POSITION PAPER

Online Decision Making in Crowdsourcing Markets: Theoretical Challenges

ALEKSANDRS SLIVKINS
Microsoft Research
and
JENNIFER WORTMAN VAUGHAN
Microsoft Research

Over the past decade, crowdsourcing has emerged as a cheap and efficient method of obtaining solutions to simple tasks that are difficult for computers to solve but possible for humans. The popularity and promise of crowdsourcing markets has led to both empirical and theoretical research on the design of algorithms to optimize various aspects of these markets, such as the pricing and assignment of tasks. Much of the existing theoretical work on crowdsourcing markets has focused on problems that fall into the broad category of online decision making; task requesters or the crowdsourcing platform itself make repeated decisions about prices to set, workers to filter out, problems to assign to specific workers, or other things. Often these decisions are complex, requiring algorithms that learn about the distribution of available tasks or workers over time and take into account the strategic (or sometimes irrational) behavior of workers.

As human computation grows into its own field, the time is ripe to address these challenges in a principled way. However, it appears very difficult to capture all pertinent aspects of crowdsourcing markets in a single coherent model. In this paper, we reflect on the modeling issues that inhibit theoretical research on online decision making for crowdsourcing, and identify some steps forward. This paper grew out of the authors' own frustration with these issues, and we hope it will encourage the community to attempt to understand, debate, and ultimately address them.

---

1. INTRODUCTION

Crowdsourcing markets have emerged as a tool for bringing together *requesters*, who have tasks they need accomplished, and *workers*, who are willing to perform these tasks in a timely manner in exchange for payment. Some crowdsourcing platforms, such as Amazon Mechanical Turk, focus on small "microtasks" such as labeling an image or filling out a survey, with payments on the order of ten

---

cents, while other platforms, such as oDesk, focus on larger jobs like designing websites, for significantly larger payments, but most share the common feature of *repeated interaction*. In these markets, workers, requesters, and the platform itself can all adjust their behavior over time to adapt to the environment. For example, requesters can choose which workers to target (or filter out) for their tasks and which prices to offer, and can update these choices as they learn more about salient features of the environment, such as workers' skill levels, the difficulty of their tasks, and workers' willingness to accept their tasks at given prices. In principle, the platform could also make smarter decisions over time as it learns more about the quality of both requesters and workers.

Naturally, as crowdsourcing has gained popularity and human computation has grown into its own field, researchers have taken an interest in modeling and analyzing the problem of online decision making in crowdsourcing markets. There is plenty of prior work on which to build. Online decision algorithms have a rich literature in operations research, economics, and several areas of computer science including machine learning, theory of algorithms, artificial intelligence, and algorithmic mechanism design. A large portion of this literature is concerned with the tradeoff between *exploration* (obtaining new information, perhaps by sacrificing near-term gains) and *exploitation* (making optimal decisions based on the currently available information). This tradeoff is often studied under the name *multi-armed bandits*; a reader can refer to Cesa-Bianchi and Lugosi [2006], Bergemann and Välimäki [2006], Gittins et al. [2011], and Bubeck and Cesa-Bianchi [2012] for background and various perspectives on the problem space. Another (somewhat overlapping) stream of work concerns *dynamic pricing* and, more generally, *revenue management* problems; see Besbes and Zeevi [2009] for an overview of this work in operations research, and Babaioff et al. [2012] for a theory-of-algorithms perspective. Further, there is extensive literature on online decision problems in which all information pertinent to a given round is revealed, either before or after the algorithm makes its decision for this round; see Borodin and El-Yaniv [1998], Buchbinder and Naor [2009], and Cesa-Bianchi and Lugosi [2006] for background.

Despite the vast scope of this existing work, crowdsourcing brings an array of domain-specific challenges that require novel solutions. To address these challenges in a principled way, one would like to formulate a unified collection of well-defined algorithmic questions with well-specified objectives, allowing researchers to propose novel solutions and techniques that can be easily compared, leading to a deeper understanding of the underlying issues. However, it appears very difficult to capture all of the pertinent aspects of crowdsourcing in a coherent model. As a result, many of the existing theoretical papers on crowdsourcing propose their own new models. This makes it difficult to compare techniques across papers, and leads to uncertainty about which parameters or features matter most when designing new platforms or algorithms.

There are several reasons why developing a unified model of crowdsourcing is difficult. First, there is a tension between the desire to study models based on existing platforms, such as Mechanical Turk, which would allow algorithms to be implemented and tested immediately, and the desire to look ahead and model alternative platforms with novel features that could potentially lead to improved

crowdsourcing markets further down the line. A plethora of different platform designs are possible, and different designs lead to both different models and different algorithmic questions.

Second, even after a model of the platform has been determined, one must take into account the diversity of tasks and workers that use the platform. Some workers may be better at some tasks than others, and may find some tasks harder than others. It is natural to take this diversity into account when designing algorithms (especially for a problem like task assignment), but the way in which this diversity is modeled may impact the choice of algorithm and the theoretical guarantees that are attainable.

Third, crowdsourcing workers are human beings, not algorithmic agents (with the possible exception of spambots, which bring their own problems). They may act strategically, maximizing their own welfare in response to the incentives provided by the requesters and the platform. Taking into account this strategic behavior is essential. Moreover, workers also may act in ways that are even less predictable and seemingly irrational. As just one example, empirical studies have shown that workers may not know their own costs of completing a task, and are susceptible to the *anchoring effect* in which they judge the value of a task based on the first price they see (see Mason and Watts [2009], Mason and Suri [2012], Yin et al. [2013] and the literature reviews therein). One must be aware of these nuances when modeling worker behavior.

This paper offers a detailed reflection on the modeling issues that inhibit theoretical research on repeated decision making in crowdsourcing. Our goal is not to offer a single unified model, but to raise awareness of the issues and spark a discussion in the community about the best ways to move forward. To this end, we identify a multitude of modeling choices that exist, and describe a few specific models that appear promising in that they potentially capture salient aspects of the problem.

A note on our scope. There have been some recent empirical or applied research projects aimed at developing online decision making algorithms that work well in practice on existing crowdsourcing platforms, primarily for the problems of task assignment and label aggregation (see, for example, Chen et al. [2013], Ipeirotis et al. [2013], and Wang et al. [2013]). While it is of course valuable for anyone interested in theoretical models of crowdsourcing to be aware of related empirical advances, we do not attempt to provide a complete survey of this work here.

A similar attempt has been made to categorize future directions of crowdsourcing research within the human-computer interaction and computer-supportive cooperative work communities [Kittur et al. 2013]. While their paper discusses many of the same broad problems as ours (such as quality control, platform design, and reputation), the research agenda they lay out is heavily focused on the high-level design of the technical and organizational mechanisms surrounding future crowd work systems, while we focus more specifically on low-level modeling and algorithmic challenges for online decision making.

2. INFORMAL PROBLEM STATEMENT

Let us start with an informal and very general description of the class of problems that fall under the umbrella of online decision making in crowdsourcing markets.

There are three parties: workers, requesters, and the crowdsourcing platform. Over time, requesters submit tasks to the platform. Workers get matched with requesters, perform tasks, and receive payments. Workers and requesters may arrive to the platform and leave over time. Some workers may be better than others at some tasks; some tasks may be more difficult than others for some workers. Some workers may enjoy some tasks more than others. All parties make **repeated decisions** over time. All parties can **learn over time**, which may help them in their decision making. All decision makers receive **partial feedback**: they see the consequences of their decisions, but typically they do not know what would have happened if they had made different decisions. Workers and requesters can **behave strategically**, so their incentives need to be taken into account. The problem is to design algorithms for decision making, on behalf of the platform, the requesters, or perhaps even the workers.

We can think about this setting from the point of view of each of the three parties, who face their own choices and have differing motivations and goals:

—Requesters can choose the maximal price they are willing to offer for a given task, and specify any budget constraints that they have. They may also be able to choose which prices to offer to given (categories of) workers for given (categories of) tasks. Further, they may be able to choose which (categories of) workers they are willing to interact with, and which instances of tasks to assign to each (category of) worker. The utility of the requester is typically (an increasing function of) the value that he obtains from completed work minus the price that he has to pay for this work.

—The platform may match (subsets of) requesters to (subsets of) workers. In principle, the platform may also be able to modify the offered prices, within constraints specified by the requesters, and to determine how to charge requesters and/or workers for their use of the platform. In the long run, the platform cares about maximizing long-term revenue. In the short term, the platform may care about keeping workers and requesters happy to attract more business, especially in the presence of competing platforms.

—Workers can decide whether to accept a task at a given price, and how much effort to put into it. They may be able to choose between (categories of) tasks, and may be asked to provide some information such as their interests, qualifications, and asking prices. Workers care about the amount of money that they earn, the cost of their labor (in terms of time or effort), and perhaps the amount of enjoyment they receive from completing tasks.

As parties interact over time, they may be able to learn or estimate workers' skill levels, how much workers value their effort and time, the difficulty of tasks, and the quality level of particular workers on particular tasks.

We focus on two natural versions of the problem: *requester-side* and *platform-side*.[1] The requester-side problem is to design a mechanism which makes repeated

_____

[1]In principle, one could also consider the decision making problems faced by workers. This area is still largely unexplored and could be a source of interesting research questions. However, we do not consider it here.

decisions for a specific requester. The platform-side problem is to design a mechanism which makes repeated decisions for the platform. Both versions should take into account realistic constraints and incentives of workers and requesters.

3. POSSIBLE MODELING CHOICES

To study and understand any particular aspect of repeated decision making for crowdsourcing markets, one must first determine a model or framework in which to work. To date, most theoretical papers on crowdsourcing have proposed their own models, which are often hard to compare, and with good reason; one could come up with any number of valid models, and each has its own appeal. However, we would argue that in order to extract and solve the fundamental issues that are present in any crowdsourcing market, one must be keenly aware of the (implicit or explicit) modeling choices they make, and the extent to which their algorithmic ideas and provable guarantees are sensitive to their modeling assumptions — essentially how *robust* results are to modifications in the model.

In this section, we identify a variety of modeling choices for repeated decision making in crowdsourcing, some of which have been discussed in prior work and some of which have not. We break these choices roughly into five categories: task design, platform design, quality of work, incentives and human factors, and performance objectives, though other categorizations are possible.

Task design. A wide variety of tasks can be crowdsourced. In many tasks, such as translation [Zaidan and Callison-Burch 2011], audio transcription [Lasecki et al. 2013], image/video analysis [VizWiz ; Noronha et al. 2011], or trip planning [Zhang et al. 2012], the workers' output is unstructured. On the other hand, in many applications currently deployed in the Internet industry, the workers' output has a very specific, simple structure such as a multiple-choice question or a free-form labeling question. In particular, most tasks that the assess relevance of web search results are structured as multiple-choice questions. The output structure (or lack thereof) determines whether and how workers' output can be aggregated. Below we survey some possibilities in more detail.

- *Multiple-choice questions.* The simplest structure is a binary question (*Is this a correct answer? Is this spam?*). Alternatively, a question can include more than two options (*Which of these web search results is more appropriate for a given query? Which of these categories is more appropriate for this website?*) or allow workers to submit more than one answer when appropriate. Further, the possible answers may be numeric or otherwise ordered (*What age group does this person belong to?*), in which case the similarity between the answers should be taken into account.

- *Free-form labeling.* In image labeling and various classification tasks, a worker may be allowed to submit arbitrary labels or categories. A worker may be allowed to submit more than one label for the same task. Some task designs may include a list of prohibited labels so as to obtain more diverse results.

- *Rankings and weights.* Some tasks strive to compute more complex structures such as rankings or relative weights for a given pool of alternatives [Pfeiffer et al. 2012; Mao et al. 2013]. Each worker may be asked to provide either a full ranking

or weight vector or only a comparison between a given pair of alternatives.

Platform design. All interaction between workers and requesters takes place in the context of a particular crowdsourcing platform. The platform designer controls the way in which requesters and workers may interact. In modeling crowdsourcing markets, one may wish to model the features of a specific existing platform (such as Amazon Mechanical Turk, oDesk, or TaskRabbit) or explore alternative platform designs that may lead to improved crowdsourcing markets in the future. In either case, one must consider the following modeling issues.

- *Selection of workers.* Does the platform allow requesters to limit their tasks to specific (categories of) workers, and if so, how? Possibilities range from coarse platform-defined categories of workers (e.g., filtering by demographic information or feedback rating/reputation) to allowing requesters to specify arbitrary workers by worker ID.

- *Selection of tasks.* How does the platform allow workers to select tasks? Does a worker see *all* offerings that are currently in the market? If so, how are they sorted? Alternatively, the platform could limit the options presented to a given worker. When the task(s) from a given requester are offered to a given worker, does the platform allow the requester to restrict which offerings from other requesters this worker is exposed to?

- *Feedback to the requester.* What is the granularity of feedback provided to a given requester by the platform? In particular, does the platform expose the workers' performance by individual IDs or only coarse categories? Does the platform expose some or all of the workers' attributes? Does the platform expose workers' performance for other requesters? If a worker views a task but ultimately does not decide to complete it, does the requester of that task learn that it was viewed?

- *Temporal granularity.* How often does the platform allow a requester to adjust its selection preferences? How frequently does the platform provide feedback to requesters? When is it reasonable to assume that workers complete a given task sequentially, one at a time, and when is it necessary for a model to allow multiple tasks to be accepted or completed simultaneously?

- *Price negotiation.* How do requesters negotiate prices with workers? Typically it is assumed that requesters post one take-it-or-leave-it price, but this is not the only possibility. Can a requester post different prices to different workers depending on the workers' attributes? Can the requester update the price over time? Alternatively, one could imagine a crowdsourcing platform on which workers communicate minimal asking prices to the requesters and prices are determined via an auction.

- *Payments.* How do payments happen? The simplest version is that each worker, upon completion of a task, receives the posted price for that task. Can a requester specify a payment that depends on the quality of submitted work? If so, must he declare the exact payment rule up front? In particular, must he commit to a specific way of determining the quality of submitted work?

- *Persistent identities.* Does a platform ensure persistent worker identities? How difficult or expensive is it for a worker to abandon his old identity and create a new

one, or to create multiple identities?

- *Platform-wide reputation.* Does the platform maintain a platform-wide reputation score for each worker and/or requester? If so, then how exactly is this score defined? What properties of this score are guaranteed? How are reputations used? How are workers and requesters prevented from manipulating their own reputations and the reputations of others?

- *Qualifying tests.* Does the platform allow requesters to require workers to pass a qualification test before being allowed to work for money?

The quality of work. In crowdsourcing environments, both tasks and workers are highly diverse. Naturally certain workers will be able to complete certain tasks more proficiently than others, yielding higher quality results, possibly for less effort. Although the difficulty of a task and the skills of each worker are generally not known to the platform or requester a priori, they can be learned and refined over time, though this is made more difficult when the quality of a completed task is not observable or easily measured, or when the users of the market are themselves changing over time.

- *Worker skills and task difficulty levels.* Some workers may be better than others at some tasks, and worse than others at other tasks. Ideally, an online algorithm should learn estimates of workers' skills for various tasks over time. However, if each worker has an arbitrary skill level for each task, it will be impossible to generalize based on observed performance. Therefore, it is desirable to model some type of structure on tasks or workers. Can the inherent skill level of a given worker be summarized by a single number? Can the inherent difficulty of a task be summarized by a single number? More generally, how should one model a given worker's proficiency at a given task?

- *Effort.* The quality of a worker's output may also depend on the amount of effort that the worker puts into the task. How can one model the effect of varying levels of effort on the quality of completed work?

- *Time dependence.* Workers' skills and the quality of their completed work may change over time, either because of learning (perhaps after completing the same task several times) or changes in effort level. Additionally, properties of the population of workers (such as the size of the population or their average quality) may change over time, for example based on time of day.

- *Measurable vs. unmeasurable quality.* Some tasks, such as labeling an image as containing a face or not, or answering a multiple-choice question, have objective measures of quality; either the provided answer is correct, or it is not. For other tasks, such as translating a paragraph from one language to another or drawing a picture, the quality of the work is more difficult to quantify. For tasks without objective measures of quality, it is necessary to clearly define criteria on which a task will be judged.

- *Observable vs. unobservable quality.* Even if the response to a task is objectively right or wrong, as is the case in the image labeling task above, the quality of submitted work may not be immediately observable; if the requester already knew

the correct label, he wouldn't need the worker to provide it. For many tasks of this form, it is possible for the requester to assign the same question to multiple workers and estimate the correct answer by consensus; if a statistically significant (weighted) majority agrees on some answer, it may be safe to assume it is correct.

• *Availability of gold standard tasks.* In some cases in which task quality is not observable, a requester may have access to a set of "gold standard" tasks for which the correct answer is known a priori. He can then use these tasks to bootstrap the process of learning workers' skill levels (possibly as part of a separate qualifying test, if allowed).

Incentives and other human factors. Perhaps the most difficult and controversial aspect of formalizing the repeated decision making problem in a crowdsourcing environment is modeling the incentives of participants. In the economics literature, it is standard to model agents as self-interested and rational utility maximizers. Empirical research studying the behavior of workers on Mechanical Turk suggests that there may be difficulties applying such models in crowdsourcing settings in which workers do not have a good sense of their own costs of completing a task and may choose particular tasks based on intangible qualities such as fun. This leads to another set of modeling issues.

• *Rationality.* Do workers behave rationally when choosing tasks to complete at particular prices? The standard model from economics is that each worker maximizes a simple well-defined notion of "utility," (an increasing function of) payment received minus production cost. However, workers may also take into account other factors, such as perceived value to society and personal enjoyment. The significance of these intangible factors in crowdsourcing environments is not well-understood.

• *Effort levels.* Can the workers strategically alter the amount of effort that they put into a given task, or is it assumed that workers complete each task to the best of their ability?

• *Costs.* How do the workers evaluate their internal costs for completing a given task or putting forth a given level of effort? Is this internal cost even a well-defined quantity? It may, in principle, depend on subjective perceptions (such as value to society and personal enjoyment), and also on the offered prices (the so-called *anchoring effect* [Tversky and Kahneman 1974; Mason and Watts 2009]). Even if the internal cost is a well-defined quantity, do workers know it?

• *Fair price.* Workers may have a perception of a "fair price" which is separate from their internal costs, and may be reluctant to accept a task at a lower price [Mason and Watts 2009; Yin et al. 2013].

• *Task-specific motivations.* In some cases, workers may be motivated by task-specific factors. For example, if a task involves creating content to be posted on the web, workers may be motivated by the possibility of receiving attention [Ghosh and Hummel 2013].

• *Myopia vs. long term strategies.* How myopic are the workers? When making a decision about a particular task, do they take into account the effects of this decision

on their future? For example, a worker may be reluctant to accept highly priced tasks that she is not suitable for, because her bad performance on these tasks would impact her reputation score with the platform and/or the requester, and therefore affect what she is offered in the future. For a similar reason, a worker may accept an easy but low paying task, or exert additional effort. More generally, workers may be able to strategically alter their behavior in order to affect the long-term behavior of a particular requester or the platform.

• *Strategic timing.* Can the workers strategically choose the time intervals during which they are online? If a requester is using an online learning algorithm, some phases of this algorithm may be more beneficial for workers than others. For example, if a requester starts out with a low price for a given category of tasks, and then gradually adjusts it upwards, then workers may want to arrive later rather than sooner.

Performance objectives. The statement of an algorithmic problem should include a specific performance objective to be optimized, perhaps under some constraints.

A requester-side decision making algorithm typically maximizes the utility of this requester, i.e., the value of completed tasks minus the payment to the workers. If the requester has a pre-specified budget, the algorithm may instead maximize the value of completed tasks subject to budget constraints. In some settings it may be feasible to take into account the workers' happiness, so as to encourage them to stay with this requester in the future. Specifically, in incentive-compatible mechanisms that elicit workers' production costs one may wish to optimize the *social welfare* – the utility of the requester plus the total utility of the workers – or some other weighted sum of these two quantities.

For platform-side algorithms, the performance objective might take into account the utility of requesters, workers, and the platform itself. Assuming the platform receives a fixed percentage of every transaction, the platform's revenue is simply a fixed fraction of the total payment from requesters to workers. One reasonable objective is a weighted sum of the platform's revenue and the total utility of the requesters; the choice of weights is up to the platform. Moreover, the platform may wish to ensure some form of fairness, so that no requesters are starved out. Further, the platform may have additional forward-looking objectives that are not immediately comparable with the platform revenue and requesters' utility, such as the market share and the workers' happiness (to encourage workers to stay with the platform in the future).

4. SPECIFIC DIRECTIONS

Given the abundance of modeling choices identified, it appears hopeless to seek out a single unified model that attempts to capture all aspects of repeated decision making in crowdsourcing markets. Nevertheless, it may be possible to study, understand, and solve the fundamental issues that arise in this problem space by carefully choosing specific (families of) models that encapsulate specific salient features — especially if we, as a community, keep generalizability and robustness in mind as we make our modeling and algorithmic choices.

Below we identify and discuss several directions in the problem space that appear

ripe for near-term progress. As it happens, these directions encompass most existing work of which we are aware.

4.1 Adaptive task assignment

One problem of interest is assigning tasks to workers with the goal of maximizing the quality of completed tasks at a low price or subject to budget constraints. In this *task assignment* problem, strategic issues are ignored in order to gain analytical tractability; the model typically does not touch on the way in which prices are set, and does not include workers' strategic responses to these prices. It can be studied in settings in which the quality of work performed is immediately observable or settings in which it is not.

Much of the existing work on task assignment focuses on *classification tasks*, in which workers are asked multiple-choice questions to provide labels for images, websites, web search results, or other types of queries. Then it is natural to assume that the quality of performed work (i.e., the correct label) is not immediately observable; instead, the requester may have a limited supply of gold standard tasks that can be used to learn more about workers' skills. While the problem of inferring the solutions to classification problems using labels from multiple sources has been studied for decades in various forms [Dawid and Skene 1979; Sheng et al. 2008; Crammer et al. 2005; 2008; Dekel and Shamir 2009; Welinder et al. 2010], here we focus on the problem of choosing workers in an online fashion to provide the labels; this problem is relatively new.

Existing research on task assignment typically focuses on the problem faced by a single requester. Each worker charges a fixed and known cost per task completed (which may or may not be the same for all workers), and each task may be assigned to multiple workers in order to the improve the quality of the final result (as is necessary for classification tasks). The objective is to optimize a tradeoff between the total cost and the quality of the completed tasks, for example, minimizing the total cost of work subject to the quality being above a given threshold. Partial information on the task difficulty and the workers' skills may or may not be known, e.g., in the form of reputation scores or Bayesian priors.

In the most common variant of this problem, workers arrive online and the requester must assign a task (or sequence of tasks) to each new worker as she arrives. Karger et al. [2011; 2013] introduced one such model for classification tasks and proposed a non-adaptive assignment algorithm based on random graph generation along with a message-passing inference algorithm inspired by belief propagation for inferring the correct solution to each task. They proved that their technique is order-optimal in terms of budget when each worker finds all tasks equally difficult. Other models of this form have been studied both for tasks with observable quality [Ho and Vaughan 2012] and for classification tasks with unobservable quality but access to gold standard tasks [Ho et al. 2013]. In these papers, the authors utilize online primal-dual techniques [Buchbinder and Naor 2009; Devanur et al. 2011] and show that adaptive task assignment yields an improvement over non-adaptive assignment when the pool of available workers and set of tasks are diverse.

Alternatively, one might consider a model in which the requester may choose a particular worker or category of workers to assign to a particular task rather than choosing a task to assign to a particular worker. This setting already encompasses

a variety of specific models of varying levels of difficulty, but even a version with a single task assigned multiple times is quite challenging [Abraham et al. 2013].

It is worth noting that the models described above are not covered by prior work on multi-armed bandits (where an algorithm observes a reward after each round and the goal is to maximize the total reward over time). Essentially, this is because in adaptive task assignment each rounds brings *information* rather than reward; the value of this information is not immediately known, and in any case the goal is not to maximize the total value of collected information but to arrive at high-quality solutions for the tasks.

On the other hand, multi-armed bandit formulations are more directly applicable to versions of adaptive task assignment in which the quality or utility of each completed task is immediately observable, and the goal is to maximize the total utility over time. The basic version in which the tasks are homogeneous and the algorithm chooses among workers has been studied in Tran-Thanh et al. [2012]; the novelty here is that one needs to incorporate budget constraints. This version falls under the general framework that was later defined and optimally solved in Badanidiyuru et al. [2013a].

The problem formulations surveyed above are quite different from one another, and the algorithms and theoretical guarantees are therefore incomparable. There is a lot of room for further work extracting the key features of the task assignment problem and take-away messages that generalize to a wider range of models and assumptions.

4.2 Dynamic procurement

Dynamic procurement focuses on repeated *posted pricing*, as applied to hiring workers in a crowdsourcing market.[2] Each interaction between the requester and the workers follows a very simple protocol: the requester posts a price (for a given task) which can be either accepted or rejected by the workers.

The basic model of dynamic procurement is as follows. In each round, the requester posts a task and a price for this task, and waits until either some worker accepts the task or the offer times out. The objective is to maximize the number of completed tasks under fixed constraints on budget and waiting time. A standard model of workers' rationality is assumed. In particular, each worker has a fixed "internal cost" which is known to the worker but not to the mechanism. A worker accepts a task only if the posted price for this task matches or exceeds her internal cost. If multiple such tasks are offered at the same time, the worker picks the one which maximizes her utility (offered price minus the internal cost). The requester typically has a very limited knowledge of the *supply curve*, the distribution of workers' internal costs. However, the supply curve may be learned over time.

The basic model described above assumes several significant simplifications as compared to the discussion in Section 3. First, workers cannot strategically manipulate their effort level in response to the offered prices. Second, workers are myopic, in the sense that they do not take into account the effects of their decision in a given round on the future rounds. A standard way to model myopic workers is to assume that the requester interacts with a new worker in each round. Third,

---

[2]The term "dynamic procurement" is from Badanidiyuru et al. [2012; 2013a; 2013b].

the supply curve does not change over time. Finally, the issues of (explicit) task assignment are completely ignored.

Despite all the simplifications, this basic model is very non-trivial. Considerable progress has been made in a recent line of work [Badanidiyuru et al. 2012; Badanidiyuru et al. 2013a; 2013b; Singla and Krause 2013], but it is not clear whether the current results are optimal. Further, multiple extensions are possible *within* the basic model. For instance, the requester may be interested in multiple categories of tasks, and may offer more complicated "menus" that allow workers to perform multiple tasks of the same type.[3]

One can also consider a platform-side version of dynamic procurement, in which the platform is in charge of setting posted prices for all requesters, under budget constraints submitted by the requesters or imposed by the platform. The most basic goal would be to maximize the total number of completed tasks over all requesters, perhaps under some fairness constraints.

It is worth noting that dynamic procurement is closely related to *dynamic pricing*: repeated posted pricing for selling items. In particular, dynamic procurement on a budget with unknown supply curve has a natural "dual" problem about selling items: dynamic pricing with limited inventory and unknown demand curve. The latter problem has received some attention (e.g., [Besbes and Zeevi 2009; Babaioff et al. 2012; Besbes and Zeevi 2012; Badanidiyuru et al. 2013a]). Moreover, Badanidiyuru et al. [2013a] consider a general framework which subsumes both problems.

Dynamic procurement can be compared to a more general class of game-theoretic mechanisms in which workers are required to explicitly submit their preferences (see, e.g., Singer and Mittal [2013]). Dynamic procurement is appealing for several reasons (as discussed, for example, in Babaioff et al. [2012] and Chawla et al. [2010]). First, a worker only needs to evaluate a given offer rather than exactly determine her internal costs and preferences; humans tend to find the former task to be much easier than the latter. Second, a worker reveals very little information about themselves: only whether she is willing to accept a particular offer. Third, posted pricing tends to rule out the possibility that workers may strategically alter their behavior in order to manipulate the requester. However, it is possible that more general mechanisms may achieve better guarantees in some scenarios.

Behavioral effects. In order to ensure that results are applicable to real-world markets, apart from addressing dynamic procurement under the traditional assumptions on workers' rationality, it is desirable to incorporate more complicated behaviors, such as perceived fair prices and the anchoring effect. Then the prices offered by the requester may influence the properties of the worker population, namely change workers' perception of fair prices and/or production costs. The challenge here is to model such influence in a sufficiently quantitative way, and design dynamic pricing algorithms that take this influence into account. However, the existing empirical evidence [Mason and Watts 2009; Yin et al. 2013] appears insufficient to suggest a particular model to design algorithms for; further, even surveying the potentially relevant models is not easy. To make progress, there needs to

---

[3]The general result in Badanidiyuru et al. [2013a; 2013b] applies to some of these extensions, but the particular guarantees are likely to be suboptimal.

be a convergence of empirical work on modeling that is geared towards algorithmic applications, and algorithm design work that is fully aware of the modeling issues.

One can draw upon a considerable amount of prior work on reference prices [Tversky and Kahneman 1991; Putler 1992; Kalyanaram and Winer 1995; Thaler 2008], and some recent work on dynamic pricing with reference price effects [Popescu and Wu 2007; Nasiry and Popescu 2011]. However, all of this work is usually in the context of selling items to consumers, and therefore may not be directly applicable to crowdsourcing. Moreover, Popescu and Wu [2007] and Nasiry and Popescu [2011] assume a known demand distribution.

In view of the above, one wonders how significant the behavioral effects are in real-life crowdsourcing markets, and whether it is safe to ignore them in the context of dynamic procurement. To the best of our understanding, this issue is currently unresolved; while significant behavioral effects have been observed in specifically designed, very short-term experiments, it is not clear whether the significance is preserved in longer-running systems.

4.3 Repeated principal-agent problem

Consider an extension of dynamic procurement to a scenario in which the workers can strategically change their effort level depending on the price or contract that they are offered. The chosen effort level probabilistically affects the quality of completed work, determining a distribution over the possible quality levels. Each worker is characterized by a mapping from effort levels to costs and distributions over the quality levels; this mapping, called worker's *type*, is known to the worker but not to the mechanism. Crucially, the choice of effort level is not directly observable by the requester, and cannot (in general) be predicted or inferred from the observed output. The single-round version of this setting is precisely the *principal-agent problem* [Laffont and Martimort 2002], the central model in contract theory, a branch of Economics.[4]

Posted pricing is not adequate to incentivize workers to produce high-quality work in this setting since workers could exert the minimal amount of effort without any loss of payment. Instead, requesters may want to use more complicated contracts in which the payment may depend on the quality of the completed work. Note that contracts *cannot* directly depend on the worker's effort level, as the latter is not known to the requester. The mechanism may adjust the offered contract over time, as it learns more about the worker population.

To make the problem more tractable, it would probably help to assume that workers are myopic, and that the quality of completed tasks is immediately observable. As in dynamic procurement, the issues of task assignment are completely ignored.

Even with these simplifications, the *repeated principal-agent* setting described above is significantly more challenging than dynamic procurement. Essentially, this because the principal-agent problem is a vast generalization of the simple accept/reject interaction in dynamic procurement. The space of possible contracts and the space of possible types are much richer than their counterparts in dynamic procurement, and the mapping from an action to requester's expected utility is

---

[4]In the language of contract theory, the requester is the "principal" and workers are "agents."

more difficult to characterize and analyze. In particular, contracts are not limited to mappings from observed outcomes to payments. In principle, the requester can specify a *menu* of several such mappings, and allow a worker to choose among them. (It is well-known that using such "menus" one can maximize the requester's expected utility among all possible interaction protocols between a single requester and a single worker [Laffont and Martimort 2002].)

A notable additional complication is that it may be advantageous to reduce the action space – the class of contracts that an algorithm considers. While reducing the action space may decrease the quality of the best feasible contract, it may improve the speed of convergence to this contract, and make the platform more user-friendly for workers. For example, one may want to reduce the number of items in the "menu" described above (perhaps all the way to single-item menus), reduce the granularity of the quality levels considered, or restrict attention to human-friendly "monotone" contracts, in which higher quality levels always result in higher payments to the workers.

Perhaps surprisingly, the rich literature in contract theory sheds little light on this setting. Most work focuses on a single interaction with an agent whose type is either known or sampled from a known distribution, see Laffont and Martimort [2002] for background. Some papers [Babaioff et al. 2006; Misra et al. 2012] have studied settings in which the principal interacts with multiple agents, but makes all its decisions at a single time point. In Sannikov [2008], Williams [2009], and Sannikov [2012], the principal makes repeated decisions over time, but the requester interacts with only a single agent who optimizes the long-term utility. The agent's type is sampled from a known prior.

Conitzer and Garera [2006] consider the basic version of the repeated principal-agent problem — with single-item "menus" and no budget constraints — adapt several algorithms from prior work to this setting, and empirically compare their performance. In an ongoing project, Ho et al. [2013] design algorithms with (strong) provable guarantees for the same basic version, drawing on the connection with some prior work [Kleinberg et al. 2008; Bubeck et al. 2011; Slivkins 2011] on multi-armed bandit problems. However, their guarantees make significant restrictions on the action space. We are not aware of any other directly relevant work (beyond the work on dynamic pricing, which does not capture the unobservable choice of worker's effort level).

Progress on repeated principal-agent problems might suggest improvements in the design of crowdsourcing platforms. In particular, it may inform the platform's choice of the action space. Currently even the basic version of the repeated principal-agent problem is not well-understood.

4.4   Reputation systems

Persistent reputation scores for workers may help limit spam and encourage high quality work. Likewise, persistent reputation scores for requesters may encourage requesters to be more considerate towards the workers. Thus, one may want to design a stand-alone "reputation system" which defines and maintains such scores (perhaps with the associated confidence levels), as a tool which can be used in different applications and in conjunction with different higher-level algorithms. Reputation systems may be designed to address issues related to either *moral hazard*

(when workers' effort is desirable but not immediately observable) or *adverse selection* (when similar-looking workers may be different from one another, and it is desirable to attract workers who are more skilled or competent), or a combination of the two.

There is already a rich literature on reputation systems for online commerce, peer-to-peer networks, and other domains; see, for example, Friedman et al. [2007], Resnick et al. [2000], or the seminal paper of Dellarocas [2005]. However, the basic models for reputation systems have several limitations when applied to crowdsourcing markets. First, there may be domain-specific design goals that depend on a particular application or (even worse) on a particular algorithm which uses the reputation scores. Then it may be necessary to design the reputation system and the platform's or requesters' algorithms in parallel, as one inter-related problem. Second, reputation systems are typically designed separately from task assignment, which artificially restricts the possibilities and can therefore lead to sup-optimal designs. Third, whenever the properties of the worker population are not fully known, there is an issue of exploration: essentially, it may be desirable to give the benefit of a doubt to some low-rated workers, so as to obtain more information about them.

Zhang and van der Schaar [2012] and Ho et al. [2012] examine the problem of designing a reputation system specific to crowdsourcing markets, building on the idea of *social norms* first proposed by Kandori [1992]. Social norms 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. They pair reputation systems with *differential service schemes* that allocate more resources or better treatment to those with higher reputation in such a way that market participants have the incentive to follow the prescribed rules of the market. The social norms designed by Zhang and van der Schaar [2012] and Ho et al. [2012] address only the moral hazard problem, ignoring issues related to adverse selection by making assumptions about known types, and do so on greatly simplified models that capture some salient features of crowdsourcing systems (such as the difficulty of perfectly evaluating the quality of work and the fact that the population may be changing over time) but ignore many others (such as the vast diversity of workers and requesters in real systems and the fact that workers may be learning over time).

There is significant room for further research studying more realistic models, looking simultaneously at moral hazard and adverse selection, and better defining what an "optimal" reputation system means in general.

4.5 One common theme: The exploration-exploitation tradeoff

Most models surveyed in this section exhibit a tradeoff between *exploration* and *exploitation*, i.e., between obtaining new information and making optimal per-round decisions based on the information available so far [Cesa-Bianchi and Lugosi 2006; Bergemann and Välimäki 2006; Bubeck and Cesa-Bianchi 2012]. This tradeoff is present in any setting with repeated decisions and partial, action-dependent feedback, i.e., whenever the feedback received by an algorithm in a given round depends on the algorithm's action in this round. The explore-exploit tradeoff occurs in many areas, including the design of medical trials, pay-per-click ad auctions, and adaptive network routing; numerous substantially different models have been

studied in the literature, to address the peculiarities of the various application domains.

An issue which cuts across most settings with explore-exploit tradeoff, including those arising in crowdsourcing markets, is whether to use a fixed schedule for exploration, or to adapt this schedule as new observations come in. In the former case, one usually allocates each round to exploration or exploitation (either randomly or deterministically, but without looking at the data), chooses uniformly at random among the available alternatives in exploration rounds, and maximizes per-round performance in each exploitation round given the current estimates for the unknown quantities. In the latter case, which we call *adaptive exploration*, exploration is geared towards more promising actions, without sacrificing too much in short-term performance. Often there is no explicit separation between exploration and exploitation, so that in each round the algorithm does a little of both. Adaptive exploration tends to be better at taking advantage of the "niceness" of the problem instance (where the particular notion of "niceness" depends on the model), whereas non-adaptive exploration often suffices to achieve the optimal worst-case performance.

For a concrete example, consider a simple, stylized model in which an algorithm chooses among two actions in each round, each action x brings a 0-1 reward sampled independently from a fixed distribution with unknown expectation μ_x, and the goal is to maximize the total reward over a given time horizon of 100 rounds. One algorithm involving non-adaptive exploration would be to try each action 10 times, pick the one with the best empirical reward, and stick with this action from then on. (However, while in this example choosing the best action in each exploitation round is trivial, in some settings this choice can be the crux of the solution [Ho et al. 2013].) On the other hand, one standard approach for adaptive exploration is to maintain a numerical score (index or upper confidence bound) for each action, which reflects both the average empirical reward and the sampling uncertainty, and in each round to pick an action with the highest score.

A closely related issue, endemic to all work on dynamic procurement (and dynamic pricing), is *discretization* of the action space (in this case, the price space). Essentially, it is complicated to consider *all* possible prices or price vectors, and instead one often focuses on a small but representative subset of thereof. For example, one may focus on prices that are integer multiples of some a priori chosen $\epsilon \in (0,1)$. While optimizing among the remaining prices is very simple in the most basic setting of dynamic procurement without budget constraints [Kleinberg and Leighton 2003], it can be very difficult in more advanced settings [Badanidiyuru et al. 2013a; 2013b; Singla and Krause 2013]. Prior work on related, but technically different explore-exploit problems [Kleinberg et al. 2008; Bubeck et al. 2011; Slivkins 2011] suggests that it may be advantageous to choose the discretization *adaptively*, depending on the previous observations, so that the algorithm zooms in faster on the more promising regions of the action space.

Understanding these issues is crucial to the design of algorithms for repeated decision making in crowdsourcing systems. At the same time, models designed specifically for the crowdsourcing domain often present new algorithmic challenges in explore-exploit learning (as is the case for most work on adaptive task assignment

and dynamic procurement). Techniques designed to address these challenges may be applicable more broadly in other scenarios in which the exploration-exploitation tradeoff arises. (For example, dynamic procurement for crowdsourcing markets is one of the major motivating examples behind the general framework in Badanidiyuru et al. [2013a].)

5. CONCLUSIONS

Crowdsourcing is a new application domain for online decision making algorithms, opening up a rich and exciting problem space in which the relevant problem formulations vary significantly along multiple modeling "dimensions." This richness presents several challenges. Most notably, it is difficult for the community to converge on any particular collection of models, and as a result, it is difficult to compare results and techniques across papers. Additionally, any particular modeling choices have inherent limitations that are sometimes hard to see. It is not clear a priori which limitations will impact the practical performance of algorithms, making claims about robustness difficult to achieve.

In this paper, we attempt to raise awareness of these issues. Towards this goal, we identify a multitude of possible modeling choices, and discuss several specific directions in which progress can be made in the near future. We hope that this will spark discussion and aide the community as it moves forward.

Acknowledgements

REFERENCES

ABRAHAM, I., ALONSO, O., KANDYLAS, V., AND SLIVKINS, A. 2013. Adaptive crowdsourcing algorithms for the bandit survey problem. In *26th Conf. on Learning Theory (COLT)*.

BABAIOFF, M., DUGHMI, S., KLEINBERG, R., AND SLIVKINS, A. 2012. Dynamic pricing with limited supply. In *13th ACM Conf. on Electronic Commerce (EC)*.

BABAIOFF, M., FELDMAN, M., AND NISAN, N. 2006. Combinatorial agency. In *7th ACM Conf. on Electronic Commerce (EC)*.

BADANIDIYURU, A., KLEINBERG, R., AND SINGER, Y. 2012. Learning on a budget: posted price mechanisms for online procurement. In *13th ACM Conf. on Electronic Commerce (EC)*. 128–145.

BADANIDIYURU, A., KLEINBERG, R., AND SLIVKINS, A. 2013a. Bandits with knapsacks. In *54th IEEE Symp. on Foundations of Computer Science (FOCS)*.

BADANIDIYURU, A., KLEINBERG, R., AND SLIVKINS, A. 2013b. Bandits with knapsacks: Dynamic procurement for crowdsourcing. In *The 3rd Workshop on Social Computing and User Generated Content, co-located with* ACM EC 2013.

BERGEMANN, D. AND VÄLIMÄKI, J. 2006. Bandit Problems. In *The New Palgrave Dictionary of Economics, 2nd ed.*, S. Durlauf and L. Blume, Eds. Macmillan Press.

BESBES, O. AND ZEEVI, A. 2009. Dynamic pricing without knowing the demand function: Risk bounds and near-optimal algorithms. *Operations Research 57*, 1407–1420.

BESBES, O. AND ZEEVI, A. J. 2012. Blind network revenue management. *Operations Research 60,* 6, 1537–1550.

BORODIN, A. AND EL-YANIV, R. 1998. *Online computation and competitive analysis.* Cambridge University Press.

BUBECK, S. AND CESA-BIANCHI, N. 2012. Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems. *Foundations and Trends in Machine Learning 5,* 1, 1–122.

BUBECK, S., MUNOS, R., STOLTZ, G., AND SZEPESVARI, C. 2011. Online Optimization in X-Armed Bandits. *J. of Machine Learning Research (JMLR) 12*, 1587–1627.

BUCHBINDER, N. AND NAOR, J. 2009. The design of competitive online algorithms via a primal-dual approach. *Foundations and Trends in Theoretical Computer Science 3,* 2-3, 93–263.

CESA-BIANCHI, N. AND LUGOSI, G. 2006. *Prediction, learning, and games.* Cambridge Univ. Press.

CHAWLA, S., HARTLINE, J. D., MALEC, D. L., AND SIVAN, B. 2010. Multi-parameter mechanism design and sequential posted pricing. In *ACM Symp. on Theory of Computing (STOC).* 311–320.

CHEN, X., LIN, Q., AND ZHOU, D. 2013. Optimistic knowledge gradient for optimal budget allocation in crowdsourcing. In *30th Intl. Conf. on Machine Learning (ICML).*

CONITZER, V. AND GARERA, N. 2006. Online learning algorithms for online principal-agent problems (and selling goods online). In *23rd Intl. Conf. on Machine Learning (ICML).*

CRAMMER, K., KEARNS, M., AND WORTMAN, J. 2005. Learning from data of variable quality. In *19th Advances in Neural Information Processing Systems (NIPS).*

CRAMMER, K., KEARNS, M., AND WORTMAN, J. 2008. Learning from multiple sources. *Journal of Machine Learning Research 9*, 1757–1774.

DAWID, P. AND SKENE, A. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Journal of the Royal Statistical Society, Series C (Applied Statistics) 28,* 1, 20–28.

DEKEL, O. AND SHAMIR, O. 2009. Vox populi: Collecting high-quality labels from a crowd. In *22nd Conf. on Learning Theory (COLT).*

DELLAROCAS, C. 2005. Reputation mechanism design in online trading environments with pure moral hazard. *Information Systems Research 16,* 2, 209–230.

DEVANUR, N., JAIN, K., SIVAN, B., AND WILKENS, C. 2011. Near optimal online algorithms and fast approximation algorithms for resource allocation problems. In *12th ACM Conf. on Electronic Commerce (EC).*

FRIEDMAN, E., RESNICK, P., AND SAMI, R. 2007. Manipulation-resistant reputation systems. In *Algorithmic Game Theory*, N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani, Eds. Cambridge University Press.

GHOSH, A. AND HUMMEL, P. 2013. Learning and incentives in user-generated content: Multi-armed bandits with endogenous arms. In *4th Innovations in Theoretical Computer Science Conference (ITCS).*

GITTINS, J., GLAZEBROOK, K., AND WEBER, R. 2011. *Multi-Armed Bandit Allocation Indices.* John Wiley & Sons.

HO, C.-J., JABBARI, S., AND VAUGHAN, J. W. 2013. Adaptive task assignment for crowdsourced classification. In *30th Intl. Conf. on Machine Learning (ICML).*

HO, C.-J., SLIVKINS, A., AND VAUGHAN, J. W. 2013. Adaptive contract design for crowdsourcing. Working Paper. Preliminary version to appear in the NIPS '13 Workshop on Crowdsourcing: Theory, Algorithms, and Applications.

HO, C.-J. AND VAUGHAN, J. W. 2012. Online task assignment in crowdsourcing markets. In *26th AAAI Conference on Artificial Intelligence (AAAI).*

HO, C.-J., ZHANG, Y., VAUGHAN, J. W., AND VAN DER SCHAAR, M. 2012. Towards social norm design for crowdsourcing markets. In *4th Human Computation Workshop (HCOMP).*

IPEIROTIS, P. G., PROVOST, F., SHENG, V. S., AND WANG, J. 2013. Repeated labeling using multiple noisy labelers. *Data Mining and Knowledge Discovery*.

KALYANARAM, G. AND WINER, R. S. 1995. Empirical generalizations from reference price research. *Marketing Science 14,* 3.

KANDORI, M. 1992. Social norms and community enforcement. *The Review of Economic Studies 59,* 1, 63–80.

KARGER, D., OH, S., AND SHAH, D. 2011. Iterative learning for reliable crowdsourcing systems. In *25th Advances in Neural Information Processing Systems (NIPS)*.

KARGER, D., OH, S., AND SHAH, D. 2013. Budget-optimal task allocation for reliable crowdsourcing systems. To appear.

KITTUR, A., NICKERSON, J. V., BERNSTEIN, M. S., GERBER, E. M., SHAW, A., ZIMMERMAN, J., LEASE, M., AND HORTON, J. J. 2013. The future of crowd work. In *16th ACM Conference on Computer-Supported Cooperative Work and Social Computing (CSCW)*.

KLEINBERG, R. AND LEIGHTON, T. 2003. 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)*. 594–605.

KLEINBERG, R., SLIVKINS, A., AND UPFAL, E. 2008. Multi-Armed Bandits in Metric Spaces. In *40th ACM Symp. on Theory of Computing (STOC)*. 681–690.

LAFFONT, J.-J. AND MARTIMORT, D. 2002. *The Theory of Incentives: The Principal-Agent Model*. Princeton University Press.

LASECKI, W. S., MILLER, C. D., KUSHALNAGAR, R. S., AND BIGHAM, J. P. 2013. Legion scribe: real-time captioning by the non-experts. In *Intl. Cross-Disciplinary Conf. on Web Accessibility (W4A)*. 22.

MAO, A., PROCACCIA, A. D., AND CHEN, Y. 2013. Better human computation through principled voting. In *27th AAAI Conference on Artificial Intelligence (AAAI)*.

MASON, W. AND SURI, S. 2012. Conducting behavioral research on Amazon's Mechanical Turk. *Behavior Research Methods*. To appear.

MASON, W. AND WATTS, D. J. 2009. Financial incentives and the "performance of crowds". In *1st Human Computation Workshop (HCOMP)*.

MISRA, S., NAIR, H. S., AND DALJORD, O. 2012. Homogenous contracts for heterogeneous agents: Aligning salesforce composition and compensation. Working Paper.

NASIRY, J. AND POPESCU, I. 2011. Dynamic pricing with loss averse consumers and peak-end anchoring. *Operations Research 59,* 6, 1361–1368.

NORONHA, J., HYSEN, E., ZHANG, H., AND GAJOS, K. Z. 2011. Platemate: crowdsourcing nutritional analysis from food photographs. In *24th ACM Symp. on User Interface Software and Technology (UIST)*. 1–12.

PFEIFFER, T., GAO, X. A., CHEN, Y., MAO, A., AND RAND, D. G. 2012. Adaptive polling for information aggregation. In *26th AAAI Conference on Artificial Intelligence (AAAI)*.

POPESCU, I. AND WU, Y. 2007. Dynamic pricing strategies with reference effects. *Operations Research 55,* 3, 413–429.

PUTLER, D. S. 1992. Incorporating reference price effects into a theory of consumer choice. *Marketing Science 11,* 3, 287–309.

RESNICK, P., ZECKHAUSER, R., FRIEDMAN, E., AND KUWABARA, K. 2000. Reputation systems. *43,* 12, 45–48.

SANNIKOV, Y. 2008. A continuous-time version of the principal-agent problem. In *The Review of Economics Studies*.

SANNIKOV, Y. 2012. Contracts: The theory of dynamic principal-agent relationships and the continuous-time approach. In *10th World Congress of the Econometric Society*.

SHENG, V., PROVOST, F., AND IPEIROTIS, P. 2008. Get another label? Improving data quality using multiple, noisy labelers. In *KDD*.

SINGER, Y. AND MITTAL, M. 2013. Pricing mechanisms for crowdsourcing markets. In *Intl. World Wide Web Conf. (WWW)*.

SINGLA, A. AND KRAUSE, A. 2013. Truthful incentives in crowdsourcing tasks using regret minimization mechanisms. In *22nd Intl. World Wide Web Conf. (WWW)*. 1167–1178.

SLIVKINS, A. 2011. Contextual Bandits with Similarity Information. In *24th Conf. on Learning Theory (COLT)*. To appear in J. of Machine Learning Research (JMLR), 2013.

THALER, R. H. 2008. Mental accounting and consumer choice. *Marketing Science 27,* 1, 15–25.

TRAN-THANH, L., STEIN, S., ROGERS, A., AND JENNINGS, N. R. 2012. Efficient crowdsourcing of unknown experts using multi-armed bandits. In *20th European Conf. on Artificial Intelligence (ECAI)*. 768–773.

TVERSKY, A. AND KAHNEMAN, D. 1974. Judgment under uncertainty: Heuristics and biases. *Science 185,* 4157, 1124–1131.

TVERSKY, A. AND KAHNEMAN, D. 1991. Loss aversion in riskless choice: A reference-dependent model. *The Quarterly J. of Economics 106,* 4, 1039–1061.

VIZWIZ. vizwiz.org: crowdsourcing to answer visual questions for the visually impaired.

WANG, J., IPEIROTIS, P. G., AND PROVOST, F. 2013. Quality-based pricing for crowdsourced workers. NYU-CBA Working Paper CBA-13-06.

WELINDER, P., BRANSON, S., BELONGIE, S., AND PERONA, P. 2010. The multidimensional wisdom of crowds. In *24th Advances in Neural Information Processing Systems (NIPS)*. 2424–2432.

WILLIAMS, N. 2009. On dynamic principal-agent problems in continuous time. Working Paper.

YIN, M., CHEN, Y., AND SUN, Y.-A. 2013. The effects of performance-contingent financial incentives in online labor markets. In *27th AAAI Conference on Artificial Intelligence (AAAI)*.

ZAIDAN, O. AND CALLISON-BURCH, C. 2011. Crowdsourcing translation: Professional quality from non-professionals. In *49th Annual Meeting of the Assn. for Computational Linguistics: Human Language Technologies (ACL-HLT)*. 1220–1229.

ZHANG, H., LAW, E., MILLER, R., GAJOS, K., PARKES, D. C., AND HORVITZ, E. 2012. Human computation tasks with global constraints. In *CHI*. 217–226.

ZHANG, Y. AND VAN DER SCHAAR, M. 2012. Reputation-based incentive protocols in crowdsourcing applications. In *31st IEEE International Conference on Computer Communications (IEEE Infocom)*.