V_{w,CUR}$ and $\Delta V_{w,FUT}$ are maximum. The worst condition is satisfied if only if the general conditions are satisfied. Since we know that $v_w^\infty(\theta)$ is non-decreasing in θ from Proposition A.2.1, setting θ_w to 0 maximizes the value of $\Delta V_{w,FUT}$. Since we are not considering the trivial case when no workers are required to exert high effort, i.e., $k_r < L$, setting θ_r to L maximize $\Delta V_{w,CUR}$. Therefore, the sustainable conditions are satisfied if and only if

$$\Delta V_{w,CUR}(0, L) + \Delta V_{w,FUT}(0, L) \leq 0$$

Intuitively speaking, when a worker with reputation 0 has no incentive to deviate from the social norm when she is matched with a requester with reputation L, no worker would has incentive to deviate from the social norm. Therefore, the social norm is sustainable for workers.

Based on the above formulas, we can derive the sustainable conditions as in the lemma. The detailed derivations are list as follows. We first insert the value of $\Delta V_{w,CUR}$ and $\Delta V_{r,FUT}$ into the formulas.

$$C - \delta(1 - 2\epsilon_w)[v_w^\infty(1) - v_w^\infty(0)] \leq 0$$

Given the definition of $v_w^\infty(\theta)$

$$v_w^\infty(\theta) = v_w(\theta) + \delta\epsilon_w v_w^\infty(0) + \delta(1 - \epsilon_w)v_w^\infty(\min\{\theta + 1, L\})$$

we can calculate the result of $v_w^\infty(1) - v_w^\infty(0)$

$$\begin{aligned} v_w^\infty(1) - v_w^\infty(0) &= v_w(1) - v_w(0) + \delta(1 - \epsilon_w)(v_w^\infty(2) - v_w^\infty(1)) \\ &= \sum_{i=1}^{L} \delta^{i-1}(1 - \epsilon_w)^{i-1}(v_w(i) - v_w(i-1)) \end{aligned}$$

The definition of $v_w(\theta)$ is as follows,

$$v_w(\theta) = \begin{cases} P - C \cdot \sum_{\tilde{\theta}=k_r}^{L} \eta(\tilde{\theta}) & \text{if } \theta \geq k_w \\ -C \cdot \sum_{\tilde{\theta}=k_r}^{L} \eta(\tilde{\theta}) & \text{otherwise} \end{cases}$$

Note that $v_w(\theta)$ is a step function. $v_w(\theta) - v_w(\theta - 1) = 0$ if θ is not k_w, and $v_w(\theta) - v_w(\theta - 1) = P$ if $\theta = k_w$. Therefore, we can rewrite $v_w^\infty(1) - v_w^\infty(0)$ as follows,

$$v_w^\infty(1) - v_w^\infty(0) = \begin{cases} \delta^{k_w - 1}(1 - \epsilon_w)^{k_w - 1}P & \text{if } k_w > 0 \\ 0 & \text{otherwise} \end{cases}$$

If $k_w = 0$, the sustainable condition would never be satisfied. Since $v_w^\infty(1) - v_w^\infty(0) = 0$ and $C > 0$, the sustainable condition would become

$$C - \delta(1 - 2\epsilon_w)0 \leq 0$$

which is never satisfied. Using the same argument in requester side, we know that k_r and k_w must be larger than 0 for a social norm to be sustainable. Since $k_r, k_w > 0$, we can substitute the values to the formulation and get

$$\delta(1 - 2\epsilon_w)[\delta(1 - \epsilon_w)]^{k_w - 1}P \geq C$$

$$\delta(1 - 2\epsilon_r)[\delta(1 - \epsilon_r)]^{k_r - 1}V \geq P$$

which is equivalent to the conditions in the lemma.

A.2.2 Proof of Theorem 3.4.1

In the setting of dynamic population with turnover rates α_w and α_r, the objective value is still non-increasing in k_w and constant in k_r. The sustainable conditions in Lemma 3.3.2 will be modified as follow

$$\begin{aligned}
&\frac{1}{\delta^{k_w}(1-\alpha_w)^{k_w}(1-\epsilon_w)^{k_w-1}(1-2\epsilon_w)}\ C \\
&\leq P \\
&\leq \delta^{k_r}(1-\alpha_r)^{k_r}(1-\epsilon_r)^{k_r-1}(1-2\epsilon_r)\ V
\end{aligned}$$

Using the same argument in Theorem 3.3.3, we can set k_w and k_r to 1 and get the sustainable conditions in the optimal setting. This completes the proof.

Below we show how to derive the modified sustainable conditions as above. Similar to the proof in Lemma 3.3.2, we need only consider the following conditions,

$$\begin{aligned}
\Delta V_{w,CUR}(0,L) + \Delta V_{w,FUT}(0,L) \leq 0 \\
\Delta V_{r,CUR}(L,0) + \Delta V_{r,FUT}(L,0) \leq 0
\end{aligned}$$

Since the difference of the current payoff ΔV_{CUR} only depends on the social strategies, it stays the same as in Lemma 3.3.2. Below we calculate ΔV_{FUT}. With turnover rate α, each user needs to consider if she will stay in the market in the future. Therefore, she will discount the future payoff by $(1-\alpha)$ as follows,

$$\begin{aligned}
\Delta V_{w,FUT}(0,L) &= -\delta(1-\alpha_w)(1-2\epsilon_w)[v_w^\infty(1) - v_w^\infty(0)] \\
\Delta V_{r,FUT}(L,0) &= -\delta(1-\alpha_r)(1-2\epsilon_r)[v_r^\infty(1) - v_r^\infty(0)]
\end{aligned}$$

With turnover rate α, the definition of $v^\infty(\theta)$ We can calculate $v^\infty(1) - v^\infty(0)$ as follows,

$$v_w^\infty(1) - v_w^\infty(0) = \begin{cases} [\delta(1-\alpha_w)(1-\epsilon_w)]^{k_w-1}P & \text{if } k_w > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$v_r^\infty(1) - v_r^\infty(0) = \begin{cases} [\delta(1-\alpha_r)(1-\epsilon_r)]^{k_r-1}V & \text{if } k_r > 0 \\ 0 & \text{otherwise} \end{cases}$$

Combining the above results, we can get the following conditions and therefore complete the proof.

$$\delta(1-2\epsilon_w)(1-\alpha_w)[\delta(1-\alpha_w)(1-\epsilon_w)]^{k_w-1}P \geq C$$
$$\delta(1-2\epsilon_r)(1-\alpha_r)[\delta(1-\alpha_r)(1-\epsilon_r)]^{k_r-1}V \geq P$$

A.2.3 Proof of Theorem 3.4.2

Similar to the proof in Theorem 3.4.1, the sustainable conditions can be expressed as follows,

$$\Delta V_{w,CUR}(0,L) + \Delta V_{w,FUT}(0,L) \leq 0$$
$$\Delta V_{r,CUR}(L,0) + \Delta V_{r,FUT}(L,0) \leq 0$$

The differences of the current payoff ΔV_{CUR} are the same as in Lemma 3.3.2. The differences of the expected future payoff are discounted by the proportion of strategic opponents.

$$\begin{aligned}
\Delta V_{w,FUT}(0,L) &= \delta(1-2\epsilon_w)(1-f_r)(v_w^\infty(1) - v_w^\infty(0)) \\
&= \delta^{k_w}(1-\epsilon_w)^{k_w-1}(1-2\epsilon_w)(1-f_r)\ P \\
\Delta V_{r,FUT}(L,0) &= \delta(1-2\epsilon_r)(1-f_w)(v_r^\infty(1) - v_r^\infty(0)) \\
&= \delta^{k_r}(1-\epsilon_r)^{k_r-1}(1-2\epsilon_r)(1-f_w)\ V
\end{aligned}$$

Combining the above results, we can get the following results

$$\delta^{k_w}(1-\epsilon)^{k_w-1}(1-2\epsilon_w)(1-f_r)P \geq C$$
$$\delta^{k_r}(1-\epsilon)^{k_r-1}(1-2\epsilon_r)(1-f_w)V \geq P$$

Since we know the objective value is non-increasing in k_w and constant in k_r, setting k_w and k_r to 1 leads to the optimal social norm. Therefore, we can get the conditions as in the theorem.

A.2.4 Proof of Theorem 3.4.3

Similar to the proof in Theorem 3.4.1, the sustainable conditions can be expressed as follows,

$$\Delta V_{w,CUR}(0,L) + \Delta V_{w,FUT}(0,L) \leq 0$$
$$\Delta V_{r,CUR}(L,0) + \Delta V_{r,FUT}(L,0) \leq 0$$

Since the introduction of the transaction fees only affects the payment received by the worker, the differences of the current payoff are the same as in Lemma 3.3.2. The differences of the expected future payoff are

$$\begin{aligned}\Delta V_{w,FUT}(0,L) &= \delta(1-2\epsilon_w)(v_w^{\infty}(1) - v_w^{\infty}(0)) \\ &= \delta(1-2\epsilon_w)\delta^{k_w-1}(1-\epsilon)^{k_w-1}\,(1-m)P \\ \Delta V_{r,FUT}(L,0) &= \delta(1-2\epsilon_r)(v_r^{\infty}(1) - v_r^{\infty}(0)) \\ &= \delta(1-2\epsilon_w)\delta^{k_w-1}(1-\epsilon)^{k_w-1}\,V\end{aligned}$$

Combining the above results, we can get the following conditions

$$\delta(1-2\epsilon_w)[\delta(1-\epsilon_w)]^{k_w-1}\,(1-m)P \geq C$$
$$\delta(1-2\epsilon_r)[\delta(1-\epsilon_r)]^{k_r-1}\,V \geq P$$

We know the objective function is non-increasing in k_w and k_r, setting k_w and k_r to 1 leads to the optimal social norm. Therefore, we can get the conditions as in the theorem. Since the revenue of the platform designer is Pm, and the optimal value can be decided regardless of the value of m, the optimal m would be the largest possible m which still satisfies the condition. By solving the equality in the condition, we can get optimal m as stated in the theorem.

A.3 Additional Proofs from Chapter 4

A.3.1 Proof of Lemma 4.3.1 (virtual width)

For two vectors $\mathbf{x}, \mathbf{x}' \in \Re^m$, write $\mathbf{x}' \succeq \mathbf{x}$ if $\mathbf{x}'$ pointwise dominates $\mathbf{x}$, i.e., if $\mathbf{x}'_j \geq \mathbf{x}_j$ for all j. For two monotone contracts x, x', write $x' \succeq x$ if $\texttt{incr}(x') \succeq \texttt{incr}(x)$.

Claim A.3.1. *Consider a worker whose type satisfies the FOSD assumption and two weakly bounded contracts x, x' such that $x' \succeq x$. Let e (resp., e') be the effort levels exerted by this worker when he is offered contract x (resp., x'). Then e does not have FOSD over e'.*

Proof. For the sake of contradiction, assume that e has FOSD over e'. Note that $e \neq e'$.

Let i be the worker's type. Recall that $F_i(\pi|e)$ denotes the probability of generating an outcome $\pi' \geq \pi$ given the effort level e. Define $\mathbf{F} = (\ F_i(1|e)\ ,\ \ldots\ , F_i(m|e)\ )$, and define $\mathbf{F}'$ similarly for e'.

Let $\mathbf{x}$ and $\mathbf{x}'$ be the increment representations for x and x'. Given contract x, the worker's expected utility for effort level e is $U_i(x|e) = \mathbf{x} \cdot \mathbf{F} - c_i(e)$. Since e is the optimal effort level given this contract, we have $U_i(x|e) \geq U_i(x|e')$, and therefore

$$\mathbf{x} \cdot \mathbf{F} - \mathbf{x} \cdot \mathbf{F}' \geq c_i(e) - c_i(e').$$

Similarly, since e' is the optimal effort level given contract $\mathbf{x}'$, we have

$$\mathbf{x}' \cdot \mathbf{F}' - \mathbf{x}' \cdot \mathbf{F} \geq c_i(e') - c_i(e).$$

Combining the above two inequalities, we obtain

$$(\mathbf{x} - \mathbf{x}') \cdot (\mathbf{F} - \mathbf{F}') \geq 0. \tag{A.11}$$

Note that if (A.11) holds with equality then $U_i(x|e) = U_i(x|e')$ and $U_i(x'|e) = U_i(x'|e')$, so the worker breaks the tie between e and e' in a different way for two different contracts. This contradicts the consistent tie-breaking assumption. However, (A.11) cannot hold with a strict equality, either, because $\mathbf{x}' \succeq \mathbf{x}$ and (since e has FOSD over e') we have $\mathbf{F} \succeq \mathbf{F}'$ and $\mathbf{F}_\pi > \mathbf{F}'_\pi$ for some outcome $\pi > 0$. Therefore we obtain a contradiction, completing the proof. □

The proof of Claim A.3.1 is the only place in the paper where we directly use the consistent tie-breaking assumption. (But the rest of the paper relies on this claim.)

Claim A.3.2. *Assume all types satisfy the FOSD assumption. Consider weakly bounded contracts x, x' such that $x' \succeq x$. Then $V(x') \geq V(x)$ and $P(x') \geq P(x)$.*

Proof. Consider some worker, let i be his type. Let e and e' be the chosen effort levels for contracts $\mathbf{x}$ and $\mathbf{x}'$, respectively. By the FOSD assumption, either $e = e'$, or e' has FOSD over e, or e has FOSD over e'. Claim A.3.1 rules out the latter possibility.

Define vectors $\mathbf{F}$ and $\mathbf{F}'$ as in the proof of Claim A.3.1. Note that $\mathbf{F}' \succeq \mathbf{F}$.

Then $P = \mathbf{x} \cdot \mathbf{F}$ and $P' = \mathbf{x}' \cdot \mathbf{F}'$ is the expected payment for contracts x and x', respectively. Further, letting $\mathbf{v}$ denote the increment representation of the requester's value for each outcome, $V = \mathbf{v} \cdot \mathbf{F}$ and $V' = \mathbf{v} \cdot \mathbf{F}'$ is the expected requester's value for contracts x and x', respectively. Since $\mathbf{x}' \succeq \mathbf{x}$ and $\mathbf{F}' \succeq \mathbf{F}$, it follows that $P' \geq P$ and $V' \geq V$. Since this holds for each worker, this also holds in expectation over workers. □

To finish the proof of Lemma 4.3.1, fix a contract $x \in C$ and observe that $V(x^+) \geq V(x) \geq V(x^-)$ and $P(x^+) \geq P(x) \geq P(x^-)$, where $x^+ = x^+(C)$ and $x^- = x^-(C)$ are the two anchors.

A.3.2 Proof of Theorem 4.4.1

We now prove the main result from Section 4.4. Our high-level approach is to define a *clean execution* of an algorithm as an execution in which some high-probability events are satisfied, and derive bounds on regret conditional on the clean execution. The analysis of a clean execution does not involve any "probabilistic" arguments. This approach tends to simplify regret analysis.

We start by listing some simple invariants enforced by `AgnosticZooming`:

Invariant A.3.3. *In each round t of each execution of* `AgnosticZooming`*:*

(a) All active cells are relevant,

(b) Each candidate contract is contained in some active cell,

(c) $W_t(C) \leq 5\,\mathtt{rad}_t(C)$ for each active composite cell C.

Note that the zooming rule is essential to ensure Invariant A.3.3(c).

A.3.3 Analysis of the randomness

Definition 3 (Clean Execution)**.** *An execution of* `AgnosticZooming` *is called* clean *if for each round* t *and each active cell* C *it holds that*

$$|U(C) - U_t(C)| \leq \texttt{rad}_t(C), \tag{A.12}$$

$$|\texttt{VirtWidth}(C) - W_t(C)| \leq 4\,\texttt{rad}_t(C) \qquad \textit{(if } C \textit{ is composite)}. \tag{A.13}$$

Lemma A.3.4. *Assume* $c_{\texttt{rad}} \geq 16$ *and* $T \geq \max(1 + 2^m, 18)$*. Then:*

(a) $\Pr[$ *(A.12) holds* $\quad \forall$ *rounds* t*, active cells* $C\ ] \geq 1 - 2\,T^{-2}$.

(b) $\Pr[$ *(A.13) holds* $\quad \forall$ *rounds* t*, active composite cells* $C\ ] \geq 1 - 16\,T^{-2}$.

Consequently, an execution of `AgnosticZooming` *is clean with probability at least* $1 - 1/T$.

Lemma A.3.4 follows from the standard concentration inequality known as "Chernoff Bounds". However, one needs to be careful about conditioning and other details.

Proof of Lemma A.3.4(a). Consider an execution of `AgnosticZooming`. Let N be the total number of activated cells. Since at most 2^m cells can be activated in any one round, $N \leq 1 + 2^m T \leq T^2$. Let C_j be the $\min(j, N)$-th cell activated by the algorithm. (If multiple "quadrants" are activated in the same round, order them according to some fixed ordering on the quadrants.)

Fix some feasible cell C and $j \leq T^2$. We claim that

$$\Pr[\ |U(C) - U_t(C)| \leq \texttt{rad}_t(C) \text{ for all rounds } t \mid C_j = C\ ] \geq 1 - 2\,T^{-4}. \tag{A.14}$$

Let $n(C) = n_{1+T}(C)$ be the total number of times cell C is chosen by the algorithm. For each $s \in \mathbb{N}$: $1 \leq s \leq n(C)$ let U_s be the requester's utility in the round when C is chosen for the s-th time. Further, let $\mathcal{D}_C$ be the distribution of U_1, conditional on the event $n(S) \geq 1$. (That is, the per-round reward from choosing cell C.) Let $U'_1, \ldots, U'_T$ be a family of mutually independent random variables, each with distribution $\mathcal{D}_C$. Then for each $n \leq T$, conditional on the event $\{C_j = C\} \wedge \{n(C) = n\}$, the tuple $(U_1, \ldots, U_n)$ has the same joint distribution as the tuple

$(U'_1, \ldots, U'_n)$. Consequently, applying Chernoff Bounds to the latter tuple, it follows that

$$\begin{aligned}\Pr\Big[\ \big|U(C) - \tfrac{1}{n}\textstyle\sum_{s=1}^{n} U_s\big| \le \sqrt{\tfrac{1}{n}\, c_{\texttt{rad}}\, \log(T)}\ \Big|\ \{C_j = C\} \wedge \{n(C) = n\}\ \Big] \\ \ge 1 - 2\,T^{-2c_{\texttt{rad}}} \ge 1 - 2\,T^{-5}.\end{aligned}$$

Taking the Union Bound over all $n \le T$, and plugging in $\texttt{rad}_t(C_j)$, $n_t(C_j)$, and $U_t(C_j)$, we obtain (A.14).

Now, let us keep j fixed in (A.14), and integrate over C. More precisely, let us multiply both sides of (A.14) by $\Pr[C_j = C]$ and sum over all feasible cells C. We obtain, for all $j \le T^2$:

$$\Pr\left[\ |U(C_j) - U_t(C_j)| \le \texttt{rad}_t(C_j) \text{ for all rounds } t\ \right] \ge 1 - 2\,T^{-4}. \tag{A.15}$$

(Note that to obtain (A.15), we do *not* need to take the Union Bound over all feasible cells C.) To conclude, we take the Union Bound over all $j \le 1 + T^2$. □

Proof Sketch of Lemma A.3.4(b). We show that

$$\Pr\left[\ \big|V^+(C) - V_t^+(C)\big| \le \texttt{rad}_t(C)\ \forall \text{rounds } t, \text{active composite cells } C\ \right] \ge 1 - \tfrac{4}{T^2}, \tag{A.16}$$

and similarly for $V^-()$, $P^+()$ and $P^-()$. Each of these four statements is proved similarly, using the technique from Lemma A.3.4(a). In what follows, we sketch the proof for one of the four cases, namely for (A.16).

For a given composite cell C, we are only interested in rounds in which anchor $x^+(C)$ is selected by the algorithm. Letting $n_t^+(C)$ be the number of times this anchor is chosen up to time t, let us define the corresponding notion of "confidence radius":

$$\texttt{rad}_t^+(C) = \frac{1}{2}\sqrt{\frac{c_{\texttt{rad}}\,\log T}{n_t^+(C)}}.$$

With the technique from the proof of Lemma A.3.4(a), we can establish the following high-probability event:

$$\big|V^+(C) - V_t^+(C)\big| \le \texttt{rad}_t^+(C). \tag{A.17}$$

More precisely, we can prove that

$$\Pr[\text{ (A.17) holds} \quad \forall \text{ rounds } t, \text{ active composite cells } C\ ] \geq 1 - 2\,T^{-2}.$$

Further, we need to prove that w.h.p. the anchor $x^+(C)$ is played sufficiently often. Noting that $\mathbb{E}[n_t^+(C)] = \frac{1}{2}\,n_t(C)$, we establish an auxiliary high-probability event:[1]

$$n_t^+(C) \geq \tfrac{1}{2}\,n_t(C) - \tfrac{1}{4}\,\texttt{rad}_t(C). \tag{A.18}$$

More precisely, we can use Chernoff Bounds to show that, if $c_{\texttt{rad}} \geq 16$,

$$\Pr[\text{ (A.18) holds} \quad \forall \text{ rounds } t, \text{ active composite cells } C\ ] \geq 1 - 2\,T^{-2}. \tag{A.19}$$

Now, letting $n_0 = (c_{\texttt{rad}}\,\log T)^{1/3}$, observe that

$$\begin{aligned} n_t(C) \geq n_0 \quad &\Rightarrow \quad n_t^+(C) \geq \tfrac{1}{4}\,n_t(C) \quad \Rightarrow \quad \texttt{rad}_t^+(C) \leq \texttt{rad}_t(C), \\ n_t(C) < n_0 \quad &\Rightarrow \quad \texttt{rad}_t(C) \geq 1 \quad \Rightarrow \quad \left|V^+(C) - V_t^+(C)\right| \leq \texttt{rad}_t(C). \end{aligned}$$

Therefore, once Equations Equation A.17 and Equation A.18 hold, we have $\left|V^+(C) - V_t^+(C)\right| \leq \texttt{rad}_t(C)$. This completes the proof of (A.16). □

A.3.4 Analysis of a clean execution

The rest of the analysis focuses on a clean execution. Recall that C_t is the cell chosen by the algorithm in round t.

Claim A.3.5. *In any clean execution, $I(C_t) \geq \texttt{OPT}(X_{\texttt{cand}})$ for each round t.*

Proof. Fix round t, and let x^* be any candidate contract. By Invariant A.3.3(b), there exists an active cell, call it C_t^*, which contains x^*.

We claim that $I_t(C_t^*) \geq U(x^*)$. We consider two cases, depending on whether C_t^* is atomic. If C_t^* is atomic then the anchor is unique, so $U(C_t^*) = U(x^*)$, and $I_t(C_t^*) \geq U(x^*)$ by the clean

[1] The constant $\frac{1}{4}$ in (A.18) is there to enable a consistent choice of n_0 in the remainder of the proof.

execution. If C_t^* is composite then

$$\begin{aligned}
I_t(C_t^*) &\geq U(C_t^*) + \texttt{VirtWidth}(C_t^*) && \text{by clean execution} \\
&\geq U(C_t^*) + \texttt{width}(C_t^*) && \text{by Lemma 4.3.1} \\
&\geq U(x^*) && \text{by definition of } \texttt{width}, \text{ since } x^* \in C_t^*.
\end{aligned}$$

We have proved that $I_t(C_t^*) \geq U(x^*)$. Now, by the selection rule we have $I_t(C_t) \geq I_t(C_t^*) \geq U(x^*)$. Since this holds for any candidate contract x^*, the claim follows. □

Claim A.3.6. *In any clean execution, for each round t, the index $I_t(C_t)$ is upper-bounded as follows:*

(a) if C_t is atomic then $I(C_t) \leq U(C_t) + 2\,\texttt{rad}_t(C_t)$.

(b) if C_t is composite then $I(C_t) \leq U(x) + O(\texttt{rad}_t(C_t))$ for each contract $x \in C_t$.

Proof. Fix round t. Part (a) follows because $I_t(C_t) = U_t(C_t) + \texttt{rad}_t(C_t)$ by definition of the index, and $U_t(C_t) \leq U(C_t) + \texttt{rad}_t(C_t)$ by clean execution.

For part (b), fix a contract $x \in C_t$. Then:

$$\begin{aligned}
U_t(C_t) &\leq U(C_t) + \texttt{rad}_t(C_t) && \text{by clean execution} \\
&\leq U(x) + \texttt{width}(C_t) + \texttt{rad}_t(C_t) && \text{by definition of width} \\
&\leq U(x) + \texttt{VirtWidth}(C_t) + \texttt{rad}_t(C_t) && \text{by Lemma 4.3.1} \\
&\leq U(x) + W_t(C_t) + 5\,\texttt{rad}_t(C_t) && \text{by clean execution.} \qquad \text{(A.20)} \\
I_t(C_t) &= U_t(C_t) + W_t(C_t) + 5\,\texttt{rad}_t(C_t) && \text{by definition of index} \\
&\leq U(x) + 2\,W_t(C_t) + 10\,\texttt{rad}_t(C_t) && \text{by (A.20)} \\
&\leq U(x) + 20\,\texttt{rad}_t(C_t) && \text{by Invariant A.3.3(c).} \qquad \square
\end{aligned}$$

For each relevant cell C, define badness $\Delta(C)$ as follows. If C is composite, $\Delta(C) = \sup_{x \in C} \Delta(x)$ is the maximal badness among all contracts in C. If C is atomic and $x \in C$ is the unique candidate contract in C, then $\Delta(C) = \Delta(x)$.

Claim A.3.7. *In any clean execution, $\Delta(C) \leq O(\texttt{rad}_t(C))$ for each round t and each active cell C.*

Proof. By Claims A.3.5 and A.3.6, $\Delta(C_t) \leq O(\texttt{rad}_t(C_t))$ for each round t. Fix round t and let C be an active cell in this round. If C has never be selected before round t, the claim is trivially true. Else, let s be the most recent round before t when C is selected by the algorithm. Then $\Delta(C) \leq O(\texttt{rad}_s(C))$. The claim follows since $\texttt{rad}_s(C) = \texttt{rad}_t(C)$. □

Claim A.3.8. *In a clean execution, each cell C is selected $\leq O(\log T/(\Delta(C))^2)$ times.*

Proof. By Claim A.3.7, $\Delta(C) \leq O(\texttt{rad}_T(C))$. The claim follows from the definition of $\texttt{rad}_T$ in (4.3). □

Let $n(x)$ and $n(C)$ be the number of times contract x and cell C, respectively, are chosen by the algorithm. Then regret of the algorithm is

$$R(T|X_{\texttt{cand}}) = \sum_{x \in X} n(x)\ \Delta(x) \leq \sum_{\text{cells } C}\ n(C)\,\Delta(C). \tag{A.21}$$

The next result (Lemma A.3.9) upper-bounds the right-hand side of (A.21) for a clean execution. By Lemma A.3.4, this suffices to complete the proof of Theorem 4.4.1

Lemma A.3.9. *Consider a clean execution of* `AgnosticZooming`. *For any $\delta \in (0,1)$,*

$$\sum_{\text{cells } C}\ n(C)\,\Delta(C) \leq \delta T + O(\log T) \sum_{\epsilon = 2^{-j} \geq \delta:\ j \in \mathbb{N}}\ \frac{|\mathcal{F}_\epsilon(X_{2\epsilon})|}{\epsilon}.$$

The proof of Lemma A.3.9 relies on some simple properties of $\Delta(\cdot)$, stated below.

Claim A.3.10. *Consider two relevant cells $C \subset C_{\texttt{p}}$. Then:*

(a) $\Delta(C) \leq \Delta(C_{\texttt{p}})$.

(b) If $\Delta(C) \leq \epsilon$ for some $\epsilon > 0$, then C overlaps with X_ϵ.

Proof. To prove part (a), one needs to consider two cases, depending on whether cell $C_{\texttt{p}}$ is composite. If it is, the claim follows trivially. If $C_{\texttt{p}}$ is atomic, then C is atomic, too, and so $\Delta(C) = \Delta(C_{\texttt{p}}) = \Delta(x)$, where x is the unique candidate contract in $C_{\texttt{p}}$.

For part (b), there exists a candidate contract $x \in C$. It is easy to see that $\Delta(x) \leq \Delta(C)$ (again, consider two cases, depending on whether C is composite.) So, $x \in X_\epsilon$. □

Proof of Lemma A.3.9. Let Σ denote the sum in question. Let $\mathcal{A}^*$ be the collection of all cells ever activated by the algorithm. Among such cells, consider those with badness on the order of ϵ:

$$\mathcal{G}_\epsilon := \{\, C \in \mathcal{A}^* : \Delta(C) \in [\epsilon, 2\epsilon) \,\}.$$

By Claim A.3.8, the algorithm chooses each cell $C \in \mathcal{G}_\epsilon$ at most $O(\log T/\epsilon^2)$ times, so $n(C)\,\Delta(C) \leq O(\log T/\epsilon)$.

Fix some $\delta \in (0,1)$ and observe that all cells C with $\Delta(C) \leq \delta$ contribute at most δT to Σ. Therefore it suffices to focus on $\mathcal{G}_\epsilon$, $\epsilon \geq \delta/2$. It follows that

$$\Sigma \leq \delta T + O(\log T) \textstyle\sum_{\epsilon=2^{-i}\geq\delta/2} \frac{|\mathcal{G}_\epsilon|}{\epsilon}. \tag{A.22}$$

We bound $|\mathcal{G}_\epsilon|$ as follows. Consider a cell $C \in \mathcal{G}_\epsilon$. The cell is called a *leaf* if it is never zoomed in on (i.e., removed from the active set) by the algorithm. If C is activated in the round when cell $C_{\mathtt{p}}$ is zoomed in on, $C_{\mathtt{p}}$ is called the *parent* of C. We consider two cases, depending on whether or not C is a leaf.

(i) Assume cell C is not a leaf. Since $\Delta(C) < 2\epsilon$, C overlaps with $X_{2\epsilon}$ by Claim A.3.10(b). Note that C is zoomed in on in some round, say in round $t-1$. Then

$$\begin{aligned} 5\,\mathtt{rad}_t(C) &\leq W_t(C) && \text{by the zooming rule} \\ &\leq \mathtt{VirtWidth}(C) + 4\,\mathtt{rad}_t(C) && \text{by clean execution,} \end{aligned}$$

so $\mathtt{rad}_t(C) \leq \mathtt{VirtWidth}(C)$. Therefore, using Claim A.3.7, we have

$$\epsilon \leq \Delta(C) \leq O(\mathtt{rad}_t(C)) \leq O(\mathtt{VirtWidth}(C)).$$

It follows that $C \in \mathcal{F}_{\Omega(\epsilon)}(X_{2\epsilon})$.

(ii) Assume cell C is a leaf. Let $C_{\mathtt{p}}$ be the parent of C. Since $C \subset C_{\mathtt{p}}$, we have $\Delta(C) \leq \Delta(C_{\mathtt{p}})$ by Claim A.3.10(a). Therefore, invoking case (i), we have

$$\epsilon \leq \Delta(C) \leq \Delta(C_{\mathtt{p}}) \leq O(\mathtt{VirtWidth}(C_{\mathtt{p}})).$$

Since $\Delta(C) < 2\epsilon$, C overlaps with $X_{2\epsilon}$ by Claim A.3.10(b), and therefore so does $C_{\mathtt{p}}$. It follows that $C_{\mathtt{p}} \in \mathcal{F}_{\Omega(\epsilon)}(X_{2\epsilon})$.

Combing these two cases, it follows that $|\mathcal{G}_\epsilon| \leq (2^m + 1)\left|\mathcal{F}_{\Omega(\epsilon)}(X_{2\epsilon})\right|$. Plugging this into Equation A.22 and making an appropriate substitution $\epsilon \to \Theta(\epsilon)$ to simplify the resulting expression, we obtain the regret bound in Theorem 4.4.1 □

A.3.5 Discussion: Monotone contracts may not be optimal

In this section we provide an example of a problem instance for which all monotone contracts are suboptimal (at least when restricting attention to only those contracts with non-negative payoffs). In this example, there are three non-null outcomes (i.e., $m = 3$), and two non-null effort levels, "low" effort and "high" effort, which we denote e_ℓ and e_h respectively. There is only a single worker type. Since there is only one type, we drop the subscript when describing the cost function c. We let $c(e_\ell) = 0$, and let $c(e_h)$ be any positive value less than $0.5(v(2) - v(1))$. If a worker chooses low effort, the outcome is equally likely to be 1 or 3. If the worker chooses high effort, it is equally likely to be 2 or 3. It is easy to verify that this type satisfies the FOSD assumption. Finally, for simplicity, we assume that all workers break ties between high effort and any other effort level in favor of high effort, and that all workers break ties between low effort and the null effort level in favor of low effort.

Let's consider the optimal contract. Since there is just a single worker type and all workers of this type break ties in the same way, we can consider separately the best contract that would make all workers choose the null effort level, the best contract that would make all workers choose low effort, and the best contract that would make all workers choose high effort, and compare the requester's expected value for each.

Since $c(e_\ell) = 0$ and workers break ties between low effort and null effort in favor of low effort, there is no contract that would cause workers to choose null effort; workers always prefer low effort to null effort.

It is easy to see that the best contract (in terms of requester expected value) that would make workers choose low effort would set $x(1) = x(3) = 0$ and $x(2)$ sufficiently low that workers would not be enticed to choose high effort; setting $x(2) = 0$ is sufficient. In this case, the expected value of the requester would be $0.5(v(1) + v(3))$.

Now let's consider contracts that cause workers to choose high effort. If a worker chooses high effort, the expected value to the requester is

$$0.5(v(2) - x(2) + v(3) - x(3)). \tag{A.23}$$

Workers will choose high effort if and only if

$$0.5(x(1) + x(3)) \leq 0.5(x(2) + x(3)) - c(e_h)$$

or

$$0.5x(1) \leq 0.5x(2) - c(e_h). \tag{A.24}$$

So to find the contract that maximizes the requester's expected value when workers choose high effort, we want to maximize Equation A.23 subject to the constraint in Equation A.24. Since $x(3)$ doesn't appear in Equation A.24, we can set it to 0 to maximize Equation A.23. Since $x(1)$ does not appear in Equation A.23, we can set $x(1) = 0$ to make Equation A.24 as easy as possible to satisfy. We can then see that the optimal occurs when $x(2) = 2c(e_h)$.

Plugging this contact x into Equation A.23, the expected utility in this case is $0.5(v(2)+v(3))-c(e_h)$. Since we assumed that $c(e_h) < 0.5(v(2) - v(1)))$, this is strictly preferable to the constant 0 contract, and is in fact the unique optimal contract. Since $x(2) > x(3)$, the unique optimal contract is not monotonic.

A.4 Additional Proofs from Chapter 6

A.4.1 Tools for Converting Regret-Minimizing Algorithms

Lemma A.4.1 (Lemma 6.3.1)**.** *If Algorithm 5 is run with (nonzero) probabilities $q_1, \ldots, q_T$ and FTRL as the learning algorithm, then the expected regret is bounded by*

$$\frac{\beta}{\eta} + 2\eta\mathbb{E}\left[\sum_{t=1}^{T} \frac{\Delta_{h_t,f_t}^2}{q_t}\right],$$

where β is a constant depending on $\mathcal{H}$, η is a parameter of the algorithm, and the expectation is over any randomness in the choices of h_t and q_t.

Proof. Let $\mathbf{h}^* = \inf_{h \in \mathcal{H}} \sum_t f_t(h)$. We wish to prove that

$$\mathbb{E}_{\{h_t, q_t\}} \sum_t f_t(h_t) \leq \sum_t f_t(\mathbf{h}^*) \ + \ R$$

where $\{h_t, q_t\}$ is shorthand for $\{h_1, q_1, \ldots, h_T, q_T\}$ and

$$R = \frac{\beta}{\eta} + 2\eta \mathbb{E}_{\{h_t, q_t\}} \left[\sum_t \frac{\Delta^2_{h_t, f_t}}{q_t} \right].$$

As a prelude, note that in general these expectations could be quite tricky to deal with. We consider a fixed input sequence $f_1, \ldots, f_T$, but each random variable q_t, h_t depends on the prior sequence of variables and outcomes. However, we will see that the nice feature of the importance-weighting technique of Algorithm 5 helps make this problem tractable.

Some preliminaries: Define the importance-weighted loss function at time t to be the random variable

$$\hat{f}_t(h) = \begin{cases} \frac{f_t(h)}{q_t} & \text{obtain } f_t \\ 0 & \text{o.w.} \end{cases}$$

Let $\mathbb{1}_t$ be the indicator random variable equal to 1 if we obtain f_t, which occurs with probability q_t, and equal to 0 otherwise. Then notice that for any hypothesis h,

$$\begin{aligned} \hat{f}_t(h) &= \mathbb{1}_t \frac{f_t(h)}{q_t} \\ \implies \mathbb{E}_{\mathbb{1}_t} \left[\hat{f}_t(h) \mid q_t \right] &= f_t(h). \end{aligned} \tag{A.25}$$

To be clear, the expectation is over the random outcome whether or not we obtain datapoint f_t conditioned on the value of q_t; and conditioned on the value of q_t, by definition we obtain datapoint f_t with probability q_t and obtain the 0 function otherwise.

Now we proceed with the proof. For any method of choosing $q_1, \ldots, q_T$ and any resulting outcomes of $\mathbb{1}_t$, Algorithm 5 reduces to running the Follow-the-Regularized-Leader algorithm on the sequence of convex loss functions $\hat{f}_1, \ldots, \hat{f}_T$. Thus, by the regret bound proof for FTRL (Lemma A.4.2), FTRL guarantees that for every fixed "reference hypothesis" $\mathbf{h} \in \mathcal{H}$:

$$\sum_t \hat{f}_t(h_t) \leq \sum_t \hat{f}_t(\mathbf{h}) \ + \ \hat{R}$$

where

$$\hat{R} = \frac{\beta}{\eta} \; + \; 2\eta \sum_t \Delta^2_{h_t, \hat{f}_t}$$
$$= \frac{\beta}{\eta} \; + \; 2\eta \sum_t \mathbb{1}_t \frac{\Delta^2_{h_t, f_t}}{q_t^2}.$$

(Recall that $\Delta_{h,f} = \|\nabla f(h)\|_\star$.) Now we will take the expectation of both sides, separating out the expectation over the choice of q_t, over h_t, and over $\mathbb{1}_t$:

$$\sum_t \mathbb{E}_{h_t, q_t}\left[\mathbb{E}_{\mathbb{1}_t}\left[\hat{f}_t(h_t) \mid h_t, q_t\right]\right] \leq \sum_t \mathbb{E}_{h_t, q_t}\left[\mathbb{E}_{\mathbb{1}_t}\left[\hat{f}_t(\mathbf{h}) \mid h_t, q_t\right]\right] \; + \; \mathbb{E}_{\{h_t, q_t\}}\left[\mathbb{E}_{\{\mathbb{1}_t\}}\left[\hat{R} \mid \{h_t, q_t\}\right]\right].$$

Use the importance-weighting observation above (A.25):

$$\mathbb{E}_{\{h_t, q_t\}} \sum_t f_t(h_t) \leq \sum_t f_t(\mathbf{h}) \; + \; R$$

where

$$R = \frac{\beta}{\eta} \; + \; 2\eta \mathbb{E}_{\{h_t, q_t\}}\left[\sum_t \frac{\Delta^2_{h_t, f_t}}{q_t}\right].$$

In particular, because this holds for every reference hypothesis $\mathbf{h}$, it holds for $\mathbf{h}^*$. □

Lemma A.4.2. *Let G be 1-strongly convex with respect to some norm $\|\cdot\|$. The regret of Follow-The-Regularized-Leader algorithm with regularizer G and convex loss functions $f_1, \ldots, f_T$ can be bounded by*

$$\frac{\beta}{\eta} + 2\eta \sum_t \Delta^2_{h_t, f_t} \; ,$$

where β is the upper bound of $G(\cdot)$.

Proof. We reproduce the standard proof. First, the regret of Follow-The-Regularized-Leader can be bounded by

$$\frac{1}{\eta}(R(h_T) - R(h_1)) + \sum_{t=1}^{T}(\ell(h_t, f_t) - \ell(h_{t+1}, f_t)).$$

Below we show that $\ell(h_t, f_t) - \ell(h_{t+1}, f_t) \leq 2\eta\|\nabla\ell(h_t, f_t)\|^2_\star$.

Define $\Phi_t(h) = R(h)/\eta + \sum_{i=1}^t \ell(h, f_i)$. By definition, we know $h_t = \arg\min_h \Phi_{t-1}(h)$. Since $\ell(\cdot)$ is convex and $R(\cdot)$ is 1-strongly convex, we know $\Phi_t(\cdot)$ is $(1/\eta)$-strongly convex for all t. Therefore, since h_{t+1} minimizes Φ_t, by definition of strong convex, we get

$$\Phi_t(h_t) \geq \Phi_t(h_{t+1}) + \frac{1}{2\eta}\|h_t - h_{t+1}\|^2$$

After simple manipulations, we get

$$\begin{aligned}\|h_t - h_{t+1}\|^2 &\leq 2\eta(\Phi_t(h_t) - \Phi_t(h_{t+1})) \\ &= 2\eta(\Phi_{t-1}(h_t) - \Phi_{t-1}(h_{t+1})) + 2\eta(\ell(h_t, f_t) - \ell(h_{t+1}, f_t)) \\ &\leq 2\eta(\ell(h_t, f_t) - \ell(h_{t+1}, f_t))\end{aligned}$$

The last inequality comes from the fact that h_t is the minimizer of Φ_{t-1}.

Since $\ell(\cdot)$ is convex, we have

$$\begin{aligned}\ell(h_t, f_t) - \ell(h_{t+1}, f_t) &\leq (h_t - h_{t+1})\nabla\ell(h_t, f_t) \\ &\leq \|h_t - h_{t+1}\|\|\nabla\ell(h_t, f_t)\|_\star\end{aligned}$$

The last inequality comes from the generalized Cauchy-Schwartz inequality.

Combining the above two inequalities together, we get

$$\ell(h_t, f_t) - \ell(h_{t+1}, f_t) \leq \|\nabla\ell(h_t, f_t)\|_\star\sqrt{2\eta(\ell(h_t, f_t) - \ell(h_{t+1}, f_t))}$$

By squaring and shifting sides,

$$\ell(h_t, f_t) - \ell(h_{t+1}, f_t) \leq 2\eta\|\nabla\ell(h_t, f_t)\|_\star^2$$

The proof is completed by inserting the inequality into the regret bound. □

A.4.2 No regret "at-cost" setting

A.4.2.1 At-cost upper bounds

Lemma A.4.3 (Lemma 6.5.1)**.** *To minimize the regret bound of Lemma 6.3.1, the optimal choice of sampling probability is of the form*

$$q_t = \min\left\{1\ ,\ \frac{\Delta_{h_t,f_t}}{K^*\sqrt{c_t}}\right\}.$$

The normalization factor $K^* \approx \frac{T}{B}\gamma$.

Proof. Recall that the regret bound of Lemma 6.3.1 is

$$\frac{\beta}{\eta} + 2\eta\mathbb{E}\sum_t \frac{\Delta^2_{h_t,f_t}}{q_t}$$

where q_t is the probability with which we choose to purchase arrival (c_t, f_t). We will solve for the choices of q_t for each t.

Since β is a constant and η a parameter to be tuned later, our problem is to minimize the summation term in this regret bound. This yields the following optimization problem:

$$\begin{aligned} \min_{q_t} & \sum_t \frac{\Delta^2_{h_t,f_t}}{q_t} \\ \text{s.t.} \quad & \sum_t q_t \cdot c_t \le B \\ & q_t \le 1 \qquad (\forall t). \end{aligned}$$

The first constraint is the expected budget constraint, as we take each point (c_t, f_t) with probability q_t and pay c_t if we do. The second constrains each q_t to be a probability.

To be completely formal, our goal is to minimize the expectation of the summation in the objective, as each h_t and q_t are random variables (they depend on the previous steps). However, our approach will be to optimize this objective pointwise: For every prior sequence $h_1, \ldots, h_t$ and $q_1, \ldots, q_{t-1}$, we pick the optimal q_t. Therefore in the proof we will elide the expectation operators and argument. Similarly, since the budget constraint holds for all choices of q_t that we make, we elide the expectation over the randomness in q_t.

The Lagrangian of this problem is

$$L(\lambda, \{q_t, \alpha_t\}) = \sum_t \frac{\Delta^2_{h_t,f_t}}{q_t} + \lambda\left(\sum_t q_t \cdot c_t \; - \; B\right) \; + \; \sum_t \alpha_t \left(q_t - 1\right)$$

with each $\lambda, q_t, \alpha_t \ge 0$. At optimum,

$$\begin{aligned} 0 &= \frac{\partial L}{\partial q_t} \\ &= -\frac{\Delta^2_{h_t,f_t}}{q_t^2} + \lambda c_t + \alpha_t, \end{aligned}$$

implying that

$$q_t = \frac{\Delta_{h_t,f_t}}{\sqrt{\lambda c_t + \alpha_t}}.$$

By complementary slackness, $\alpha_t(q_t - 1) = 0$ at optimum, so consider two cases. If $\alpha_t > 0$, then $q_t = 1$. On the other hand, if $q_t < 1$, then $\alpha_t = 0$. Thus we may more simply write

$$q_t = \min\left\{1\ ,\ \frac{\Delta_{h_t,f_t}}{\sqrt{\lambda c_t}}\right\}.$$

Therefore, our normalization constant $K^* = \sqrt{\lambda}$. To solve for λ, by complementary slackness, $\lambda\left(\sum_t q_t \cdot c_t\ -\ B\right) = 0$. If $\lambda = 0$, then the form of q_t and prior discussion implies that all $q_t = 1$, and we have $\sum_t c_t \leq B$; in other words, we have enough budget to purchase every point. Otherwise, the budget constraint is tight and $\sum_t q_t \cdot c_t = B$, so

$$\sum_t c_t \cdot \min\left\{1\ ,\ \frac{\Delta_{h_t,f_t}}{\sqrt{\lambda c_t}}\right\} = B.$$

Let us call those points that are taken with provability $q_t = 1$ "valuable" and the others "less valuable", and let S be the set of less valuable points, $S = \{t : q_t < 1\}$. Then we can rewrite as

$$\sum_{t\notin S} c_t\ +\ \sum_{t\in S} \frac{\Delta_{h_t,f_t}\sqrt{c_t}}{\sqrt{\lambda}} = B,$$

so

$$K^* = \sqrt{\lambda} = \frac{1}{B - \sum_{t\notin S} c_t} \sum_{t\in S} \Delta_{h_t,f_t}\sqrt{c_t}.$$

This completes the proof. Let us make several final comments and observations, however. First, if the budget is small relative to the amount of data, then with Lipschitz loss functions, no data points will be taken with probability $q_t = 1$, so S will equal all of T. In this case, the expectation of K^* is exactly $\frac{T}{B}\gamma$, which is the meaning of our informal statement $K^* \approx \frac{T}{B}\gamma$.

Second, this K^* is optimal "pointwise", in that it includes advance knowledge of which data points will be taken and which hypotheses will be posted. However, notice that, to satisfy the budget constraint, it suffices to take the expectation and choose a normalization constant

$$K = \mathbb{E}\left[\frac{1}{B - \sum_{t\notin S} c_t} \sum_{t\in S} \Delta_{h_t,f_t}\sqrt{c_t}\right].$$

Third, as noted above, the extreme case is when all $q_t < 1$ and in this case the above $K = \frac{T}{B}\gamma$ exactly. While this will not be "as optimal" for the specific random outcomes of this sequence, it will suffice to prove good upper bounds on regret. Furthermore, it holds that *any* choice of $K \geq \frac{T}{B}\gamma$ satisfies the expected budget constraint; and (by setting η as a function of K) suffices to prove an upper bound on regret. □

Theorem A.4.4 (Theorem 6.3.3)**.** *There is a mechanism for the "at-cost" problem of data purchasing for regret minimization that interfaces with FTRL and guarantees to meet the expected budget constraint, where for a parameter $\gamma \in [0,1]$ (Definition 2),*

1. *The expected regret is bounded by $O\left(\max\left\{\frac{T}{\sqrt{B}}\gamma\ ,\ \sqrt{T}\right\}\right)$.*
2. *This is optimal in that no mechanism can have a better guarantee (up to constant factors).*
3. *The pricing strategy is to set $K = O\left(\frac{T}{B}\gamma\right)$ and draw $\pi_t(f)$ randomly according to a distribution such that*

$$\Pr[\pi_t(f) \geq c] = \min\left\{1\ ,\ \frac{\Delta_{h_t,f}}{K\sqrt{c}}\right\}.$$

The only prior knowledge required is an estimate of γ up to a constant factor.

Proof. The lower bound proof appears in Theorem 6.5.3.

For the upper bound, we will give a more careful argument first, obtaining a more subtle bound capturing the two extremes in the regret bound as well as the spectrum in between. We will then simplify to get the theorem statement.

First, note as pointed out in the proof of Lemma 6.5.1 that choosing any $K \geq \frac{T}{B}\gamma \geq \mathbb{E}[K^*]$ satisfies the expected budget constraint, as each probability of purchase q_t only decreases. We now just need to show that if we know γ to within a constant factor larger, *i.e.* set $K = O\left(\frac{T}{B}\gamma\right)$ and η appropriately, then we achieve the regret bound.

By Lemma 6.3.1, for any choices of q_t and the learning parameter η, the regret bound satisfies

$$Regret \leq \frac{\beta}{\eta} + 2\eta\mathbb{E}\sum_t \frac{\Delta^2_{h_t,f_t}}{q_t} \tag{A.26}$$

where β is a constant. Our strategy is to set

$$q_t = \min\left\{1\ ,\ \frac{\Delta_{h_t,f_t}}{K\sqrt{c_t}}\right\}.$$

Recall from the proof of Lemma 6.5.1 that in the optimal solution there were in general "valuable" points for which the probability of purchase was $q_t = 1$ and "less-valuable" points where $q_t < 1$. We had $S = \{t : q_t < 1\}$. Thus the summation term in the regret bound becomes

$$\mathbb{E}\sum_{t\notin S}\Delta^2_{h_t,f_t}\ +\ \mathbb{E}\sum_{t\in S}\Delta_{h_t,f_t}\sqrt{c_t}K. \tag{A.27}$$

Before we prove the theorem statement, let us show how to achieve the more subtle bound. So for the sake of this argument, let $\gamma_S = \frac{1}{|S|}\mathbb{E}\sum_{t\in S}\Delta_{h_t,f_t}\sqrt{c_t}$. Let K_S approximate the more precise form derived in the proof of Lemma 6.5.1; that is,

$$K_S = O\left(\frac{|S|}{B - \sum_{t\notin S} c_t}\gamma_S\right).$$

Then the summation term of the regret bound (Expression A.27) is at most a constant times

$$\begin{aligned}&\sum_{t\notin S}\Delta^2_{h_t,f_t} \ + \ \frac{|S|^2}{B - \sum_{t\notin S} c_t}\gamma_S^2 \\ &\leq T - |S| \ + \ + \ \frac{|S|^2}{B - \sum_{t\notin S} c_t}\gamma_S^2 \end{aligned} \tag{A.28}$$

as each $\Delta_{h_t,f_t} \leq 1$. It remains to select the parameter η to use for the learning algorithm and plug into the original regret bound, Expression A.26. If the algorithm has an accurate estimate of K_S, $|S|$, and $\sum_{t\notin S} c_t$, then it can set η equal to the square root of one over Expression A.28. (Note this may be achievable by tuning η online as well, perhaps even with a theoretical guarantee.) In this case, the regret bound is

$$Regret \leq O\left(\sqrt{T - |S| \ + \ \frac{|S|^2}{B - \sum_{t\notin S} c_t}\gamma_S^2}\right).$$

Note that as $B \to 0$, $|S| \to T$, and as $B \to \sum_t c_t$, $|S| \to 0$.

Now let us actually prove the Theorem as stated. Let $\gamma = \sum_t \Delta_{h_t,f_t}\sqrt{c_t}$ and let $K = \frac{T}{B}\gamma$. The summation term in the regret bound, Expression A.27, is upper-bounded by

$$\begin{aligned}&T + (T\gamma)K \\ &= T + \frac{T^2}{B}\gamma^2\end{aligned}$$

using that $T\gamma \geq \sum_{t\in S}\Delta_{h_t,f_t}\sqrt{c_t}$ since it is a summation over more (positive) terms. Now by Expression A.27,

$$Regret \leq \frac{\beta}{\eta} + 2\eta\left(T + \frac{T^2}{B}\gamma^2\right).$$

Setting

$$\eta = \Theta\left(1/\max\left\{\sqrt{T}\,,\ \frac{T}{\sqrt{B}}\gamma\right\}\right)$$

gives a regret bound of the order of $1/\eta$. □

A.4.2.2 At-cost lower bounds

Theorem A.4.5 (Theorem 6.5.2)**.** *Suppose all costs $c_t = 1$. No algorithm for the at-cost online data-purchasing problem has regret better than $O\left(\frac{T}{\sqrt{B}}\right)$; that is, for every algorithm, there exists an input sequence on which its regret is $\Omega\left(\frac{T}{\sqrt{B}}\right)$.*

Proof. Consider two possible input distributions: i.i.d. flips of a coin that has probability $\frac{1}{2} + \epsilon$ of heads, or of one with probability $\frac{1}{2} - \epsilon$.

It will suffice to prove the following:

Claim 1: If there is an algorithm with budget B and expected regret at most $T\epsilon/6$, then there is an algorithm to distinguish whether a coin is ϵ-heads-biased or ϵ-tails-biased with probability at least $2/3$ using $18B$ coin flips.

This claim implies the theorem because it is known that distinguishing these coins requires $\Omega\left(1/\epsilon^2\right)$ coin flips; in other words, it implies that $\epsilon \geq \Omega\left(1/\sqrt{B}\right)$, so the algorithm's expected regret must be $\Omega\left(T/\sqrt{B}\right)$.

We prove Claim 1 by proving the following two claims:

Claim 2: If an algorithm's expected regret is at most $T\epsilon/6$, then under the ϵ-heads-biased coin, with probability at least $5/6$, it outputs the heads hypothesis more times than the tails hypothesis. (And symmetrically under the tails-biased coin.)

Claim 3: An algorithm in this coin setting with budget B can, with probability at least $5/6$, be simulated for T rounds using at most $18B$ coin flips – in the sense that its behavior is identical to its behavior on a full sequence of T coin flips.

Proof of Claim 1 from 2 and 3. We will take an algorithm with budget B and regret $T\epsilon$ and use it to distinguish the coin using $18B$ coin flips: Using Claim 3, we can simulate the algorithm's behavior for all T rounds using at most $18B$ coin flips, except with probability $1/6$. Then, if the algorithm used the hypothesis heads more times than tails, we guess that the coin is heads-biased, and symmetrically. By Claim 2, our guess is correct except with probability $1/6$. By a union bound, therefore, this procedure correctly distinguishes the coin except with probability $1/3$, proving Claim 1.

Proof of Claim 2. Suppose the coin being flipped is the heads-biased coin; everything that follows will hold symmetrically for the tails-biased coin. Now, suppose that the algorithm outputs the hypothesis tails for M of the T rounds. Since each round is an independent coin toss, if the hypothesis is tails then its expected loss on that round is $\frac{1}{2} + \epsilon$; if heads, $\frac{1}{2} - \epsilon$. This gives an expected loss of $M\left(\frac{1}{2} + \epsilon\right) + (T - M)\left(\frac{1}{2} - \epsilon\right) = \frac{T}{2} + (2M - T)\epsilon$.

Meanwhile, the expected loss of the optimal hypothesis is at most $T\left(\frac{1}{2} - \epsilon\right)$, since this is the expected loss of the heads hypothesis. Therefore, the algorithm's expected regret, if it outputs the hypothesis tails M times on average, is at least

$$\frac{T}{2} + (2\mathbb{E}M - T)\epsilon - T\left(\frac{1}{2} - \epsilon\right) = 2\mathbb{E}M\epsilon.$$

If the algorithm's regret is at most $T\epsilon/6$, then this implies that $2\mathbb{E}M\epsilon \leq T\epsilon/6$, or $\mathbb{E}M \leq T/12$. Thus by Markov's inequality, the probability that half or more of the hypotheses are tails is bounded by

$$\begin{aligned}\Pr[M \geq T/2] &\leq \frac{\mathbb{E}M}{T/2} \\ &\leq 1/6.\end{aligned}$$

.

Proof of Claim 3. Here, we assume that $\epsilon < 1/6$, or B is larger than a (relatively small) constant.

On each data point, there are four possible menus: whether to buy or not to buy if the point is a heads or is a tails.[2] If the menu is (don't buy, don't buy), then no coin flip is needed (the behavior of the algorithm is identical whether the coin is actually flipped or not). Otherwise, the coin must be flipped, but the algorithm buys the data point with probability at least $\frac{1}{2} - \epsilon \geq \frac{1}{3}$ (the lowest probability of the remaining three menus). Thus the expected number of flips needed before the budget is exhausted is at most $3B$, and by Markov's inequality, the probability that it exceeds $18B$ is at most $1/6$. □

Theorem A.4.6 (Theorem 6.5.3)**.** *No algorithm for the non-strategic online data-purchasing problem has expected regret better than $O\left(\gamma T/\sqrt{B}\right)$; that is, for every γ, for every algorithm, there*

[2]The algorithm may make this a randomized menu, but we can simply consider the outcome of that random menu.

exists a sequence with parameter γ on which its regret is $\Omega\left(\gamma\frac{T}{\sqrt{B}}\right)$. Similarly, for $\bar{c} = \frac{1}{T}\sum_t \sqrt{c}$ and $\mu = \frac{1}{T}\sum_t c_t$, we have the lower bounds $\Omega\left(T\bar{c}/\sqrt{B}\right)$ and $\Omega\left(T\mu/\sqrt{B}\right)$.

Proof. We reduce to the previous theorem. Consider the following distribution on input sequences. There are three possible data points: heads, tails, and "no coin". There are still two hypotheses, heads and tails. Both have loss 1 on the "no coin" data point.

Now fix any $\gamma \in [0, 1]$. We will first send $(1-\gamma)T$ data points, all of which are "no coin". The loss of either hypothesis on all of these points is 1, and the cost of these points is zero. Then, we will choose either the ϵ-heads-biased or ϵ-tails-biased coin, with $\epsilon = 1/\sqrt{B}$, and send $T' = \gamma T$ coin flips, just as in the previous proof.

Because the first $(1-\gamma)T$ points are irrelevant to the regret, the regret of any algorithm is simply its regret on these final T' data points, which by the previous proof is at least on the order of $T'\epsilon = T'/\sqrt{B} = \gamma T/\sqrt{B}$.

Now to check that the parameter γ chosen above really is the γ value of the data sequence, note that the convexified hypothesis space for this problem is the space of distributions $p \in \mathbb{R}^2$ on $\{heads, tails\}$, with loss $1 - p \cdot (1, 0)$ if the coin is heads or $1 - p \cdot (0, 1)$ if the coin is tails. The gradient of the loss on either point for all p is $(1, 0)$ or $(0, 1)$ respectively, and both have norm 1. So $\Delta_{h_t, f_t} = 1$ for all "heads" and "tails" data points. Thus we have that $\frac{1}{T}\sum_t \Delta_{h_t,f_t}\sqrt{c_t} = \frac{T'}{T} = \gamma$.

Finally, noting that $\gamma = \bar{c}$ in this case gives the bound containing $\bar{c}$. For the lower bound with μ, take the exact construction in Theorem 6.5.2 and let each point have $c_t = \mu$ instead of $c_t = 1$. □

A.4.3 No regret — main setting

Theorem A.4.7 (Theorem 6.3.4)**.** *If Mechanism 6 is run with prior knowledge of γ and of γ_{max} (up to a constant factor), then it can choose K and η to satisfy the expected budget constraint and obtain a regret bound of*

$$O\left(\max\left\{\frac{T}{\sqrt{B}}g\ ,\ \sqrt{T}\right\}\right),$$

where $g = \sqrt{\gamma \cdot \gamma_{max}}$ (by setting $K = \frac{T}{B}\gamma_{max}$). Similarly, knowledge only of γ, respectively $\bar{c} = \frac{1}{T}\sum_t \sqrt{c_t}$, respectively $\mu = \frac{1}{T}\sum_t c_t$ suffices for the regret bound with $g = \sqrt{\gamma}$, respectively

$g = \sqrt{\bar{c}}$, *respectively* $g = \mu^{1/4}$.

Proof. The proof will proceed by finding a close-to-optimal value K of the normalizing constant by considering the budget constraint, then plugging this into the regret term to get a bound. The constant maximum price plays into this proof in a slightly non-obvious way. Because of this, instead of setting this maximum price equal to 1, we consider the generalization where costs may lie in $[0, c_{max}]$.

Consider time t when (c_t, f_t) arrives. Recall that the approach at time t is to draw a price for f_t from the distribution where

$$A_t(c) = \Pr[\text{price} \geq c] = \min\left\{1\ ,\ \frac{\Delta_{h_t,f_t}}{K\sqrt{c}}\right\}.$$

Consider then the induced posted-price distribution, which is pictured in Figure 6.2. It has a point mass at c_{max} of probability[3] $\Delta_{h_t,f_t}/K\sqrt{c_{max}}$. Otherwise, it is continuous on the interval $[c^*, c_{max}]$ with density

$$-A_t'(\pi) = \frac{\Delta_{h_t,f_t}}{2K\pi^{3/2}},$$

and the lower endpoint c^* satisfies $A_t(c^*) = 1$, *i.e.* $c^* = \Delta_{h_t,f_t}^2/K^2$.

We first find the bound on K such that the expected budget constraint is satisfied. The expected amount spent on arrival t can be computed as follows.

$$\begin{aligned}
&c_{max}\Pr[\text{price } = c_{max}] \ + \int_{\max\{c_t,c^*\}}^{c_{max}} x\,(\text{pdf at } x)\,dx \\
&= c_{max}\frac{\Delta_{h_t,f_t}}{K\sqrt{c_{max}}} + \int_{\max\{c_t,c^*\}}^{c_{max}} x\frac{\Delta_{h_t,f_t}}{2Kx^{3/2}}dx \\
&= \frac{\Delta_{h_t,f_t}}{K}\left(\sqrt{c_{max}} + \int_{\max\{c_t,c^*\}}^{c_{max}} \frac{1}{2\sqrt{x}}dx\right) \\
&= \frac{\Delta_{h_t,f_t}}{K}\left(2\sqrt{c_{max}} - \sqrt{\max\{c_t, c^*\}}\right).
\end{aligned}$$

Now let c_t^* be the value of c^* for arrival t (to distinguish its value in different timesteps). By the budget constraint, we need to pick K so that

$$\sum_t \mathbb{E}\left[\text{spend on arrival } (c_t, f_t)\right] \leq B,$$

[3]If this quantity is greater than 1, then we post a price of c_{max} for this datapoint, and what follows will only be a looser upper bound on the amount spent.

so

$$\sum_t \frac{\Delta_{h_t,f_t}}{K}\left(2\sqrt{c_{max}} - \sqrt{\max\{c_t, c^*\}}\right) \leq B.$$

Now we make a simplification: If we substitute c_t in for $\max\{c_t, c^*\}$, then the left-hand side only increases. Thus, to satisfy the previous inequality, it suffices to choose K to satisfy

$$\sum_t \frac{\Delta_{h_t,f_t}}{K}\left(2\sqrt{c_{max}} - \sqrt{c_t}\right) \leq B.$$

Thus, we let

$$K_{min} = \frac{1}{B}\sum_t \Delta_{h_t,f_t}\left(2\sqrt{c_{max}} - \sqrt{c_t}\right).$$

Note that it also suffices to choose any normalizer greater than K.

Recall our definition of the "difficulty-of-the-input" parameter

$$\gamma = \mathbb{E}\frac{1}{T}\sum_t \Delta_{h_t,f_t}\sqrt{c_t},$$

and let

$$\gamma_{max} = \mathbb{E}\frac{1}{T}\sum_t \Delta_{h_t,f_t}\sqrt{c_{max}}.$$

Then we have

$$K_{min} = \frac{T}{B}\left(2\gamma_{max} - \gamma\right).$$

We now have the setup to quickly derive bounds such as the theorem statements. Note that any choice of $K \geq K_{min}$ satisfies the expected budget constraint.

For the first regret bound, suppose that we know both γ and γ_{max} up to a constant factor. Then we can set $K = O(K_{min})$. By Lemma 6.3.1, the expected regret is bounded by

$$Regret \leq \frac{\beta}{\eta} + \eta\sum_t \frac{\Delta^2_{h_t,f_t}}{A_t(c_t)}$$

where β is a constant and η will be chosen later.

As in the known-costs scenario, let us split into those arrivals that we purchase with probability 1 (this corresponds to $c_t < c_t^*$) and the others, letting $S = \{t : A_t(c_t) < 1\}$. Then the summation term in the regret bound is bounded by a constant times

$$\begin{aligned}&\sum_{t\notin S}\Delta^2_{h_t,f_t} \;+\; \sum_{t\in S}\Delta_{h_t,f_t}\sqrt{c_t}K \\ &\leq T \;+\; \frac{T^2}{B}\gamma\left(2\gamma_{max} - \gamma\right) \qquad \text{(A.29)}\end{aligned}$$

where we have used the Lipschitz assumption on the loss function $\Delta_{h_t,f_t} \leq 1$.

As $\gamma_{max} \geq \gamma$, we do not lose much by taking the upper bound

$$M = T \ + \ 2\frac{T^2}{B}\gamma \cdot \gamma_{max}. \tag{A.30}$$

Now we can choose $\eta = \Theta(1/M)$ and obtain our regret bound of

$$\begin{aligned} Regret &\leq O\left(\sqrt{M}\right) \\ &\leq O\left(\max\left\{\sqrt{T}\,,\ \frac{T}{\sqrt{B}}\sqrt{\gamma \cdot \gamma_{max}}.\right\}\right). \end{aligned}$$

The other regret bounds all follow by (1) upper-bounding $\gamma_{max} \leq \sqrt{c_{max}}$; (2) letting $K = \frac{T}{B}\sqrt{c_{max}}$; (3) upper-bounding γ; and (4) setting η appropriately. Note that this can only increase K, so the expected budget constraint is still satisfied. The modifications simply give a different bound in Expression A.30, from which the rest of the argument follows analogously.

From (1) and (2), Expression A.30 becomes

$$M = T \ + \ 2\frac{T^2}{B}\gamma\sqrt{c_{max}}.$$

First, if we know γ, then picking $\eta = \Theta(1/M)$ gives the corresponding bound.

Second, with only knowledge of $\bar{c} = \frac{1}{T}\sum_t \sqrt{c_t}$, observe that $\gamma \leq O(\bar{c})$ and plug in.

Third, observe that by Jensen's inequality $\bar{c} \leq \sqrt{\mu}$ (where $\mu = \frac{1}{T}\sum_t c_t$) and plug in. □

BIBLIOGRAPHY

[1] J. Abernethy, Y. Chen, C.-J. Ho, and B. Waggoner. Low-cost learning via active data procurement. In *16th ACM Conf. on Economics and Computation (EC)*, 2015.

[2] R. Agrawal. The continuum-armed bandit problem. *SIAM J. Control and Optimization*, 33 (6):1926–1951, 1995.

[3] S. Agrawal and N. R. Devanur. Bandits with concave rewards and convex knapsacks. In *15th ACM Conf. on Economics and Computation (EC)*, 2014.

[4] N. Alon, B. Awerbuch, Y. Azar, N. Buchbinder, and J. S. Naor. A general approach to online network optimization problems. In *15th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, 2004.

[5] O. Alonso. Implementing crowdsourcing-based relevance experimentation: an industrial perspective. *Inf. Retr.*, 16(2):101–120, 2013.

[6] R. M. Araujo. 99designs: An analysis of creative competition in crowdsourced design. In *1st Conf. on Human Computation and Crowdsourcing (HCOMP)*, 2013.

[7] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002. Preliminary version in *15th ICML*, 1998.

[8] P. Auer, R. Ortner, and C. Szepesvári. Improved Rates for the Stochastic Continuum-Armed Bandit Problem. In *20th Conf. on Learning Theory (COLT)*, pages 454–468, 2007.

[9] A. Badanidiyuru, R. Kleinberg, and Y. Singer. Learning on a budget: posted price mechanisms for online procurement. In *13th ACM Conf. on Electronic Commerce (EC)*, pages 128–145, 2012.

[10] A. Badanidiyuru, R. Kleinberg, and A. Slivkins. Bandits with knapsacks. In *54th IEEE Symp. on Foundations of Computer Science (FOCS)*, 2013.

[11] M.-F. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. In *23rd Intl. Conf. on Machine Learning (ICML)*, 2006.

[12] N. Bansal, N. Buchbinder, and J. S. Naor. A primal-dual randomized algorithm for weighted paging. In *48th IEEE Symp. on Foundations of Computer Science (FOCS)*, 2007.

[13] J. E. Berg, R. Forsythe, F. D. Nelson, and T. A. Rietz. Results from a dozen years of election futures markets research. *Handbook of Experimental Economic Results*, 2001.

[14] A. Beygelzimer, D. Hsu, J. Langford, and T. Zhang. In *24th Advances in Neural Information Processing Systems (NIPS)*.

[15] A. Beygelzimer, S. Dasgupta, and J. Langford. Importance weighted active learning. In *26th Intl. Conf. on Machine Learning (ICML)*, 2009.

[16] S. E. Bonner, R. Hastie, S. G. B., and S. M. Young. A review of the effects of financial incentives on performance in laboratory tasks: Implications for management accounting. *Journal of Management Accounting Research*, 12(1):19–64, 2000.

[17] S. Bubeck, R. Munos, G. Stoltz, and C. Szepesvari. Online Optimization in X-Armed Bandits. *J. of Machine Learning Research (JMLR)*, 12:1587–1627, 2011. Preliminary version in *NIPS 2008*.

[18] N. Buchbinder and J. S. Naor. Online primal-dual algorithms for covering and packing problems. In *13th Annual European Symp. on Algorithms (ESA)*, 2005.

[19] N. Buchbinder, K. Jain, and J. S. Naor. Online primal-dual algorithms for maximizing ad-auctions revenue. In *15th Annual European Symp. on Algorithms (ESA)*, 2007.

[20] M. Buhrmester, T. Kwang, and S. D. Gosling. Amazon's Mechanical Turk: A new source of inexpensive, yet high-quality, data? *Perspectives on Psychological Science*, 2011.

[21] A. D. Bull. Adaptive-treed bandits. Technical Report 1302.2489, arxiv.org, 2013.

[22] C. F. Camerer and R. Hogarth. The effects of financial incentives in economics experiments: A review and capital-labor-production framework. *Journal of Risk and Uncertainty*, 19(1): 7–42, 1999.

[23] J. Cameron and W. D. Pierce. Reinforcement, reward, and intrinsic motivation: A meta-analysis. *Review of Educational Research*, 64(3):363–423, 1994.

[24] N. Cesa-Bianchi and G. Lugosi. Prediction, learning, and games. 2006.

[25] N. Cesa-Bianchi, A. Conconi, and C. Gentile. On the generalization ability of on-line learning algorithms. *Information Theory, IEEE Transactions on*, 50(9):2050–2057, 2004.

[26] J. Chandler, P. Mueller, and G. Paolacci. Nonnaïveté among Amazon Mechanical Turk workers: Consequences and solutions for behavioral researchers. *Behavior Research Methods*,

46(1):112–130, 2014.

[27] S. Cooper, F. Khatib, A. Treuille, J. Barbero, J. Lee, M. Beenen, A. Leaver-Fay, D. Baker, Z. Popović, and F. Players. Predicting protein structures with a multiplayer online game. *Nature*, 2010.

[28] K. Crammer, M. Kearns, and J. Wortman. Learning from data of variable quality. In *19th Advances in Neural Information Processing Systems (NIPS)*, 2005.

[29] K. Crammer, M. Kearns, and J. Wortman. Learning from multiple sources. *Journal of Machine Learning Research*, 9:1757–1774, 2008.

[30] R. Cummings, K. Ligett, A. Roth, Z. S. Wu, and J. Ziani. Accuracy for sale: Aggregating data with a variance constraint. In *6th Innovations in Theoretical Computer Science Conf. (ITCS)*, pages 317–324. ACM, 2015.

[31] P. Dawid and A. Skene. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Journal of the Royal Statistical Society, Series C (Applied Statistics)*, 1979.

[32] O. Dekel and O. Shamir. Vox populi: Collecting high-quality labels from a crowd. In *22nd Conf. on Learning Theory (COLT)*, 2009.

[33] O. Dekel, F. Fischer, and A. D. Procaccia. Incentive compatible regression learning. In *19th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, 2008.

[34] C. Dellarocas. Reputation mechanism design in online trading environments with pure moral hazard. *Information Systems Research*, 16(2):209–230, 2005.

[35] C. Dellarocas. How often should reputation mechanisms update a trader's reputation profile? *Information Systems Research*, 17(3):271–285, 2006.

[36] N. Devanur and J. Hartline. Limited and online supply and the bayesian foundations of prior-free mechanism design. In *10th ACM Conf. on Electronic Commerce (EC)*, 2009.

[37] N. R. Devanur and T. P. Hayes. The AdWords problem: Online keyword matching with budgeted bidders under random permutations. In *10th ACM Conf. on Electronic Commerce (EC)*, pages 71–78, 2009.

[38] N. R. Devanur, K. Jain, B. Sivan, and C. A. Wilkens. Near optimal online algorithms and fast approximation algorithms for resource allocation problems. In *12nd ACM Conf. on Electronic Commerce (EC)*, 2011.

[39] M. Dudik, D. Hsu, S. Kale, N. Karampatziakis, J. Langford, L. Reyzin, and T. Zhang. Efficient optimal leanring for contextual bandits. In *27th Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2011.

[40] D. Easley and A. Ghosh. Behavioral mechanism design: Optimal crowdsourcing contracts and prospect theory. In *16th ACM Conf. on Electronic Commerce (EC)*, 2015.

[41] R. Eisenberger and J. Cameron. Detrimental effects of reward: Reality or myth? *American psychologist*, 51(11):1153, 1996.

[42] G. Ellison. Cooperation in the prisoner's dilemma with anonymous random matching. *Review of Economic Studies*, 61(3):567–588, 1994.

[43] B. S. Frey and F. Oberholzer-Gee. The cost of price incentives: An empirical analysis of motivation crowding-out. *The American Economic Review*, pages 746–755, 1997.

[44] E. Friedman, P. Resnick, and R. Sami. Manipulation-resistant reputation systems. Cambridge University Press, 2007.

[45] E. J. Friedman and P. Resnick. The social cost of cheap pseudonyms. *Journal of Economics and Management Strategy*, 10(2):173–199, 2001.

[46] A. Garivier and O. Cappé. The KL-UCB Algorithm for Bounded Stochastic Bandits and Beyond. In *24th Conf. on Learning Theory (COLT)*, 2011.

[47] A. Ghosh and P. Hummel. A game-theoretic analysis of rank-order mechanisms for user-generated content. In *12th ACM Conf. on Electronic Commerce (EC)*, 2011.

[48] A. Ghosh and P. Hummel. Learning and incentives in user-generated content: Multi-armed bandits with endogenous arms. In *7th Innovations in Theoretical Computer Science Conf. (ITCS)*, 2013.

[49] A. Ghosh and R. Kleinberg. Optimal contest design for simple agents. In *15th ACM Conf. on Electronic Commerce (EC)*, 2014.

[50] A. Ghosh and P. McAfee. Incentivizing high-quality user-generated content. In *20th Intl. World Wide Web Conf. (WWW)*, 2011.

[51] A. Ghosh and A. Roth. Selling privacy at auction. In *12th ACM Conf. on Electronic Commerce (EC)*, 2011.

[52] A. Ghosh, S. Kale, and P. McAfee. Who moderates the moderators? Crowdsourcing abuse

detection in user-generated content. In *12th ACM Conf. on Electronic Commerce (EC)*, 2011.

[53] A. Ghosh, K. Ligett, A. Roth, and G. Schoenebeck. Buying private data without verification. In *15th ACM Conf. on Economics and Computation (EC)*. ACM, 2014.

[54] D. Gilchrist, M. Luca, and D. Malhotra. When 3+1 ¿ 4: Gift structure and reciprocity in the field. Technical report, 2014. Working Paper.

[55] U. Gneezy and A. Rustichini. Pay enough or don't pay at all. *Quarterly Journal of Economics, August*, pages 791–810, 2000.

[56] S. Hanneke. *Theoretical foundations of active learning*. ProQuest, 2009.

[57] C. G. Harris. You're hired! an examination of crowdsourcing incentive models in human resource tasks. In *CSDM*, 2011.

[58] R. Hertwig and A. Ortmann. Experimental practices in economics: A methodological challenge for psychologists? *Behavioral and Brain Sciences*, 24(3):383–403, 2001.

[59] C.-J. Ho and J. W. Vaughan. Online task assignment in crowdsourcing markets. In *26th AAAI Conference on Artificial Intelligence (AAAI)*, 2012.

[60] C.-J. Ho, Y. Zhang, J. W. Vaughan, and M. van der Schaar. Towards social norm design for crowdsourcing markets. In *4th Human Computation Workshop (HCOMP)*, 2012.

[61] C.-J. Ho, S. Jabbari, and J. W. Vaughan. Adaptive task assignment for crowdsourced classification. In *30th Intl. Conf. on Machine Learning (ICML)*, 2013.

[62] C.-J. Ho, A. Slivkins, and J. W. Vaughan. Adaptive contract design for crowdsourcing markets: Bandit algorithms for repeated principal-agent problems. In *15th ACM Conf. on Electronic Commerce (EC)*, 2014.

[63] C.-J. Ho, A. Slivkins, S. Suri, and J. W. Vaughan. Incentivizing high quality crowdwork. In *24th Intl. World Wide Web Conf. (WWW)*, 2015.

[64] T. Horel, S. Ioannidis, and M. Muthukrishnan. Budget feasible mechanisms for experimental design. In *Latin American Theoretical Informatics (LATIN-14)*, 2014.

[65] J. J. Horton and L. B. Chilton. The labor economics of paid crowdsourcing. In *11th ACM Conf. on Electronic Commerce (EC)*, 2010.

[66] J. J. Horton, D. Rand, and R. Zeckhauser. The online laboratory: Conducting experiments in a real labor market. *Experimental Economics*, 14(3):399–425, 2011.

[67] P. G. Ipeirotis. Analyzing the Amazon Mechanical Turk marketplace. *ACM XRDS*, 17(2): 16–21, 2010.

[68] P. G. Ipeirotis, F. Provost, and J. Wang. Quality management on Amazon Mechanical Turk. In *2nd Human Computation Workshop (HCOMP)*, 2010.

[69] S. Jain, Y. Chen, and D. C. Parkes. Designing incentives for online question and answer forums. In *10th ACM Conf. on Electronic Commerce (EC)*, 2009.

[70] S. Jain, Y. Chen, and D. Parkes. Designing incentives for online question-and-answer forums. *Games and Economic Behavior*, 2012.

[71] G. D. Jenkins Jr., N. Gupta, A. Gupta, and J. D. Shaw. Are financial incentives related to performance? A meta-analytic review of empirical research. *Journal of Applied Psychology*, 83(5):777–787, 1998.

[72] M. Kandori. Social norms and community enforcement. *Review of Economic Studies*, 59(1): 63–80, 1992.

[73] D. R. Karger, S. Oh, and D. Shah. Iterative learning for reliable crowdsourcing systems. In *25th Advances in Neural Information Processing Systems (NIPS)*, 2011.

[74] I. A. Kash, E. J. Friedman, and J. Y. Halpern. Optimizing scrip systems: Efficiency, crashes, hoarders, and altruists. In *8th ACM Conf. on Electronic Commerce (EC)*, 2007.

[75] I. A. Kash, E. J. Friedman, and J. Y. Halpern. Manipulating scrip systems: Sybils and collusion. In *AMMA*, 2009.

[76] R. H. Keshavan, A. Montanari, and S. Oh. Matrix completion from noisy entries. In *23rd Advances in Neural Information Processing Systems (NIPS)*, 2009.

[77] A. Kittur, E. H. Chi, and B. Suh. Crowdsourcing user studies with Mechanical Turk. In *Conf. on Human Factors in Computing Systems (CHI)*, 2008.

[78] R. Kleinberg. Nearly tight bounds for the continuum-armed bandit problem. In *18th Advances in Neural Information Processing Systems (NIPS)*, 2004.

[79] R. Kleinberg and T. Leighton. The value of knowing a demand curve: Bounds on regret for online posted-price auctions. In *44th IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 594–605, 2003.

[80] R. Kleinberg, A. Slivkins, and E. Upfal. Multi-Armed Bandits in Metric Spaces. In *40th*

ACM Symp. on Theory of Computing (STOC), pages 681–690, 2008.

[81] L. Kocsis and C. Szepesvari. Bandit Based Monte-Carlo Planning. In *17th European Conf. on Machine Learning (ECML)*, pages 282–293, 2006.

[82] J.-J. Laffont and D. Martimort. *The Theory of Incentives: The Principal-Agent Model*. Princeton University Press, 2002.

[83] E. P. Lazear. Performance pay and productivity. *American Economic Review*, pages 1346–1361, 2000.

[84] K. Ligett and A. Roth. Take it or leave it: Running a survey when privacy comes at a cost. In *8th Workshop on Internet & Network Economics (WINE)*, 2012.

[85] L. Litman, J. Robinson, and C. Rosenzweig. The relationship between motivation, monetary compensation, and data quality among US- and India-based workers on Mechanical Turk. *Behavioral Research Methods*, 2014.

[86] Q. Liu, J. Peng, and A. Ihler. Variational inference for crowdsourcing. In *26th Advances in Neural Information Processing Systems (NIPS)*, 2012.

[87] U.-V. Marti and H. Bunke. The iam-database: an english sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition*, 5(1): 39–46, 2002.

[88] W. Mason and S. Suri. Conducting behavioral research on Amazon's Mechanical Turk. *Behavior Research Methods*, 2012.

[89] W. Mason and D. Watts. Financial incentives and the "performance of crowds". In *1st Human Computation Workshop (HCOMP)*, 2009.

[90] H. Masum and Y.-C. Zhang. Manifesto for the reputation society. *First Monday*, 9(7), 2004.

[91] R. Meir, A. D. Procaccia, and J. S. Rosenschein. Algorithms for strategyproof classification. *Artificial Intelligence*, 186:123–156, 2012.

[92] R. Munos and P.-A. Coquelin. Bandit algorithms for tree search. In *23rd Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2007.

[93] M. Okuno-Fujiwara and A. Postlewaite. Social norms and random matching games. *Games and Economic Behavior*, 9(1):79—109, 1995.

[94] D. Oleson, V. Hester, A. Sorokin, G. Laughlin, J. Le, and L. Biewald. Programmatic gold:

Targeted and scalable quality assurance in crowdsourcing. In *3rd Human Computation Workshop (HCOMP)*, 2011.

[95] S. Pandey, D. Agarwal, D. Chakrabarti, and V. Josifovski. Bandits for Taxonomies: A Model-based Approach. In *SIAM Intl. Conf. on Data Mining (SDM)*, 2007.

[96] P. Resnick, K. Kuwabara, R. Zeckhauser, and E. Friedman. Reputation systems. *Communications of the ACM*, 43:45–48, 2000.

[97] J. Rogstadius, V. Kostakos, A. Kittur, B. Smus, J. Laredo, and M. Vukovic. An assessment of intrinsic and extrinsic motivation on task performance in crowdsourcing markets. 2011.

[98] A. Roth and G. Schoenebeck. Conducting truthful surveys, cheaply. In *13th ACM Conf. on Electronic Commerce (EC)*, 2012.

[99] D. Schall. *Service-Oriented Crowdsourcing - Architecture, Protocols and Algorithms*. Springer Briefs in Computer Science. Springer, 2012.

[100] H. J. Seltman. *Experimental Design and Analysis*. 2012.

[101] S. Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 2012.

[102] A. D. Shaw, J. J. Horton, and D. L. Chen. Designing incentives for inexpert human raters. In *ACM Conference on Computer Supported Cooperative Work (CSCW)*, 2011.

[103] V. Sheng, F. Provost, and P. Ipeirotis. Get another label? Improving data quality using multiple, noisy labelers. In *ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining (KDD)*, 2008.

[104] Y. Singer and M. Mittal. Pricing mechanisms in crowdsourcing markets. In *22nd Intl. World Wide Web Conf. (WWW)*, 2013.

[105] A. Singla and A. Krause. Truthful incentives in crowdsourcing tasks using regret minimization mechanisms. In *22nd Intl. World Wide Web Conf. (WWW)*, pages 1167–1178, 2013.

[106] A. Slivkins. Multi-armed bandits on implicit metric spaces. In *25th Advances in Neural Information Processing Systems (NIPS)*, 2011.

[107] A. Slivkins. Contextual Bandits with Similarity Information. In *24th Conf. on Learning Theory (COLT)*, 2011. To appear in J. of Machine Learning Research (JMLR), 2014.

[108] W. R. Thompson. On the likelihood that one unknown probability exceeds another in view

of the evidence of two samples. *Biometrika*, 25(3-4), 1933.

[109] L. Tran-Thanh, S. Stein, A. Rogers, and N. R. Jennings. Efficient crowdsourcing of unknown experts using multi-armed bandits. In *20th European Conf. on Artificial Intelligence (ECAI)*, pages 768–773, 2012.

[110] L. von Ahn and L. Dabbish. Labeling images with a computer game. In *Conf. on Human Factors in Computing Systems (CHI)*, 2004.

[111] L. von Ahn and L. Dabbish. Designing games with a purpose. *Communications of the ACM*, 51(8):58–67, 2008.

[112] L. von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum. reCAPTCHA: Human-based character recognition via web security measures. *Science*, 2008.

[113] 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.

[114] P. Wais, S. Lingamneni, D. Cook, J. Fennell, B. Goldenberg, D. Lubarov, and D. Martin. Towards building a high-quality workforce with mechanical turk. Presented at the *NIPS Workshop on Computational Social Science and the Wisdom of Crowds*, 2010.

[115] P. Welinder, S. Branson, S. Belongie, and P. Pietro. The Multidimensional Wisdom of Crowds. In *24th Advances in Neural Information Processing Systems (NIPS)*, 2010.

[116] M. Yin, Y. Chen, and Y.-A. Sun. The effects of performance-contingent financial incentives in online labor markets. In *27th AAAI Conference on Artificial Intelligence (AAAI)*, 2013.

[117] M. Yin, Y. Chen, and Y.-A. Sun. Monetary interventions in crowdsourcing task switching. In *2nd Conf. on Human Computation and Crowdsourcing (HCOMP)*, 2014.

[118] Y. Zhang and M. van der Schaar. Peer-to-peer multimedia sharing based on social norms. *Elsevier Journal Signal Processing: Image Communication*, 2012.

[119] Y. Zhang, J. Park, and M. van der Schaar. Social norms for networked communities, 2011. Preprint available at `http://arxiv.org/abs/1101.0272`.

[120] D. Zhou, S. Basu, Y. Mao, and J. Platt. Learning from the wisdom of crowds by minimax entropy. In *26th Advances in Neural Information Processing Systems (NIPS)*, 2012.