2017

Essays In Algorithmic Market Design Under Social Constraints

Hoda Heidari
University of Pennsylvania, hoda.heidari@gmail.com

Essays In Algorithmic Market Design Under Social Constraints

Abstract
Rapid technological advances over the past few decades---in particular, the rise of the internet---has significantly reshaped and expanded the meaning of our everyday social activities, including our interactions with our social circle, the media, and our political and economic activities

This dissertation aims to tackle some of the unique societal challenges underlying the design of automated online platforms that interact with people and organizations---namely, those imposed by legal, ethical, and strategic considerations.

I narrow down attention to fairness considerations, learning with repeated trials, and competition for market share. In each case, I investigate the broad issue in a particular context (i.e. online market), and present the solution my research offers to the problem in that application.

Addressing interdisciplinary problems, such as the ones in this dissertation, requires drawing ideas and techniques from various disciplines, including theoretical computer science, microeconomics, and applied statistics.

The research presented here utilizes a combination of theoretical and data analysis tools to shed light on some of the key challenges in designing algorithms for today's online markets, including crowdsourcing and labor markets, online advertising, and social networks among others.

Degree Type
Dissertation

Degree Name
Doctor of Philosophy (PhD)

Graduate Group
Computer and Information Science

First Advisor
Michael Kearns

Second Advisor
Ali Jadbabaie

Subject Categories
Computer Sciences

ESSAYS IN ALGORITHMIC MARKET DESIGN UNDER SOCIAL CONSTRAINTS

Hoda Heidari

A DISSERTATION

in

Computer and Information Science

Presented to the Faculties of the University of Pennsylvania

in

Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

2017

Supervisor of Dissertation

Michael Kearns
Professor and National Center Chair, CIS

Co-Supervisor of Dissertation

Ali Jadbabaie, JR East Professor, MIT

Graduate Group Chairperson

Lyle Ungar, Professor, CIS

Dissertation Committee:

Rakesh Vohra, George A. Weiss and Lydia Bravo Weiss University Professor (chair)
Shivani Agarwal, Rachleff Family Associate Professor, CIS
Aaron Roth, Associate Professor, CIS
Vahab S. Mirrokni, Principal Scientist, Google Research (external)

ESSAYS IN ALGORITHMIC MARKET DESIGN UNDER SOCIAL CONSTRAINTS

To Behiye and Reza,

for their love and support.

ACKNOWLEDGEMENT

First and foremost, I would like to warmly thank my doctoral advisor, Professor Michael Kearns, for having been an exemplary advisor to me: There is so much to be said about Michael's amazing ability to come up with, formulate, and guide promising research projects. Aside from the great deal I have learned from him, Michael gave me the freedom to explore my own ideas and interests, and provided me with numerous opportunities to expand my academic network. Michael, beyond being a stellar scholar, you are an extraordinary human being, a role-model, and someone I will look up to for the rest of my life. I am also thankful to my wonderful co-advisor, Professor Ali Jadbabaie, who constantly pushed me to not settle, but to work hard to expand my knowledge beyond what was standard for my area of research. I am grateful to both of my advisors for their continued patience, encouragement, and generous support throughout my doctoral studies.

I have been extremely fortunate to have had the opportunity to learn from several fantastic mentors, including Dr Jenn Wortman Vaughan, Sergei Vassilvitskii, and Mohammad Mahdian. This dissertation would not have been possible without the help of my brilliant coauthors, including Sanjeev Goyal, Moez Draeif, Aaron Roth, David Pennock, Sebastien Lahaie, Umar Syed, Hossein Azari, and Bill Heavlin.

Thanks to my committee members, Professors Rakesh Vohra, Aaron Roth, Shivani Agarwal, and Dr Vahab Mirrokni, who took time out of their busy schedules to give me feedback and advice on this dissertation. A special thanks to the Penn community as a whole, especially faculty and staff at the CIS department, for fostering an environment of not only academic, but also personal growth.

Last but not least, I am forever grateful to my family and friends. Mom, you always hold me up to the highest standards. Dad, from early childhood you instilled in me the scientific curiosity that led me to pursue this degree. Elham, you constantly remind me that my mistakes and failures do not take away from what's most important in life. Michele,

Hassan, David, Mehdi, Amir, Salar, and Mariann; Saba, Negin, Atena, Morteza, Jamie, Zara, Beth, and Christine, and Maryam—thanks for being there for me at various points of my life in grad school; and thanks for all the wonderful times we spent together. Our friendship has been the most meaningful part of my non-academic life, and I am truly blessed for having met you.

ABSTRACT

ESSAYS IN ALGORITHMIC MARKET DESIGN UNDER SOCIAL CONSTRAINTS

Hoda Heidari

Michael Kearns

Ali Jadbabaie

Rapid technological advances over the past few decades—in particular, the rise of the Internet—has significantly reshaped and expanded the meaning of our everyday social activities, including our interactions with our social circle, the media, and our political and economic activities This dissertation aims to tackle some of the unique *societal* challenges underlying the design of automated online platforms that interact with people and organizations—namely, those imposed by *legal*, *ethical*, and *strategic* considerations. I narrow down attention to fairness considerations, learning with repeated trials, and competition for market share. In each case, I investigate the broad issue in a particular context (i.e. online market), and present the solution my research offers to the problem in that application. Addressing interdisciplinary problems, such as the ones in this dissertation, requires drawing ideas and techniques from various disciplines, including theoretical computer science, microeconomics, and applied statistics. The research presented here utilizes a combination of theoretical and data analysis tools to shed light on some of the key challenges in designing algorithms for today's online markets, including crowdsourcing and labor markets, online advertising, and social networks among others.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF ILLUSTRATIONS

CHAPTER 1 : Introduction

There is little doubt left that today's social relations are heavily influenced by the new online world. The impact can be seen in our everyday interactions with our friends, acquaintances, and media, and even in our political and economic activities. In the last two decades, the online world has expanded at an extraordinary pace, and all the aforementioned activities have witnessed a significant shift to high-tech, online, and electronic platforms. In addition to fundamental changes to existing markets, new markets—and hence new relationships and roles—emerge everyday.

In the past, it was the job of social sciences, such as sociology and economics, to address problems related to systems of humans,organizations, and their interactions, whereas computer science and engineering were concerned mostly with improving the performance of machines. Today the online world is a complex system comprised of machines, humans, and organizations. As the result areas of research that seemed unrelated to each other in the past now need to join forces to further our understanding of this complicated system as a whole (see Nisan et al., 2007; Easley and Kleinberg, 2010, for numerous applications and case-studies.).

The current dissertation on algorithmic market design contributes to this recent endeavor, and tackles some of the unique challenges that arise when automated online platforms interact with the society. In particular, my focus will be on the design constraints imposed by *legal*, *ethical*, and *strategic* considerations. I investigate each of these broad issues in a number of specific market settings, and present the solution my research offers to the problem in detail.

1.1. Algorithmic Market Design

My dissertation contains essays on *algorithmic market design*, a subfield of Algorithmic Game Theory, broadly concerned with the design and analysis of algorithms that run on-

line markets (see Nisan et al., 2007). The problems studied in this interdisciplinary area of research usually have two components aspects: an *algorithmic*, as well as a *societal* aspect. A market designer is by definition concerned with the efficiency of the market—both in its economic and computational sense: The market algorithm must be computationally tractable and more broadly practical. Furthermore, it is imperative that the designer take into account the impact of the algorithm on the behavior of the human/self-interested participants of the system. Classical examples of the constraints imposed by strategic considerations are participation constraints and incentive compatibility (see Mas-Colell et al., 1995, Chapter 23).

Online markets frequently present themselves with further social challenges (Vulkan et al., 2013), including those imposed by legal (e.g. simultaneously implementing all contracts between the market and its participants), ethical (e.g. respecting anti-discrimination laws; providing all market participants with equal opportunities), or strategic constraints (e.g. maximizing revenue in a competitive market). Addressing the delicate trade-offs among various efficiency objectives and such constraints is the central theme in this dissertation.

1.2. Overview of This Dissertation

In this dissertation, I focus on a number of key societal challenges a market designer faces when dealing with automated online platforms:

- **Fairness and equality considerations:** The market should treat similar individuals and circumstances similarly. In Chapter 2, I tackle constraints of this type in the context of *prediction*, and *online labor* markets.
- **Learning with repeated interactions:** An online market usually interacts with the same set of participants over time. The market and its participants can utilize this to *learn* about the other party and optimize their future actions accordingly. In Chapter 3, I address problems of this flavor in the context of *ad exchanges*—both from a theoretical and an empirical perspective.

- **Competition for market share:** A major challenge for any online business operating within a competitive environment is to ensure that a sufficient proportion of potential customers prefers their service to that of its competitors. The analysis of this competition is the subject of Chapter 4. I will look into handling competition in the context of *ad exchanges* and *social networks*.

1.2.1. Fairness and Equality Considerations

Chapter 2 consists of two sections. Section 2.1 is dedicated to the study task assignment in crowdsourcing systems in presence of considerations for "equality of opportunity". A fair task assignment scheme must ensure quick turnaround time, while simultaneously provide some guarantee for equality of opportunity among workers. In (Heidari and Kearns, 2013) my coauthor and I present efficient algorithms/structures that simultaneously (approximately) minimize both the maximum workload among workers, and the number of workers that need to attempt a task before it is completed. The former guarantees a small group of high-ability workers don't receive the bulk of the workload, and the latter puts a limit on the time it takes for any given task to be completed.

Section 2.2 proposes a generalization of standard combinatorial prediction markets, where traders are allowed to specify a limit on how much they are willing to pay per share for their bundle of interest—that is, they can submit *limit orders*. This obligates the market to keep track of all open orders/contracts at every point in time and guarantee that executing one is not in conflict with respecting the other orders in the book.

By combining techniques from continuous and discrete optimization, in (Heidari et al., 2015) my coauthors and I provide the first concrete algorithm with theoretical guarantees that combines market makers and limit orders in a combinatorial prediction market with continuous trade. More precisely, we define the notion of an ϵ-fair trading path, a path in security space along which no order executes at a price more than ϵ above its limit, and every order executes when its market price falls more than ϵ below its limit. We develop an

algorithm for operating a continuous market maker with limit orders that respects all the ϵ-fairness conditions simultaneously.

1.2.2. Learning with Repeated Interaction

Chapter 3 consists of two sections. We begin in Section 3.1 by studying how an ad exchange can take advantage of the data it accumulates over time about its bidders to optimize future actions. In particular, in an *online display ad exchanges* the exchange repeatedly interacts with quota-limited bidders, making real-time decisions about which subsets of bidders are called to participate in ad-slot-specific auctions. Given the repeated nature of the interaction with its bidders, the exchange has information about each bidders' segments of interest. This information can be utilized to design smarter callout mechanisms—with the potential of improving the exchange's revenue. In a recent work (see Azari et al., 2017), my collaborators at Google and I present an empirical framework for studying the performance of callout mechanisms in such settings. To formalize the effect of a callout mechanism on long-term revenue, I propose a strategic model that captures the repeated interaction between the exchange and bidder. This model leads us to two metrics for performance: short-term revenue and social welfare. We then present an empirical framework for estimating the performance of a callout mechanism in terms of these two metrics from *historical auction data only*. We put our framework to test by performing extensive simulations on *both synthetic and real auction data* and compare several natural callout heuristics against one another.

The market is not the only side that can take advantage of the information it accumulates over time. Participants are also often times intelligent agents who can learn and improve their performance with repeated trials. In Section 3.2, we propose a stylized model of learning and performance improvement for workers in a crowdsourcing platforms. Our model is a variant of the well-studied multi-armed bandit problem in which the reward from each arm (worker) evolves monotonically in the number of times that arm is pulled (a task is assigned to that worker). We assume that the arm-dependent rates at which

the rewards increase (workers' learning rates) are unknown, and propose task assignment algorithms with provably optimal *policy regret* bounds.

1.2.3. Competition for Market Share

Chapter 4 consists of two sections. In Section 4.1, we address market competition in the context of *ad exchanges*. As the number of exchanges has grown, sellers have turned to low-regret learning mechanisms to decide which exchange has the best price for their inventory. This in turn raises the following question for the exchange: how to set reserve prices to attract a large market share and maximize revenue? In (Heidari et al., 2016b), my collaborators and I formulate this as a learning problem, and present algorithms showing that simply knowing that sellers use low-regret learning to choose among their options is enough for the exchange to have a low-regret algorithm for the optimal price.

Next and motivated by the word-of-mouth and viral marketing, in Section 4.2 my coauthors and I consider a setting in which two firms compete for the consumers located in a social network. Firms have budgets to "seed" the initial adoption of their products/services, and their goal is to maximize their market share. This defines a game among firms. In (Goyal et al., 2014), we identify general properties of the adoption dynamics—namely, decreasing returns to local adoption—which determine whether the inefficiency of resource use at equilibrium (quantified by *the price of anarchy* and *stability*) is bounded across all networks.

We also test the sensitivity of these results to the changes in the structure of the utility functions (see Draief et al., 2014). Building on the framework introduced in (Goyal et al., 2014), we first introduce a new model in which the payoff to each firm comprises not only the number of vertices who adopt its product, but also the network connectivity among those nodes. We also introduce a model in which budgeting decisions are endogenous, rather than externally given as is typically assumed in most viral marketing models.

CHAPTER 2 : Fairness and Equality Considerations

One of the most important classes of social constraints the market designer faces are those stemming from fairness and equality considerations: the market must treat similar individuals and circumstances similarly. In this chapter, I tackle constraints of this type in the context of *prediction*, and *online labor* markets[1].

Section 2.1 is dedicated to the study task assignment in crowdsourcing systems in presence of considerations for "equality of opportunity". A fair task assignment scheme must ensure quick turnaround time, while simultaneously provide some guarantee for equality of opportunity among workers. Heidari and Kearns (2013) present efficient algorithms/structures that simultaneously (approximately) minimize both the maximum workload among workers, and the number of workers that need to attempt a task before it is completed. The former guarantees a small group of high-ability workers don't receive the bulk of the workload, and the latter puts a limit on the time it takes for any given task to be completed.

More precisely, we assume that arriving tasks for the workforce are homogeneous, and that each is characterized by an unknown and one-dimensional *difficulty* value $x \in [0, 1]$. Each worker i is characterized by their *ability* $w_i \in [0, 1]$, and can solve the task if and only if $x \leq w_i$. If a worker is unable to solve a given task it must be forwarded to a worker of greater ability. For a given set of worker abilities W and a distribution P over task difficulty, we are interested in the problem of designing efficient forwarding structures for W and P. We give efficient algorithms and structures that simultaneously (approximately) minimize both the maximum workload of any worker, and the number of workers that need to attempt a task. We identify broad conditions under which workloads diminish rapidly with the workforce size, yet only a constant number of workers attempt each task.

Section 2.2 proposes a generalization of standard combinatorial prediction markets, where traders are allowed to specify a limit on how much they are willing to pay per share for their

[1]The content in Sections 2.1 and 2.2 is taken directly from (Heidari and Kearns, 2013) and (Heidari et al., 2015), respectively.

bundle of interest—that is, they can submit *limit orders*. This obligates the market to keep track of all open orders/contracts at every point in time and guarantee that executing one is not in conflict with respecting the other orders in the book. Heidari et al. (2015) provide the first concrete algorithm for combining market makers and limit orders in a prediction market with continuous trade. Our mechanism is general enough to handle both bundle orders and arbitrary securities defined over combinatorial outcome spaces.

We define the notion of an ϵ-fair trading path, a path in security space along which no order executes at a price more than ϵ above its limit, and every order executes when its market price falls more than ϵ below its limit. We show that under a certain supermodularity condition, a fair trading path exists for which the endpoint is efficient, but that under very general conditions, reaching an efficient endpoint via an ϵ-fair trading path is not possible. We develop an algorithm for operating a continuous market maker with limit orders that respects the ϵ-fairness conditions in the general case in which the supermodularity condition may not hold. We conduct simulations of our algorithm using real combinatorial predictions made during the 2008 U.S. Presidential election and evaluate it against a natural baseline according to trading volume, social welfare, and violations of the two fairness conditions.

2.1. Task Assignment in Online Labor Markets

In crowdsourcing applications and many other settings, a large workforce is available for a series of relatively homogeneous tasks — for instance, labeling images for the presence or absence of a particular object such as cars. Most existing crowdsourcing systems treat the tasks as if they were of uniform difficulty, treat the workers as if they were of uniform ability, and simply assign the next task to the next available worker [2]. In reality, it is quite natural to assume there may be significant variation in both task difficulty and worker ability, and to incorporate this variation into the design of the system.

We thus consider the problem of effectively organizing a population of workers of varying

[2]See (Salek et al., 2013) for an exception that is directly related to the models we examine here.

abilities. We consider a simple model in which we assume that arriving tasks for the workforce are homogeneous, and that each is characterized by an unknown and one-dimensional [3] *difficulty* value $x \in [0, 1]$. Each worker i is characterized by their *ability* $w_i \in [0, 1]$, and can solve the task if and only if $x \leq w_i$ If a worker is unable to solve a given task then it must be passed or *forwarded* to a worker of greater ability. We assume that the worker abilities w_i are known (perhaps via prior observation or experience)[4].

Our primary interest is in the design of efficient forwarding structures for a given workforce. While the problem of efficiently *incentivizing* a workforce (Horton and Chilton, 2010; Ho et al., 2012; Ghosh and McAfee, 2012) is also interesting and important, we separate it from our concerns here. It is perhaps fair to think of our results as applicable to settings where workers are either salaried employees willing to attempt the tasks given to them, or are paid a fixed amount per task, as is common on Amazon's Mechanical Turk. In the latter case we can view a worker's ability value as a conflation of their true underlying ability, and their willingness to perform quality work at the offered rate.

For our notion of efficient workforce organization, we consider a bicriteria approach. More specifically, we seek to find forwarding structures that simultaneously keep both *maximum workload* and *depth* small. Maximum workload measures the largest workload of any worker in the workforce under a given forwarding structure, while depth measures the number of forwarding steps required before a task is solved. Small maximum workload ensures that no single worker is (unnecessarily) overburdened, and can be viewed as a sign of efficient use of a large workforce — we might hope (and indeed shall show) that under natural assumptions, individual workloads can diminish rapidly with the workforce population size n, meaning that the system is scaling well. Small depth is motivated by the desire to solve individual tasks as rapidly as possible by keeping forwarding chains short (which also minimizes expenditures if workers are paid per task). We note that these two criteria may be

[3] The important generalization to the multi-dimensional case carries a number of definitional and technical challenges, and is left to future work.

[4] For example, by standard arguments if we draw $O(\log(n)/\epsilon^2)$ tasks at random and give them to all n workers, we can estimate all the w_i to within an additive ϵ, which suffices for our results.

in tension with each other, leading to tradeoffs. For example, if we optimize only for depth we would choose to give all tasks to our best worker, who would then have the highest possible workload. It is also easy to create examples in which optimizing the workload results in large depth. We are interested in the problem of designing forwarding structures that approximately minimize both workload and depth, and the types of structures required to do so.

We consider two variants of the framework above that differ in precisely how they measure workloads. In the *Pay-to-Forward (P2F)* model, the only way for workers to discover whether they are able to solve a given task is to attempt it, which then counts against their workload. In other words, in the P2F model, attempting a task and failing is just as onerous as solving it. In the *Free-to-Forward (F2F)* model, we assume that workers are able to quickly assess whether they are able to solve a given task *without* actually attempting it, and thus tasks they must forward do not count against their workload. Mathematically, in the P2F model a given task is charged against the workload of *every* worker along the forwarding chain that ends in a worker able to solve it, whereas in the F2F model only this final successful worker's load is charged.[5]. We note that a number of our results are the same or similar in both models, but there are sufficient differences that it is worth identifying them separately.

As more concrete motivations for these two variants, first consider a crowdsourcing system in which workers are asked to locate specific objects in images (Salek et al., 2013). While task difficulty and worker ability may vary considerably, no worker can simply glance at an image and immediately determine whether they will succeed or fail — a certain amount of time must be spent studying and scanning the image either way. In this example the P2F model is appropriate. In contrast, consider a system in which workers are asked to judge the accuracy of technical articles. A worker whose specialty is computer science might be able to immediately see that they have been assigned an article on organic chemistry, and

[5]Note that the acronyms could equally well stand for "Pay to Fail" and "Free to Fail".

know they cannot solve the task before investing any effort. In this case the F2F model seems more fitting.

Results: Omniscient Workload. We begin our results by considering the optimization of the maximum workload, regardless of depth.[6] We first consider the optimal *omniscient* algorithm for task assignment that is allowed to observe the actual difficulty x of each arriving task, and assign it to any worker in the worker population W capable of solving it; since workloads in the weaker P2F and F2F models can clearly only be higher, the optimal omniscient algorithm provides an important baseline or lower bound. For any W and task difficulty distribution P, we give a precise characterization of the optimal omniscient maximum workload $M_{W,P}$, and give an efficient algorithm for computing its forwarding policy.

Results: P2F and F2F Workloads. We then consider how closely $M_{W,P}$ can be approximated in the P2F and F2F models, and by what types of forwarding structures. Following a preliminary result showing that it always suffices to consider probabilistic DAGs as the forwarding structure in both models, we then show that in the F2F model, $M_{W,P}$ can always be achieved, and give an efficient algorithm for computing the witnessing DAG. We also show that the weaker forwarding structure of trees cannot always achieve $M_{W,P}$ in F2F in general, and that $M_{W,P}$ cannot always be achieved in the P2F model regardless of the structure.

Results: Near-Optimal Depth-Workload Tradeoffs. We conclude by giving our main positive approximation results for both models in the bicriteria setting. More specifically, we show that by hierarchically arranging the workers in balanced b-ary trees, we can simultaneously come within a multiplicative factor of b^2 of $M_{W,P}$ in terms of maximum workload, with a resulting depth of $\frac{\log(n)}{\log(b)}$ in both the P2F and F2F models. Thus, if $b = 2$ we obtain a 4-approximation to the optimal omniscient workload with only $\log(n)$ depth. More gen-

[6] As noted above, optimizing depth alone is trivial, since we can simply assign every task to the best worker.

erally, we can tune b to decrease the depth at the expense of workload. We also provide a lower bound for this class of structures, showing that maximum workloads must be at least $\sqrt{b}M_{W,P}$ in the worst case. We apply the upper bound to the natural model where worker abilities are distributed as a power law, and show that the maximum workload diminishes with the population size, even for constant depth.

We freely acknowledge that the models studied here make a variety of strong and simplifying assumptions that prevent their immediate applicability to real-world crowdsourcing systems and other labor organization problems. These include:

- One-dimensional task difficulty; surely in reality most settings demand a multi-dimensional notion.
- Lack of detailed treatment of incentive issues.
- No modeling of variable costs for forwarding tasks.
- No modeling of weaker workers being able to solve any task, but with greater effort.
- No modeling of imperfect task completion.

Despite these simplifications, we shall see that there is already a fair amount to say in our streamlined model; we leave the consideration of the important extensions above (some of which we suspect are both conceptually and technically nontrivial) to future work.

Related Work To our knowledge, this work is the first to study a model of a workforce with varying ability, and to examine workload-depth tradeoffs and associated algorithmic and representational issues. There are, however, several tangentially related strands of existing research that study labor organization design and various other tradeoffs.

Organizational Design and Hierarchical Structures: There is a large body of work on the design and architecture of organizations; see (Hax and Majluf, 1981) for a survey. In most of these works, hierarchical organizational structures are considered and their properties are

studied. In (Garicano, 2000) the focus is on the tradeoff between communication and knowledge acquisition costs. In (Cremer et al., 2007), a theory of optimal organizational languages is developed, and a tradeoff between facilitating internal communication and encouraging communication with other organizations is identified. The authors study the optimal communication structure and language, and the organizational structure that supports them. (Ferreira and Sah, 2012) studies organizations with individuals whose expertise differ in content and breadth, and derives implications for the trade-off between specialization of knowledge and communication costs, the role of top and middle managers, and the optimal design of hierarchies. (Prat, 1997) studies the properties of a hierarchy of processors where the organization must pay each employed processor a wage which is an increasing function of the processor's capacity.

Crowdsourcing: While our models are not specifically limited to crowdsourcing settings, they are related to the growing literature on this subject, as we too study efficient ways of organizing and utilizing a large pool of workers. We discuss some of the works more relevant to our interests. In (Salek et al., 2013), the authors employ a model that shares our emphasis on the variability of task difficulty and worker ability. They propose a probabilistic graphical model to localize objects in images based on responses from the workforce, and improve upon natural aggregation methods by simultaneously estimating the difficulty and skill levels. In (Ho and Vaughan, 2012), the goal is to assign tasks to a set of workers with unknown skill sets in a way that maximizes the benefit to the crowdsourcing system. The authors show that the algorithm they propose to this end is competitive with respect to the optimal offline algorithm which knows the skill levels when the number of workers is large. This model differs from ours in that its focus is on workers with unknown skill levels who arrive in an online fashion, and does not consider depth-workload tradeoffs. In (Horton and Chilton, 2010) the authors study the reservation wage one needs to offer to a workforce to incentivize them to perform the task. In (Ho et al., 2012), incentive mechanisms are devised to discourage workers from putting in as little effort as possible, and crowdsourcers from denying payments. In (DiPalantino and Vojnovic, 2009) and (Ghosh and McAfee, 2012),

a game-theoretic model of crowdsourcing is proposed, in which workers act strategically and seek to maximize their total benefit. (Karger et al., 2011) studies achieving a desired level of reliability cheaply, using the fact that crowdsourcers often increase their confidence in the result of crowdsourcing by assigning each task multiple times and combining the results. (Zhang et al., 2012) studies mechanisms for task routing that aim to harness people's abilities to both contribute to a solution and to route tasks to others, who they know can also solve and route. See also (Law and Ahn, 2011) for a thorough discussion of different task routing methods.

Load Balancing. Somewhat indirectly related to our work is the literature on load balancing, where the goal is to distribute jobs among a set of machines to optimize the overall performance of the system; for instance, see (Azar et al., 1999; Adler et al., 1995; Ghosh et al., 1999). Our model differs from these works in that task assignments are driven by worker abilities, while in the load balancing scenario, the decision to assign a task to a machine is usually based on its current workload only. Also, in our model there are no arrival and departure times defined for the tasks, rather they stay in the system until they are solved.

2.1.1. Model and Preliminaries

In our model, an organization will consist of a set of n workers $W = \{1, 2, \ldots n\}$, where the *ability* of worker i is denoted by $w_i \in [0, 1]$; we use i and w_i interchangeably to denote workers. The ability of a worker determines the *difficulty* of the tasks she is capable of solving: a worker with ability w can solve all the tasks with difficulty less than or equal to w [7]. We assume without loss of generality that $w_1 \leq w_2 \leq ... \leq w_n = 1$, and the difficulty of every task lies in $[0, 1]$; the best worker w_n can thus solve any task.

Any task given to the organization has an *unknown* difficulty denoted by $x \in [0, 1]$ and sampled from a known distribution P. Without loss of generality, we may represent the

[7] Our notion of ability can also incorporate some unobservable mixture of a worker's actual ability, and the quality of work they are willing to do at a given rate — for instance, a high-ability worker may choose to be lazy at the given rate and do no or low-quality observed work.

task distribution P by $\langle A_1, A_2, ..., A_n \rangle$ where $A_i = Pr_{x \sim P}[w_{i-1} < x \leq w_i]$ is the mass of tasks solvable by w_i but not by w_{i-1} (where we define $w_0 = 0$).

In order to solve a given task, an algorithm assigns it to one of the workers, who attempts to solve it. If the worker is unsuccessful, the algorithm forwards it to another worker of greater ability. This chain of forwarding continues until the task is solved by the first worker of sufficient ability. Note that the most general notion of forwarding would allow an arbitrary, centralized algorithm that (probabilistically) decides which worker to forward the task to, given the sequence of failures so far. However, as we shall show, it turns out that such an algorithm can always be represented by a decentralized probabilistic DAG over the workers themselves, and that in many cases near-optimal performance is possible with even simpler decentralized forwarding schemes, such as trees over the workers.

In this paper, we are interested in algorithms and representations for task forwarding, and their performance according to two criteria: workload and depth. We discuss and define workload first.

In the *Pay-to-Forward (P2F)* model, we assume that any worker who receives a task along a forwarding chain has their workload charged for attempting that task. Thus, for any algorithm A that determines how to forward tasks for W and P, we define the workload ℓ_i of worker i to be the probability that i receives a task drawn from P and forwarded according to A. In the *Free-to-Forward (F2F)* model, we assume that only the worker who actually *solves* the task is charged, and thus define the workload ℓ_i of worker i to be the probability that i is the *last* worker to receive a task drawn from P and forwarded according to A. Clearly for any A all workloads ℓ_i in the P2F model are greater than or equal to those in the F2F model.

As noted in the Introduction, the P2F model is appropriate for settings in which the only way workers can determine whether they are able to solve a task is to attempt it, which consumes the same effort whether they succeed or fail; and the F2F model is appropriate for

settings in which, despite the actual task difficulty value x being unobservable, workers can immediately assess whether they have the requisite skills to solve a problem, and forward it if not.

In both the P2F and F2F models, we define the *depth* of a forwarding algorithm A as the maximum number of workers in any forwarding chain of A under W and P. Our interests here are in forwarding algorithms and schemes that simultaneously achieve small maximum workload and depth. As noted in the Introduction, these two criteria may often be in conflict with each other, necessitating the tradeoffs that we study here. In general, we are most interested in cases in which workloads diminish rapidly with n, with depth growing only very slowly with n (logarithmic or even constant); we shall eventually see this is possible under fairly broad conditions.

2.1.2. The Omniscient Case

Regardless of W and P, it is clear how to minimize depth alone in task forwarding: simply assign every task to $w_n = 1$, who can solve all tasks (at the expense of the worst possible maximum workload of 1). In contrast, if we ask what the smallest possible maximum workload is, the answer depends strongly on W and P, and is far from obvious. Since we need to compare the workload performance of algorithms in the P2F and F2F models to some baseline, we consider an idealized *omniscient model*, in which a forwarding algorithm is actually allowed to observe the true task difficulty $x \sim P$, and immediately assign it to any worker capable of solving the task. Obviously workloads in the P2F and F2F models, where x is not observed, can only be worse than the workload-minimizing algorithm in the omniscient model. We now give a precise characterization of the optimal maximum workload in the omniscient model, and show that it can be computed efficiently. Perhaps surprisingly, later we shall see that this ideal can actually be achieved or well-approximated in the P2F and F2F models under fairly general circumstances.

Theorem 1 *In the omniscient model (and therefore in both the P2F and F2F models), any*

algorithm for task assignment has maximum workload greater than or equal to $\max_i \frac{\sum_{j=i}^{n} A_j}{n-i+1}$.

Proof Every task $x > w_{i-1}$, must eventually be solved by one of the workers $i, i+1, ..., n$. So at least one of these $(n-i+1)$ workers, say w, must have workload greater than or equal to $\frac{Pr[x>w_{i-1}]}{n-i+1} = \frac{(A_i+...+A_n)}{n-i+1}$. The maximum workload among all workers cannot be less than w's workload, therefore we have $\max_j \ell_j \geq \frac{(A_i+...+A_n)}{n-i+1}$. This holds for any i, therefore we can conclude $\max_j \ell_j \geq \max_i \frac{(A_i+...+A_n)}{n-i+1}$. ∎

We next show that in the omniscient model, there is an efficient algorithm for assigning the tasks that achieves this lower bound.

Theorem 2 *In the omniscient model, there is a task assignment algorithm whose maximum workload is equal to* $\max_i \frac{\sum_{j=i}^{n} A_j}{n-i+1}$*, and the assignment policy used by this algorithm can be computed in time* $O(n^2)$.

We omit the full proof due to space considerations but we sketch the algorithm here. The algorithm uses a policy that determines how to distribute tasks among workers, that is, it computes the probability with which a task $w_{i-1} < x \leq w_i$ is given to worker j $(j \geq i \geq 1)$. It does this inductively from the hardest to easiest tasks, always maintaining the invariant that the workloads of workers above the current difficulty level are equal. Note that the (pre-)computation of this policy occurs only once, and then is used to assign arriving tasks.

The policy computation first sets all the workloads to 0. If a task is in $(w_{n-1}, w_n]$, there is no choice other than giving it to worker n. So the current workload of worker n is increased to A_n. Now if a task is in $(w_{n-2}, w_{n-1}]$, either worker n or worker $n-1$ must eventually solve it. If $A_{n-1} \leq A_n$, the task is given to worker $n-1$ with probability 1, making its current workload A_{n-1}. If not, in order to minimize the maximum current workload, we split A_{n-1} between these two workers such that their current workload becomes equal. Similarly at each step when we are deciding how to distribute a task in $(w_{i-1}, w_i]$, we do it in a way that the maximum current workload is minimized. This continues until all the tasks assignment probabilities are computed. In Figure 1, a visualization of this procedure is given.

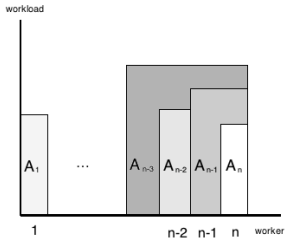


Figure 1: Workload-optimizing assignment in the omniscient model.

For the remainder of the paper, we denote the maximum workload of the omniscient algorithm on a distribution P, by $M_{W,P}$, which we have established is equal to $\max_i \frac{(A_i+...+A_n)}{n-i+1}$.

2.1.3. F2F and P2F: DAGs Suffice

In this section, we present a useful result that we shall employ in the subsequent sections. We show that in both the F2F and P2F models, the class of arbitrary randomized forwarding algorithms is no more powerful than the class of probabilistic DAGs, with respect to both workload and depth.

We require the following notation. In a DAG (and therefore a tree) structure, we denote the probability that a task is initially assigned to worker i by I_i, and thus $\sum_{i=1}^{n} I_i = 1$. In addition, the weight on edge uv, denoted by $p_{u,v}$, specifies the probability that worker u forwards a given task to v, in the case she fails to solve it herself. Thus $\sum_{v=u+1}^{n} p_{u,v} = 1$. Note that throughout, when talking about a "structure", whether DAG or tree, we not only refer to the specific network structure, we also consider the edge weights (if applicable) and initial probabilities on the vertices to be part of the structure.

Theorem 3 *In both the F2F and P2F models, for any randomized task forwarding algorithm A, there is a probabilistic DAG G such that the workload of every worker, and the depth, are the same in A and G.*

Proof Observe that upon initialization, A has no information about the value of the given task x, and therefore it must have a fixed distribution P_0 over the first worker the task is given to. Now consider the point where A has forwarded the task to $k \geq 1$ workers

$w_{i_1} < w_{i_2} < ... < w_{i_k}$, they have all failed, and A must decide who to forward the task to in the next step.

Note that the only information A has about x so far is that $x \in (w_{i_k}, w_n]$ and is distributed with respect to P in that range, since the information previous workers provided about x by failing is subsumed by the fact that w_{i_k} failed to solve it, that is:

$$Pr_{x\sim P}[x \mid (x > w_{i_1}) \wedge (x > w_{i_2}) \wedge ... \wedge (x > w_{i_k})] = Pr_{x\sim P}[x \mid (x > w_{i_k})]$$

So A has to forward the task based on a fixed distribution, call it P_{i_k}, over $w_{i_k+1}, w_{i_k+2}, ..., w_n$. It is now easy to see that A can be represented as a DAG: Let G be a complete DAG[8] in which for every worker u and every worker v with higher ability than u, the weight on edge uv is the probability that A forwards a task to v right after u's failure. This probability can be easily obtained from the fixed distribution P_u. Also the probability that a node gets selected in the initial step can be obtained from P_0. It is clear that on any given task, G and A have exactly the same forwarding behavior, and therefore the workload of every worker and the depth are the same in both of them. ∎

2.1.4. Workload-Minimizing Structures

We now consider the optimization of workload alone in the P2F and F2F models. We compare the workloads of optimal DAGs and trees with the omniscient algorithm, and show that in the F2F model there always exists an efficiently computed DAG with maximum workload equal to $M_{W,P}$. We show that that the same is not possible for trees. For the P2F model, we show that even the class of DAGs cannot always achieve maximum workload equal to $M_{W,P}$. We then proceed to consider workload-depth tradeoffs in Section 2.1.5.

The F2F Model

We first show that in the F2F model, there is a DAG that achieves maximum workload equal to the omniscient optimal $M_{W,P}$.

[8] A DAG is complete if in its topological sort, each node points to all the nodes with higher indices.

Theorem 4 *In the F2F model, there always exists a DAG G whose maximum workload is equal to $M_{W,P}$. In addition, there is an efficient algorithm for constructing G.*

Proof Consider the smallest index, say j, for which we have $\frac{(A_j+...+A_n)}{n-j+1} = \max_i \frac{(A_i+...+A_n)}{n-i+1}$. If there exists a DAG with maximum workload equal to $M_{W,P}$, then in that DAG, $w_j, ..., w_n$ should never get any task with difficulty $x \leq w_{j-1}$, otherwise at least one of them will have workload larger than $\frac{(A_j+...+A_n)}{n-j+1} = \max_i \frac{(A_i+...+A_n)}{n-i+1}$, resulting in max workload larger than $M_{W,P}$. An immediate consequence is that the initial probability of workers $j, ..., n$, i.e. $I_j, ..., I_n$ must be zero and the only node that can have edges with non-zero probability to them is worker $(j-1)$.

That being said, if $j > 1$, in order to build the optimal DAG, we do the following: first we build the optimal DAG G_1 for $w_1, ..., w_{j-1}$ and $\langle A_1, .., A_{j-1} \rangle$, then build the optimal DAG G_2 for $w_j, ..., w_n$ and $\langle A_j, ..., A_n \rangle$. To combine G_1 and G_2, we use the initial probability of vertex $i \geq j$ in G_2 as edge weight on the edge from worker $(j-1)$ to i, and we set I_i to 0. It is easy to see that combining the two DAGs this way results in the optimal DAG for $w_1, ..., w_n$ and P.

This suggests a recursive procedure for building the optimal DAG when $j > 1$. However, we still need to deal with the case where $j = 1$ and the workload must be divided equally among all workers.

For this remaining case $w_1, ..., w_n$ and $\langle A_1, ..., A_n \rangle$ are such that the omniscient algorithm gives every worker the same workload, equal to $\frac{(A_1+...+A_n)}{n}$. Algorithm 1 (which we call WED) presents an efficient procedure with running time $O(n^2)$ that generates a DAG with max workload $\frac{(A_1+...+A_n)}{n}$ for this case.

This algorithm is the recursive form of the following inductive construction:

Induction statement: for $k \in \mathbb{N}$ workers, if P is such that $M_{W,P} = \frac{(A_1+...+A_k)}{k}$, then there is a DAG G for which the maximum workload in G is equal to $M_{W,P}$.

ALGORITHM 1: Algorithm WED for Computing the Workload Equalizing DAG in the F2F Model

Data: A task distribution $P = \langle A_1, A_2, ..., A_n \rangle$ with $M_{W,P} = \frac{(A_1+...+A_n)}{n}$
Result: A weighted DAG G
$I_1 \leftarrow \frac{(A_1+A_2+...+A_n)}{nA_1}$;
$V(G) \leftarrow \{w_1\}$;
$E(G) \leftarrow \emptyset$;
if *n=1* **then**
  return;
else
  $P' \leftarrow \langle (1 - I_1)A_1 + A_2, A_3, ..., A_n \rangle$;
  $H \leftarrow$ WED(P') i.e. the workload optimal DAG for P';
  for $i \leftarrow 2$ **to** n **do**
    $I'_i \leftarrow$ the initial probability of worker i in H;
  end
  $V(G) \leftarrow V(G) \cup V(H)$;
  $E(G) \leftarrow E(H)$;
  Add edges from w_1 to every vertex in $V(H)$;
  for $i \leftarrow 2$ **to** n **do**
    $p_{1,i} \leftarrow I'_i$;
    $I_i \leftarrow (1 - I_1)I'_i$;
  end
end

Induction basis: For $k = 1$ workers, the optimal DAG is trivial. It is a single node and we have to give all the workload A_1 to the only worker available i.e. $I_1 = 1$.

Induction step: Suppose for a set of $k < n$ workers, we know how to build the optimal DAG to have the workload of every worker equal to $\frac{(A_1+...+A_k)}{k}$. Now suppose we are given a set of n workers with ability $w_1, w_2, ..., w_n$, and the task distribution $\langle A_1, .., A_n \rangle$ such that $M_{W,P} = \frac{(A_1+...+A_n)}{n}$.

Note that in order to have a DAG in which everyone has workload equal to $\frac{(A_1+...+A_n)}{n}$, this must hold for w_1 as well. Since w_1's workload is equal to $I_1 \times A_1$, I_1 must be equal to $\frac{(A_1+...+A_n)}{nA_1}$ then.

To build the rest of the optimal DAG, observe that for workers $w_2, ..., w_n$, there is no difference between a task in $[0, w_1]$ and one in $(w_1, w_2]$ i.e. all of them can solve both. Therefore we can assume the remaining mass on $[0, w_1]$ is actually on $(w_1, w_2]$ and find the optimal DAG for the remaining workers and task distribution.

According to the induction hypothesis, for the task distribution $\langle A_1(1-I_1)+A_2, A_3, ..., A_n \rangle$, we can find the optimal DAG say H, that gives equal workloads to $w_2, ..., w_n$. Suppose the initial probability on node i in H is I'_i. Now we construct G, in the following way: Add a vertex for w_1 to H and set the initial probability of this vertex to $\frac{(A_1+...+A_n)}{nA_1}$. Then set the probability on the edge from w_1 to w_i, to I'_i. Also set the initial probability of vertex $i > 1$ in G to $I_i = I'_i(1 - I_1)$.

We claim that in G, $w_1, ..., w_n$ all get equal workloads. For w_1 this obviously holds due to the way we chose I_1. For the nodes in H let's look at the task distribution they receive from outside of H, i.e. either initially or from w_1. With probability $(1 - I_1)$, a task is initially forwarded to a node in H; we know that such task, has distribution $\langle A_1, ..., A_n \rangle$. With probability I_1 the task is forwarded initially to w_1, and if w_1 forwards the task to some one in H, all we know about the task is that it is from the distribution $\langle 0, A_2, ..., A_n \rangle$. Combining the above distributions with respect to the probability that each happens, we

get the distribution $I_1\langle 0, A_2, ..., A_n\rangle + (1 - I_1)\langle A_1, A_2, ..., A_n\rangle = \langle (1 - I_1)A_1, A_2, ..., A_n\rangle$.

Now according to the induction hypothesis, H is optimal for this task distribution, giving $w_2, ..., w_n$ equal workload. Therefore we can conclude G is giving equal workload to everyone. ∎

We next show that in the F2F model, the weaker class of trees does *not* suffice to achieve maximum workload $M_{W,P}$.

Theorem 5 *In the F2F model, for $n > 2$ workers, there are worker abilities W and a task distribution P for which the maximum workload in any tree is strictly larger than $\frac{6}{5}M_{W,P}$.*

Proof Let $P_\epsilon = \langle A_1, A_2, A_3\rangle$ where $A_1 = \frac{1}{3}$, and $A_2 = (\frac{1}{3} + \epsilon)$ and $A_3 = (\frac{1}{3} - \epsilon)$ with $0 < \epsilon \leq \frac{1}{3}$. We claim that for this task distribution the optimal DAG is unique, and because this unique DAG is not a tree, we conclude that there exists no tree with the same (optimal) max workload.

Obviously $M_{W,P} = \frac{1}{3}$ in this case. To have a DAG with the same max workload, we must initially give every task to worker 1. So the workload of worker 2 is $p_{1,2}(\frac{1}{3} + \epsilon)$ which must be equal to $\frac{1}{3}$, therefore $p_{1,2} = \frac{1/3}{1/3+\epsilon} < 1$, and as a result $p_{1,3} > 0$, meaning that the optimal DAG is not a tree. It is easy to see that for $P_{\frac{1}{3}}$, the maximum workload of the optimal tree is $\frac{2}{5}$, proving the claimed lower bound. ∎

The above theorem confirms that in the F2F model, the class of trees is not as powerful as the class of DAGs in terms of the maximum workload. One question we would like to address is how large this gap can be. Later in Section 2.1.5, we will see that for any task distribution P, we can find a tree T such that the maximum workload in T is less than or equal to $4M_{W,P}$, yielding a constant-factor approximation to the optimal DAG.

The P2F Model

We next see that for the P2F model, even the optimal DAG cannot always achieve the omniscient optimal maximum workload. This already holds for a very simple case of two workers, and the task distribution $P = \langle \frac{1}{2}, \frac{1}{2} \rangle$.

Theorem 6 *In the P2F model, there are worker values W and a distribution P for which the maximum workload of the optimal DAG is strictly larger than $\frac{4}{3} M_{W,P}$.*

In Section 2.1.5 we will see that in the P2F model, for any task distribution P, the maximum workload in the optimal tree (and as a result optimal DAG), is less than or equal to $4M_{W,P}$.

2.1.5. Near-Optimal Depth-Workload Tradeoffs

Armed with a firm understanding of workload-only optimization in the omniscient model, and for DAGs and trees in the P2F and F2F models, we now finally turn our attention to workload-depth tradeoffs. We shall need the following definition:

Definition 1 *A* well-balanced b-ary *tree T is a tree with branching factor b ($1 \leq b \leq n-1$), such that:*

1. *Worker n is located at the root (first layer); the next b best workers are direct children of worker n; the next b^2 best workers are the children of the nodes in second layer, and so on. The worst workers are the leaves in the tree.*

2. *If we sort the nodes in any layer, say d, in a decreasing order, and then sort the children of each of those nodes in a decreasing order as well, it must be case that nodes in layer $(d+1)$ are properly sorted in a decreasing order.*

Less formally, we simply take a balanced b-ary tree, and place the workers from best to worst in a top-to-bottom, left-to-right fashion. It is easy to see that the depth of the well-balanced b-ary tree is $\frac{\log n}{\log b}$. One example of a well-balanced b-ary tree is illustrated in Figure 2 for $b = 3$.

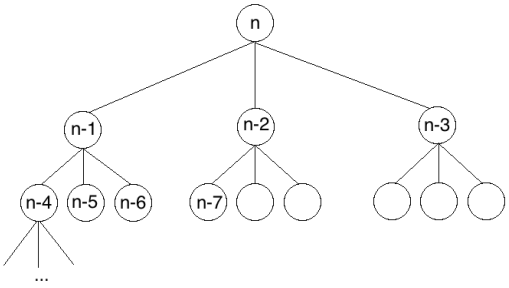


Figure 2: The b-ary tree structure with $b = 3$. Vertices are ordered by worker index, from the best (n, $w_n = 1$) down to the worst (1).

Unless otherwise specified, we assume that only leaves have non-zero probabilities of being initially assigned a task, and that these probabilities are such that every node at the same depth has the same probability of having a task initially assigned to a worker in their subtree.

We show that in a well-balanced b-ary tree of size n, not only do we achieve a depth that increases very slowly with n, the maximum workload we get is a constant factor approximation of $M_{W,P}$. The intuition for why such trees provide a good trade-off is that for a given depth, they (approximately) minimize the difference in ability between the task sender and the immediate receiver. In this way we avoid the possibility of missing too many qualified workers in between, which in turn could result in large maximum workload.

Theorem 7 *In the F2F and P2F models, for any task distribution P, the maximum workload in a well-balanced b-ary ($b \geq 2$) tree of depth $\frac{\log n}{\log b}$ is less than or equal $b^2 M_{W,P}$.*

Proof We prove the theorem for the P2F model. Since workloads are only lower in the F2F model, the theorem holds there as well.

Consider the well-balanced b-ary tree T where the only nodes with non-zero initial probabilities are the leaves. Note that here we assume $n = \frac{b^{m+1}-1}{b-1}$ ($m \in \mathbb{N}$), i.e. the last layer of the well-balanced tree is complete. If it is not the case, we can add sufficient number of workers with ability zero to the set of workers to make n of the desired form. Note that by doing so, we may at most multiply the number of workers by a factor b, but as those

additional workers cannot solve any task, $M_{W,P}$ remains the same. We also set the initial probability on every leaf to $\frac{1}{b^m}$.

We claim that for any P, in this tree the workload of each worker is less than or equal $b^2 M_{W,P}$. First note that our claim holds for the leaves in T. The workload of each leaf say l is equal to I_l. Since $I_l = \frac{1}{b^m} \leq \frac{b}{n} \leq b M_{W,P}$, we have $\ell_l \leq b^2 M_{W,P}$. Note that here we used the fact $M_{W,P} \geq \frac{1}{n}$ which holds by definition of $M_{W,P}$.

We shall make use of the following assertion.

Proposition 1 *In the P2F model, the workload of worker i in a tree T is equal to:*

$$\ell_i = I_i + \sum_{j \to i} H_j \times Pr_{x \sim P}[w_j < x]$$

where H_j is the probability that a task x is initially assigned to a worker in the subtree rooted at node j, and $j \to k$ means node j is a direct child of node k.

Observe that according to Proposition 1, for a non-leaf worker i, we can write the following:

$$\begin{aligned} \ell_i &= \sum_{j \to i} (H_j \times Pr_{x \sim P}[w_j < x]) \\ &\leq \max_{k \to i} Pr_{x \sim P}[w_k < x] \times \sum_{j \to i} H_j \\ &= Pr_{x \sim P}[w_{k^*} < x] \times \sum_{j \to i} H_j \\ &= Pr[w_{k^*} < x] \times H_i \end{aligned}$$

where $k^* = \arg\min_{k \to i} w_k$.

Using the above inequality, we next find an upper bound for the workload of a node at depth d in T. Note that at depth d, $H_i = \frac{1}{b^d}$. Also $k^* \geq i - b^{d+1}$. So:

$$\ell_i \leq H_i \times Pr[w_{k^*} < x] \leq \tfrac{1}{b^d} \times Pr[w_{i-b^{d+1}} < x].$$

Now note that since i has depth d, we have $i \geq n - (b^1 + b^2 + ... + b^d) = n - \frac{b^{d+1}-1}{b-1} + 1$.

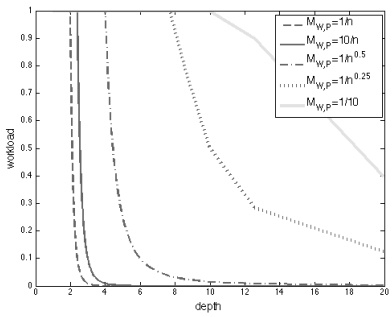


Figure 3: Illustration of Theorem 7.

Therefore $i - b^{d+1} \geq n - \frac{b^{d+2}-b}{b-1}$, and we can write:

$$\begin{aligned}
\ell_i &\leq \frac{1}{b^d} \times Pr_{x\sim P}[w_{i-b^{d+1}} < x] \\
&\leq \frac{1}{b^d} \times Pr_{x\sim P}[w_{n-b^{d+2}} < x] \\
&\leq b^2 \left(\frac{1}{b^{d+2}} \times Pr_{x\sim P}[w_{n-b^{d+2}} < x] \right) \\
&\leq b^2 M_{W,P}
\end{aligned}$$

where the last inequality follows from Theorem 1 since we have:

$$\begin{aligned}
M_{W,P} &= \max_r (A_r + ... + A_n)/(n - r + 1) \\
&= \max_r Pr_{x\sim P}[w_{r-1} < x]/(n - r + 1) \\
&\geq Pr_{x\sim P}[w_{n-b^{d+2}} < x]/(b^{d+2})
\end{aligned}$$

This completes the proof. ∎

In Figure 3 we illustrate the depth-workload tradeoff provided by Theorem 7. For a large value of n, each curve is parameterized by the branching factor b — i.e. each curve is traced out by varying b from n (smallest depth of 1) to 2 (largest depth of $\log(n)$. The x axis then measures the resulting depths, and the y axis the corresponding maximum workloads given by Theorem 7. The curves differ in the value of $M_{W,P}$ that is assumed, ranging from $1/n$ (rapidly diminishing omniscient optimal workload) to a constant independent of n. Clearly

smaller workloads are desirable; each curve can thus be thought of as providing an upper bound on the Pareto optimal curve for the corresponding bicriteria problem. We see that for small $M_{W,P}$, there is essentially threshold behavior of our bound — unless we choose a sufficiently large depth (small branching factor), our bound is vacuous, but then rapidly falls to 0. At larger $M_{W,P}$, the decay of workload with depth is more gradual.

The maximum workload upper bound given in Theorem 7 degrades with the branching factor b at a rate of b^2; it is not clear in general whether any such dependence is necessary. The following theorem, however, shows that at least $\sqrt{b}$ dependence is necessary within the class of balanced trees and the P2F model.

Theorem 8 *In the P2F model, for any $n \in \mathbb{N}$ and $b = 2, ..., (n-1)$, there is a task distribution P and a set of n workers W for which any well-balanced b-ary tree results in maximum workload greater than or equal $\frac{\sqrt{b}}{3} \times M_{W,P}$.*

Proof (Sketch) Given n and b, let P be the uniform distribution and let $w_i = 0$ for $i \leq (n - b - 1)$ and $w_i = \frac{(i-n+b+1)}{(b+1)}$. It is easy to see that in this case $M_{W,P} = \frac{1}{(b+1)}$. By solving a suitable optimization problem, it can be shown that for P, W, a well-balanced b-ary tree results in maximum workload larger than $\frac{1}{3\sqrt{b}}$, even if we allow the initial distribution of tasks over the leaves to be arbitrary, and also allow an arbitrary probability that the root receives an assigned task directly. ∎

Constant Depth and Diminishing Workloads

We conclude with an application of Theorem 7 to a natural parametric model of worker abilities that results in perhaps the best tradeoff we could hope for. Consider the case where $M_{W,P} = 1/n^\alpha$ for some constant $0 < \alpha \leq 1$ — that is, the maximum workload diminishes rapidly as a function of the workforce size n. (We shall shortly give natural assumptions on W for which this holds.) By choosing $b = n^\beta$, Theorem 7 yields maximum workload at most $b^2/n^\alpha = 1/n^{\alpha-2\beta}$. Thus as long as $2\beta < \alpha$, the well-balanced tree will also give an inverse

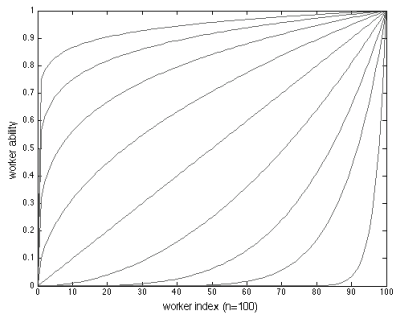


Figure 4: Worker ability as a function of index under the model $w_i = (i/n)^a$ for various a.

polynomial workload decay with n, while the depth $\frac{\log(n)}{\log(b)} = 1/\beta$ will only be *constant*.

For example, consider the case where $w_i = (\frac{i}{n})^a$ for some $a \geq 1$, and P is the uniform distribution. In this parametric family for W, if $a < 1$ we get concave improvement of workers with i, so successive workers are improving rapidly; while if $a > 1$ we have convex improvement, so the best workers may be far better than the average. See Figure 4 for examples.

Note that when $a > 1$ since $\{A_i\}_{i=1}^n$ is an increasing sequence here, $M_{W,P}$ is equal to A_n. And $A_n = 1 - (\frac{n-1}{n})^a \approx \frac{a}{n}$ using the Taylor expansion of the function $f(y) = y^a$. Thus we can immediately apply the observations above to obtain workloads bounded by $O(1/n^{1-2\beta})$ with only constant depth $1/\beta$.

Also for the case where $a < 1$ since $\{A_i\}_{i=1}^n$ is a decreasing sequence here, $M_{W,P}$ is equal to $\frac{1}{n}$. Applying Theorem 7, we obtain workloads bounded by $O(1/n^{1-2\beta})$ again with depth only $1/\beta$.

2.1.6. Discussion and Future Directions

Here are some interesting generalizations and open questions raised by the results presented here.

- Perhaps the most important generalizations would address all the unrealistic assumptions of our model mentioned in the Introduction: multi-dimensional difficulty, incen-

tive issues, variable task forwarding costs, imperfect task completion, and others.

- We showed that the optimal DAG in the P2F model is not necessarily as efficient as the omniscient algorithm in terms of maximum workload. One interesting problem is to determine how large the gap can be.

- The gap between the b^2 workload factor given in Theorem 7 and the corresponding lower bound of $\sqrt{b}$ given in Theorem 8 is large; can it be closed or improved?

- It would be interesting to conduct behavioral experiments designed to quantify the performance benefits of organizational schemes like those suggested here.

2.2. Limit Orders in Prediction markets

Buying a security is a statement that, in a trader's view, the security is underpriced compared with its expected payoff. This statement is not cheap talk; if the trader is wrong, she stands to lose money. A prediction market aggregates many statements of this form about securities whose payoffs correspond to events we would like to predict, such as election outcomes or product sales. With sufficient activity, prediction markets often outperform competing prediction methods (Berg et al., 2008). Moreover, continuous-trade markets (think of the stock market) provide real-time predictions that react with remarkable speed to news and information, such as a candidate's poor debate performance in an election race.

Prediction markets rely on trade. Without liquidity, a market faces a serious chicken-and-egg problem: a lack of trading opportunities discourages traders from participating, which in turn reduces future trading opportunities. For this reason, many prediction market mechanisms offer a subsidy to jump-start trade by providing incentives for the first trader to catalyze the process. A common form of subsidy involves an automated market maker built into the market institution. The marker maker provides guaranteed liquidity, offering to buy or sell any security at any time, at some changing price. For prediction markets, the standard market maker operates by keeping track of a *cost function* that defines how

much traders must pay to move the market from one state (vector of security quantities) to another. The cost function approach can be used to ensure that the market marker's worst-case loss is bounded, that trading is path independent, and that traders are unable to benefit from arbitrage (Hanson, 2003b; Chen and Pennock, 2007; Abernethy et al., 2013).

Even with a market maker, a trader may not see favorable transactions at current prices. If so, she may want to place a *limit order*, an offer to buy a security at a price equal to or less than a specified threshold. A limit order allows a trader to say "I won't pay the current price, but if it drops to ℓ or below, I want to buy." Limit orders are both a convenience, freeing traders from constantly monitoring for price changes, and a communication channel for traders, adding to the market's liquidity.

In this work, we provide and analyze a means of integrating limit orders into the cost function framework, which to date only accommodates market orders. We are concerned with *continuous markets*, in which trade orders may be placed, matched, and executed at any time, as opposed to *call markets*, in which orders are collected and executed at discrete, pre-specified times (e.g., once per hour). A continuous market maintains an order book of standing orders, waiting to be matched with arriving orders. It provides a complete, auditable record of trade execution over time. Both continuous and call markets present advantages, but the convenience of anytime trading has made the former the default in financial markets (Harris, 2002, Chapter 5).

In Section 2.2.1, we review the concepts of cost functions and limit orders, and describe the convex programming framework of Agrawal et al. (2011) for running call prediction markets. Their framework provides a building block for the continuous trade mechanism developed in this paper. Our approach also links with ideas of Hanson (2003a), who describes a method for continuously executing limit orders in the presence of a market scoring rule. In Section 2.2.2, we formalize Hanson's intuition by defining the notion of an ϵ-fair trading path: a path in security space along which no order executes at a price more than ϵ above its limit, and any order executes when its market price falls more than ϵ below its limit.

In Section 2.2.3, we show that under a supermodularity condition, a fair trading path exists for which the endpoint is efficient, in the sense of maximizing social welfare or gains from trade. The proof is constructive. Supermodularity holds, for example, for a standard market maker defined over disjoint securities, with limit orders allowed only on single securities. However, even a standard market maker may fail to satisfy the condition if bundle orders are allowed, or if the market is incomplete. We show that, under very general conditions, there exist market states and orders such that reaching an efficient endpoint via a fair trading path is not possible.

In Section 2.2.4, we develop the Fair Trading Path Algorithm for the general case (with or without supermodularity) for operating a continuous market maker with limit orders respecting the ϵ-fair bounds along the way. The algorithm iteratively solves for an efficient order fill of maximal volume, but limits the quantity of each order executed per step in order to stay on an approximately fair trading path and terminate at a state in which no standing limit order is profitable.

In Section 2.2.5, we conduct simulations of our algorithm in two settings: (1) a negative example in which no efficient fair trading path exists, and (2) an example constructed using combinatorial predictions made during the 2008 U.S. Presidential election. We evaluate our algorithm against a natural baseline according to trading volume, social welfare, and violations of the two fairness conditions. Section 2.2.6 concludes.

Related Work Our contribution can be viewed from a few different perspectives with respect to the literature. The one we emphasize is the integration of limit orders and cost functions, related to ideas outlined in a note by Hanson (2003a). Our contribution can also be viewed as a continuous-market counterpart to the call market of Agrawal et al. (2011).

Call prediction markets have long provided the ability to place limit orders. Building on the classic work of Shapley and Shubik (1977), Lange and Economides (2005) derive a pari-mutuel call market that accommodates both limit orders and bundle orders for binary-payoff

securities. Bossaerts et al. (2002) design an exchange mechanism called *combined value trading* that allows bundle orders and clears the market using linear programming, showing in laboratory experiments that the mechanism facilitates trade in thin markets. Fortnow et al. (2005) also use linear programming as the matching engine for a combinatorial call market for compound securities defined as arbitrary Boolean functions of an underlying state space.

In two remarkable papers, Peters et al. (2006) and Agrawal et al. (2011) provide a convex programming framework that subsumes much of this previous work, which they call the *convex pari-mutuel call auction mechanism*. The framework yields an efficient implementation of the Lange and Economides (2005) mechanism, and also introduces a market maker into the combined value trading LP. It is interesting to note that, in the evolution of call markets, limit orders were accommodated early on while market makers were only introduced more recently.

The reverse holds for the development of prediction markets with continuous trade. There we begin with the *market scoring rules* introduced by Hanson (2003b), and their equivalent cost function versions derived by Chen and Pennock (2007). Abernethy et al. (2013) extend the cost function framework to arbitrary-payoff securities, but still allow solely market orders. Recently, Chakraborty et al. (2015) have drawn on Hanson's ideas to integrate limit orders and cost functions in a market for a single security, empirically evaluating the mechanism with an emphasis on its price discovery properties.

2.2.1. Model and Preliminaries

We begin with a review of cost-function-based market making. We then introduce limit orders and explain how they are handled in call markets, specifically the convex programming framework of Agrawal et al. (2011).

Cost Functions

Let $\Omega = \{\omega_1, \ldots, \omega_N\}$ be a set of mutually exclusive and exhaustive states of the world. We consider a market that offers a fixed menu of K securities defined over these states. The payoffs of these securities are described by an arbitrary function $\boldsymbol{\phi} : \Omega \to \mathbb{R}^K$, with $\phi_j(\omega)$ specifying the value of security j if the final outcome is $\omega \in \Omega$.

The cost function framework (Chen and Pennock, 2007; Abernethy et al., 2013) consists of a continuous market where traders arrive sequentially and request bundles of shares from a centralized market maker. The model only accommodates market orders, though traders may query the cost of any bundle before placing orders. The market maker is always willing to buy or sell arbitrary security bundles $\boldsymbol{\theta} \in \mathbb{R}^K$, where θ_j denotes the number of shares of security j. The market maker charges traders according to a convex potential function C called the *cost function*. Let $\mathbf{q}$ be the current market state, with q_j denoting the number of shares of security j that have been bought from the market maker so far. A trader who requests bundle $\boldsymbol{\theta}$ is charged $C(\mathbf{q}+\boldsymbol{\theta}) - C(\mathbf{q})$. The instantaneous price of security j is then $p_j(\mathbf{q}) = \partial C(\mathbf{q})/\partial q_j$.

Choosing the cost function C fully determines the way in which market prices are set. Abernethy et al. (2013) give necessary and sufficient conditions on C for the resulting market to satisfy a set of nice properties including bounded loss for the market maker and a lack of arbitrage opportunities for traders. In particular, they show that C must be convex and that the derived price function $\mathbf{p}$ must map to the convex hull of the space $\boldsymbol{\phi}(\Omega)$, the image of the payoff function. They additionally show that any cost function of this form can be written as

$$C(\mathbf{q}) = \max_{\mathbf{p} \in \mathcal{H}(\boldsymbol{\phi}(\Omega))} \mathbf{p}^\top \mathbf{q} - R(\mathbf{p}) \tag{2.1}$$

for a strictly convex function R, where $\mathcal{H}$ denotes the convex hull. Furthermore, assuming

R is continuous and defined over $\mathcal{H}(\boldsymbol{\phi}(\Omega))$, the instantaneous price vector satisfies

$$\mathbf{p}(\mathbf{q}) = \nabla C(\mathbf{q}) = \arg \max_{\mathbf{p} \in \mathcal{H}(\boldsymbol{\phi}(\Omega))} \mathbf{p}^\top \mathbf{q} - R(\mathbf{p}). \tag{2.2}$$

We assume throughout the paper that the cost function takes this form, with an additional restriction on R. Abernethy et al. (2013) call R a *pseudo-barrier* function if $\lim_{t\to\infty} \|\nabla R(\mathbf{p}^t)\| = \infty$ for any convergent sequence $\{\mathbf{p}^t\}$ of points in $\mathcal{H}(\boldsymbol{\phi}(\Omega))$ whose limit is in the relative boundary of $\mathcal{H}(\boldsymbol{\phi}(\omega))$. Throughout the paper, we assume that R is a pseudo-barrier and bounded on $\mathcal{H}(\boldsymbol{\phi}(\omega))$. In particular, our results use the following lemma, which may be of independent interest. It shows that there is a fixed subspace over which a cost function obtained from a pseudo-barrier is linear at every point. The proof is in the appendix.[9]

Lemma 1 *Let C be a cost function defined via (2.1) with R a strictly convex pseudo-barrier function. If there exist bundles $\mathbf{q}, \mathbf{r} \in \mathbb{R}^K$ and a constant $\gamma \in \mathbb{R}$ such that $\gamma = C(\mathbf{q}+\mathbf{r}) - C(\mathbf{q}) = C(\mathbf{q}) - C(\mathbf{q}-\mathbf{r})$, then for all $\mathbf{q}' \in \mathbb{R}^K$ and all $\lambda \in \mathbb{R}$, $C(\mathbf{q}' + \lambda\mathbf{r}) - C(\mathbf{q}') = \lambda\gamma$. Additionally, for all $\omega \in \Omega$, $\boldsymbol{\phi}(\omega)^\top \mathbf{r} = \gamma$.*

One commonly used instance of a cost function market is the logarithmic market scoring rule (LMSR) (Hanson, 2003b, 2007). The LMSR is defined for complete markets and uses Arrow-Debreu securities: $K = N$, and for each $j = 1, \cdots, N$ the payoff is $\phi_j(\omega) = 1$ if $\omega = \omega_j$ and 0 otherwise. The LMSR uses cost function $C(\mathbf{q}) = b \log \sum_{k=1}^N \mathrm{e}^{q_k/b}$, resulting in instantaneous prices $p_j(\mathbf{q}) = \mathrm{e}^{q_j/b} / \sum_{k=1}^N \mathrm{e}^{q_k/b}$. Here $b > 0$ is a liquidity parameter controlling the trade-off between the rate at which prices change with trades and the market maker's worst-case loss. Lemma 1 applies to the LMSR because it is indeed based on a pseudo-barrier function (negative entropy) over the simplex of probability distributions. In this case the lemma establishes the well-known fact that the LMSR cost function is linear along the $\mathbf{1}$ vector at every market state.

[9]The appendix appears in the long version of this paper available on the authors' websites.

Limit Orders

In our setting, traders may place either market orders or limit orders. A limit order is specified by a triple $(\boldsymbol{\theta}, n, \ell)$. Here $\boldsymbol{\theta} \in \mathbb{R}^K$ is a security bundle, n is the number of shares of this bundle that the trader would like to purchase, and ℓ is the maximum price the trader is willing to pay per share. It is convenient to assume that $\boldsymbol{\theta}$ is normalized with either $\|\boldsymbol{\theta}\|_1 = 1$ or $\|\boldsymbol{\theta}\|_\infty = 1$, but this assumption is without loss of generality and our results do not depend on any specific normalization. Market orders can be represented as limit orders with $\ell = \infty$ (or the maximum payoff of bundle $\boldsymbol{\theta}$), so we may assume that all orders are limit orders. We allow for limit orders to be partially executed. That is, the trader may be sold x shares of the bundle $\boldsymbol{\theta}$ for any $0 \leq x \leq n$.

A limit-order market maintains an order book (a set of limit orders) to keep track of orders that have not yet been fully executed. In a call market, a new order book is provided at each trading session. In a continuous market, the order book contains orders that have accumulated over time without fully executing. When describing the behavior of a continuous market, we consider a single point in time in which the order book contains m existing orders $(\boldsymbol{\theta}_i, n_i, \ell_i)$ for $i = 1, \ldots, m$, and a new trader arrives with order $(\boldsymbol{\theta}_0, n_0, \ell_0)$. We record the orders in the matrix $\boldsymbol{\Theta} = [\boldsymbol{\theta}_0, \ldots, \boldsymbol{\theta}_m]$, where each bundle is a column of the matrix, and in the (column) vectors $\mathbf{n} = [n_0, \ldots, n_m]$ and $\boldsymbol{\ell} = [\ell_0, \ldots, \ell_m]$. Given an order book $(\boldsymbol{\Theta}, \mathbf{n}, \boldsymbol{\ell})$, an *order fill* $\mathbf{x} \in \mathbb{R}^{m+1}$ denotes the number of units of each order that are executed. An order fill is *feasible* if $\mathbf{0} \leq \mathbf{x} \leq \mathbf{n}$. We will refer to the 1-norm $\|\mathbf{x}\|_1$ of a feasible order fill as its *volume*. In the cost function framework, the instantaneous price of order i after executing fill $\mathbf{x}$, starting from initial market state $\mathbf{q}$, is given by

$$\pi_i(\mathbf{x}) \equiv \boldsymbol{\theta}_i^\top \mathbf{p}(\boldsymbol{\Theta}\mathbf{x} + \mathbf{q}) = \boldsymbol{\theta}_i^\top \nabla C(\boldsymbol{\Theta}\mathbf{x} + \mathbf{q}).$$

These instantaneous *order* prices should not be confused with the instantaneous *security* prices, which recall are given by $p_j(\mathbf{q}) = \partial C(\mathbf{q})/\partial q_j$ for each security $j = 1, \ldots, K$. Instan-

taneous order prices are in fact a linear combination of these instantaneous security prices at the market state $\mathbf{\Theta x} + \mathbf{q}$.

The various call markets in the literature all attempt to find an order fill $\mathbf{x}$ that controls the eventual payout $\boldsymbol{\phi}(\omega)^\top \mathbf{\Theta x} = \sum_{i=0}^{m} x_i \boldsymbol{\theta}_i^\top \boldsymbol{\phi}(\omega)$ across states of the world $\omega \in \Omega$. For instance, one might bound the maximum payout across states to be non-positive (so that there is no risk of loss), or no more than some fixed subsidy. If feasible, one might constrain the payout to always be zero. Various objectives can be layered on top of the constraints to select a particular order fill: maximizing volume, maximizing value to the traders (according to their limit prices), and so on (Rothschild and Pennock, 2014). Agrawal et al. (2011) propose a sophisticated convex program where the eventual payout is controlled by both a subsidy and a loss function over any amounts the subsidy does not cover. For our purposes, it is enough to state that their program is, as they prove, equivalent to

$$\max_{\mathbf{x}} \quad \boldsymbol{\ell}^\top \mathbf{x} - C(\mathbf{\Theta x} + \mathbf{q}) + C(\mathbf{q}) \tag{2.3}$$

$$\text{s.t.} \qquad \mathbf{0} \leq \mathbf{x} \leq \mathbf{n} \tag{2.4}$$

where C is a cost function of the form (2.1) and $\mathbf{q}$ is the current market state. We should note that the framework of Agrawal et al. (2011) is cast in terms of a complete market with Arrow-Debreu securities. We believe their framework extends to securities with general payoffs, but some of their results, such as the equivalence just mentioned, would require additional conditions. For our purposes this is immaterial because we build directly on top of the cost function framework.

One interpretation of the objective function (2.3) is as follows. If we take the limit prices as the traders' values per share, the first term in the objective captures the value of the order fill to the traders. Drawing on the known mathematical equivalence between cost functions and the convex *risk measures* that are used to quantify risk in mathematical finance (Föllmer and Schied, 2002; Othman and Sandholm, 2011), we can interpret $C(\mathbf{\Theta x} + \mathbf{q}) - C(\mathbf{q})$ as the change in the market maker's risk after filling the order $\mathbf{\Theta x}$. The objective is therefore the

value to the traders net of the risk incurred by the market maker, as quantified by its cost function. We focus on a second interpretation of this objective as the overall social welfare of traders in Section 2.2.2.

While the convex program prescribes an order fill, there are still several possible pricing schemes in a call market. The most standard is to use market clearing prices, which are formally obtained as the dual optimal solution to the program. Another possibility is to charge the limit prices (i.e., a first-price auction). The VCG payments in such an auction also have connections to cost-function payments (Agrawal et al., 2011). We will return to this efficiency-maximizing program as it will form the basis of our continuous market, and we will see that continuous order execution pins down a pricing scheme that is a hybrid of limit and cost function pricing.

2.2.2. Continuous Order Execution

Our interest is in algorithms that allow the integration of limit orders with cost-function based market making. More than a decade ago, before the cost function framework was well understood, Hanson wrote a brief note outlining an intuitively correct way of integrating limit orders with market scoring rules such as the LMSR (Hanson, 2003a). He described the following process (our notation in brackets replaces his):

> "Each book order with a limit $[\ell_i]$ imposes a constraint on the market maker prices $[\pi_i]$. It says that until that order expires, or all its quantity is used, the prices must satisfy $[\ell_i \leq \pi_i]$. This constraint is binding on a particular plane in the price vector space. Thus the above description of a new order trading with the market maker is only valid until the price vector reaches one of these planes. At that point the new order trades with both the market maker and the book order at the same time, moving the market maker prices in the plane of the book order. This continues until a price or quantity limit of one of the orders is reached, or until another book order plane is reached."

This idea can be illustrated through a simple example. Consider a complete market with two states and two Arrow-Debreu securities. Suppose trader 1 enters the market and submits a limit order $(\boldsymbol{\theta}_1 = (1, 0), n_1, \ell_1)$ on the first security. Suppose that the current market state $\mathbf{q}$ is such that the trade cannot be executed: $\ell_1 \leq p_1(\mathbf{q})$. The order would then go into the book. Now suppose trader 0 arrives and places a market order $(\boldsymbol{\theta}_0 = (0, 1), n_0, \ell_0 = \infty)$ on the second security. Following the intuition in Hanson's argument, the market maker would begin to execute this new order. Let τ be the quantity of the second security that could be traded before trader 1's limit price is hit, so $p_1(\mathbf{q} + \tau\boldsymbol{\theta}_0) = \ell_1$. If $\tau \geq n_0$, then the market order can be fully executed. Otherwise, the market maker would execute the first τ shares. At this point, the two traders would effectively "trade with each other" at the current market price. Since the market is complete, with any cost function from the class we consider, purchasing a bundle containing one of each security always costs \$1 and does not affect the instantaneous market prices (Chen and Pennock, 2007; Abernethy et al., 2013). Therefore, to "trade with each other," we can think of the traders purchasing $\min(n_0 - \tau, n_1)$ shares of the bundle $(1, 1)$ from the market maker and splitting them up so that trader 1 receives the shares of the first security at a cost of ℓ_1 per share and trader 0 receives the shares of the second at a cost of $1 - \ell_1$. If trader 0's order has not been fully executed, the trader may continue to trade with the market maker at this point.

If we alter this example so that there is a third security on a third state available, it already becomes trickier to reason about what should happen. Purchasing the bundle $(1, 1, 0)$ does not keep the price of the first security constant, complicating the question of how the traders should "trade with each other" once trader 1's limit price is hit. Matters get even more complex when we move to markets with larger state spaces, arbitrary securities for sale, and more traders. Hanson (2003a) does not provide an algorithm for determining which trades should be executed, only stating that iterative numerical methods would probably be required.

However, this example does provide principles that a market maker should strive to achieve

when implementing an order book on top of a cost-function based market. In particular, for any order fill, there should exist a corresponding continuous execution path such that for any intermediate order fill $\mathbf{x}'$ along that path, the orders in the book should be respected. This means that at any point $\mathbf{x}'$ along the path, if an order $(\boldsymbol{\theta}, n, \ell)$ has not yet been fully executed, we should have $\ell_i \leq \pi_i(\mathbf{x}')$; the book should not be crossed. Additionally, if the order is currently being executed at the point $\mathbf{x}'$, it should not be executed at a price higher than its limit price; for such $\mathbf{x}'$, we must have $\ell_i \geq \pi_i(\mathbf{x}')$. In the next section, we formalize these criteria.

Fair Trading Paths

We now formalize this intuition, leading to output criteria for an algorithm implementing limit orders on top of a cost-function based market maker.

Definition 2 (Trading path) *A* trading path *for order book* $(\boldsymbol{\Theta}, \mathbf{n}, \boldsymbol{\ell})$ *at initial market state* $\mathbf{q}$ *is a monotonically non-decreasing, continuous mapping* $\mathcal{P} : [0, 1] \to \mathbb{R}^{m+1}$ *with* $\mathcal{P}(0) = \mathbf{0}$ *and* $\mathcal{P}(1) \leq \mathbf{n}$.

A trading path describes the evolution of order execution. Note that only the image of the path matters—paths with the same image are equivalent. The domain is chosen as $[0, 1]$ only as a normalization. We next define an (approximately) fair trading path.

Definition 3 (Fair trading path) *A trading path* $\mathcal{P}$ *for orders* $(\boldsymbol{\Theta}, \mathbf{n}, \boldsymbol{\ell})$ *at initial market state* $\mathbf{q}$ *is an* ϵ-fair trading path *if*

1. *For all* $i \in \{0, \cdots, m\}$ *and any* a *and* b *such that* $0 \leq a < b \leq 1$, *if* $\mathcal{P}_i$ *is strictly increasing over* $[a, b]$, *then for all* $t \in [a, b]$

$$\pi_i(\mathcal{P}(t)) \leq \ell_i + \epsilon. \tag{2.5}$$

2. *For all* $i \in \{1, \cdots, m\}$*, for any* $t \in [0, 1]$ *such that* $\mathcal{P}_i(t) < n_i$*,*

$$\pi_i(\mathcal{P}(t)) \geq \ell_i - \epsilon. \tag{2.6}$$

If these conditions hold with $\epsilon = 0$*,* $\mathcal{P}$ *is called a* fair trading path*.*

The first condition captures the requirement that no trader should pay (too much) more than his limit price. The second captures the requirement that the market price of any unfilled order in the book can never fall (too far) below its limit price. Note that the second condition only applies to existing orders in the book. This reflects the pricing convention in continuous markets, which says that existing orders trade at their limit price while arriving orders trade at the best available price.

Let $\bar{\mathbf{x}} = \mathcal{P}(1)$ be the endpoint of a fair trading path. By integrating and aggregating order prices along the path, it is straightforward to show that the market maker collects a revenue of $C(\mathbf{\Theta}\bar{\mathbf{x}}+\mathbf{q})-C(\mathbf{q})$. This means that charging according to a fair trading path respects the principle that a cost function should reflect the revenue collected, assuming $C(\mathbf{0}) = 0$ (Chen and Pennock, 2007). From this viewpoint, the objective (2.3) has a simpler interpretation in terms of efficiency that does not use the notion of risk measure: the objective is the total utility to the traders (i.e., social welfare), where utility is the reported value (limit price) of a trade minus its cost as given by the cost function.

Integrating instantaneous prices along a fair trading path yields a unique, well-defined way to price orders. Note that when $\epsilon = 0$, the conditions together imply that orders in the book (i.e., excluding the new order) always execute exactly at their limit price. Thus each order $i \neq 0$ is charged $\ell_i \bar{x}_i$. Meanwhile, the new order 0 is charged

$$C(\mathbf{\Theta}\bar{\mathbf{x}} + \mathbf{q}) - C(\mathbf{q}) - \sum_{i \neq 0} \ell_i \bar{x}_i. \tag{2.7}$$

In some circumstances it may be unreasonable to charge a trader even a small amount

over his limit price, in which case condition (2.5) may be too weak of a requirement with $\epsilon > 0$. In practice, even when $\epsilon > 0$, a market maker could adopt the convention of charging exactly the limit price for orders in the book and charging the new order as in (2.7) as long as this does not exceed the new order's limit.

While the notion of a fair trading path yields a pricing scheme, note that the trivial trading path that does not execute any orders satisfies our definition. To obtain more interesting paths, we impose requirements on the endpoint.

Endpoint Criteria

We now derive conditions that should naturally signal the end of trading. In the process, we obtain further insights into the fair trading path conditions, and how one might construct a path satisfying them. A natural condition to impose is that the endpoint should be efficient (maximize reported social welfare). Let

$$F(\mathbf{x}; \mathbf{q}) \equiv \boldsymbol{\ell}^\top \mathbf{x} - C(\boldsymbol{\Theta}\mathbf{x} + \mathbf{q}) + C(\mathbf{q}) \tag{2.8}$$

be the objective function capturing the efficiency of $\mathbf{x}$. (For the sake of clarity we occasionally suppress the parameter $\mathbf{q}$.) Recall the convex program of Agrawal et al. (2011) introduced earlier:

$$\max_{\mathbf{x}} \quad F(\mathbf{x}; \mathbf{q}) \tag{2.9}$$

$$\text{s.t.} \quad 0 \leq x_i \leq n_i \quad (i = 0, \ldots, m) \tag{2.10}$$

Here order 0 is the arriving order and orders $1, \ldots, m$ are in the book before arrival (either completely unfilled or partially filled with n_i shares remaining). This is a straightforward convex program with box constraints. We say that a feasible order fill is *efficient* at state $\mathbf{q}$ if it maximizes this convex program. If $\mathbf{x}^*$ is an optimal (i.e., efficient) solution, then the optimality conditions for such programs (Boyd and Vandenberghe, 2009, p. 142) state that,

for each order $i = 0, 1, \ldots, m$,

$$x_i^* > 0 \Longrightarrow \pi_i(\mathbf{x}^*) \leq \ell_i, \tag{2.11}$$

$$x_i^* < n_i \Longrightarrow \pi_i(\mathbf{x}^*) \geq \ell_i. \tag{2.12}$$

See the appendix for a detailed derivation. Note that these conditions are closely related to conditions (2.5–2.6) in the definition of a fair trading path. For instance, if order i is partially filled at the optimum, so that $0 < x_i^* < n_i$, then $\pi_i(\mathbf{x}^*) = \ell_i$. Any further infinitesimal trade of the order would occur at its limit price, respecting the pricing scheme derived in the previous section.

However, we will see that efficiency is too stringent of an endpoint criterion. The reason is that fair trading paths must be monotone—orders cannot be reversed—and there may not exist a monotone path to an efficient solution respecting (2.5–2.6) for $\epsilon = 0$. Returning to the optimality conditions, (2.12) arises because it should not be possible to improve efficiency by further trading order i. Condition (2.11) arises because reversing order i should also not improve efficiency. But since we are enforcing monotonicity in the path, and orders cannot be reversed, the latter should not apply. Thus we say that an order fill is *complete* if for each order $i = 0, \ldots, m$ condition (2.12) holds. An efficient order fill is always complete, but not vice-versa. We will use the following equivalent characterization of completeness.

Definition 4 (Complete order fill) *An order fill* $\mathbf{x}$ *is* (maximally) complete *if* $\mathbf{0}$ *is a (unique) efficient fill at market state* $\mathbf{q} + \mathbf{\Theta x}$ *with quantities* $\mathbf{n} - \mathbf{x}$.

In more intuitive terms, an order fill is complete if, after executing it, it is efficient to no longer trade, and maximally complete if any further trade strictly reduces efficiency. The concept of a maximally complete order fill will prove important later for algorithmic correctness. Note that F is not strictly concave, because cost functions are not strictly convex. (For instance, the LMSR is linear at any point along the $\mathbf{1}$ direction.) Thus there may be multiple efficient order fills, or multiple complete order fills, and maximality further

selects among them.

2.2.3. Efficient Trading Paths

In this section we study conditions under which there exists a fair trading path that reaches an efficient order fill. The close connection between conditions (2.5–2.6) and conditions (2.11–2.12) suggests a few ideas for constructing such a path. First, one could simply take an efficient fill $\mathbf{x}^*$ and use a straight line from $\mathbf{0}$ to $\mathbf{x}^*$. While the fair trading path conditions would be met at the endpoint, and the path is monotone, there is no reason for the limit prices of orders in the book to match the instantaneous order prices at which they execute along the way. Alternatively, one could trace out a path by gradually filling order 0. More precisely, consider the correspondence

$$S(\alpha) = \arg\max_{\mathbf{x}} \{F(\mathbf{x}) \mid \mathbf{0} \leq \mathbf{x} \leq \mathbf{n}, 0 \leq x_0 \leq \alpha n_0\}, \tag{2.13}$$

where $\alpha \in [0, 1]$. If the correspondence admits a continuous, monotone selection, then by the optimality conditions for our convex program the result will be a fair trading path, and also an efficient path by construction. This section formally investigates whether and under what conditions such a construction is possible.

For the remainder of the paper, we will make the following standing assumption.

Assumption 1 *Before arrival of the new order, the previous order fill is maximally complete. Equivalently,* $\mathbf{0}$ *is the unique element in* $S(0)$.

The condition holds trivially if the order book is initially empty. Otherwise, one could start the continuous market with the call auction of Agrawal et al. (2011), ensuring that a solution of maximal volume is chosen. In fact, it is common practice in financial markets to initialize a continuous market via a call auction (Harris, 2002, Chapter 5). Our constructions in this section and the next will yield maximally complete fills, so that Assumption 1 continues to hold for all future arrivals.

Existence under Supermodularity

We show that a fair trading path reaching an efficient order fill exists when the objective function F is *supermodular*. Let $\mathbf{x} \wedge \mathbf{y}$ denote the component-wise minimum of vectors $\mathbf{x}$ and $\mathbf{y}$, and let $\mathbf{x} \vee \mathbf{y}$ denote their component-wise maximum. Note that for any two vectors $\mathbf{x}$ and $\mathbf{y}$ satisfying constraints (2.10), both $\mathbf{x} \wedge \mathbf{y}$ and $\mathbf{x} \vee \mathbf{y}$ are also feasible. Supermodularity is defined as follows.

Definition 5 (Supermodularity) *A function* $f : \mathbb{R}^n \to \mathbb{R}$ *is* supermodular *if* $f(\mathbf{x}) + f(\mathbf{y}) \leq f(\mathbf{x} \wedge \mathbf{y}) + f(\mathbf{x} \vee \mathbf{y})$. *A function* f *is* submodular *if* $-f$ *is supermodular.*

Note that supermodularity of the objective F depends on the cost function C but also the order bundles $\mathbf{\Theta}$. It is not sufficient for C to be submodular (but nor is it necessary). However, as an important special case, we have that F is supermodular if C is submodular and the bundles in $\mathbf{\Theta}$ do not overlap, where by "overlap" we mean intersect when neither is a subset of the other. For instance, we might use the LMSR, which is submodular, and restrict limit orders to singleton securities.

Supermodularity of F gives us the following result about S, the proof of which uses similar ideas as the proof of Topkis' Monotonicity Theorem; see Topkis (1978) or the excellent notes of Featherstone (2008).

Lemma 2 *Let* F *be supermodular, and let* S *be defined as in (2.13). For any* $\alpha, \alpha' \in [0,1]$ *with* $\alpha \leq \alpha'$*, for any* $\mathbf{x} \in S(\alpha)$ *and* $\mathbf{x}' \in S(\alpha')$*,* $\mathbf{x} \wedge \mathbf{x}' \in S(\alpha)$ *and* $\mathbf{x} \vee \mathbf{x}' \in S(\alpha')$*.*

Proof For $\alpha \in [0,1]$, define $D(\alpha) = \{\mathbf{x} \mid \mathbf{0} \leq \mathbf{x} \leq \mathbf{n}, x_0 \leq \alpha n_0\}$. We will make use of the following three facts about D, which can be verified easily:

1. $\alpha \leq \alpha' \implies D(\alpha) \subseteq D(\alpha')$.
2. $\mathbf{x} \in D(\alpha), \mathbf{x}' \in D(\alpha) \implies \mathbf{x} \vee \mathbf{x}' \in D(\alpha)$.
3. $\mathbf{x} \in D(\alpha), \mathbf{x}' \in \bigcup_{\alpha' \in [0,1]} D(\alpha') \implies \mathbf{x} \wedge \mathbf{x}' \in D(\alpha)$.

Now, consider any $\alpha, \alpha' \in [0,1]$ with $\alpha \leq \alpha'$, and any $\mathbf{x} \in S(\alpha)$ and $\mathbf{x}' \in S(\alpha')$. We have

$$0 \geq F(\mathbf{x} \vee \mathbf{x}') - F(\mathbf{x}') \geq F(\mathbf{x}) - F(\mathbf{x} \wedge \mathbf{x}') \geq 0.$$

The first inequality follows from Facts 1 and 2 about D and the optimality of $\mathbf{x}'$ at α'. The second inequality follows from the supermodularity of F. The third follows from Fact 3 about D and the optimality of $\mathbf{x}$ at α.

Since this chain of inequalities starts and ends with 0, the inequalities must all hold with equality, implying that $F(\mathbf{x} \vee \mathbf{x}') = F(\mathbf{x}')$ and $F(\mathbf{x}) = F(\mathbf{x} \wedge \mathbf{x}')$. Since $\mathbf{x}' \in S(\alpha')$ and $\mathbf{x} \vee \mathbf{x}' \in D(\alpha')$ by Fact 2, $\mathbf{x} \vee \mathbf{x}' \in S(\alpha')$. Since $x \in S(\alpha)$ and $\mathbf{x} \wedge \mathbf{x}' \in D(\alpha)$ by Fact 3, $\mathbf{x} \wedge \mathbf{x}' \in S(\alpha)$. ∎

We are now ready to state our result for the existence of efficient trading paths. The main construction in its proof is based on the correspondence (2.13). Lemma 2 implies that under supermodularity, each $S(\alpha)$ is a *sublattice* of the feasible set, allowing us to form a path by choosing the unique maximum lattice element for each α. The main details of the proof consist of establishing monotonicity and continuity of this path.

Theorem 9 *Let C be a cost function obtained via (2.1) from a pseudo-barrier function and suppose that F is supermodular in $\mathbf{x}$. Then there exists a fair trading path $\mathcal{P}$ for $(\boldsymbol{\Theta}, \mathbf{n}, \boldsymbol{\ell})$ at state $\mathbf{q}$ such that $\mathcal{P}(1)$ is efficient.*

Proof We define a particular function $\mathcal{P}$ and show that it satisfies the definition of a trading path $\mathcal{P}$ for $(\boldsymbol{\Theta}, \mathbf{n}, \boldsymbol{\ell})$ at state $\mathbf{q}$ and that its endpoint $\mathcal{P}(1)$ is efficient. We use Lemma 2 to show the path is non-decreasing and the Maximum Theorem and Lemma 1 to show continuity. Finally, we use the optimality conditions (2.11–2.12) to show that $\mathcal{P}$ is a fair trading path.

By Lemma 2, for any α, $S(\alpha)$ must be a *sublattice*, i.e., for any $\mathbf{x}, \mathbf{x}' \in S(\alpha)$, $\mathbf{x} \vee \mathbf{x}' \in S(\alpha)$ and $\mathbf{x} \wedge \mathbf{x}' \in S(\alpha)$. This implies that there is a unique maximum element in each set

$S(\alpha)$. We can therefore define the path $\mathcal{P}(\alpha) = \max\{\mathbf{x} \mid \mathbf{x} \in S(\alpha)\}$. We first show that $\mathcal{P}$ is a trading path with $\mathcal{P}(1)$ efficient. Assumption 1 immediately implies $\mathcal{P}(0) = 0$. By construction, $\mathcal{P}(1) \in S(1)$ and thus $\mathcal{P}(1) \leq \mathbf{n}$ and is efficient. To show $\mathcal{P}$ is a trading path, it remains to show that it is non-decreasing and continuous.

By Lemma 2, for any $\alpha, \alpha' \in [0,1]$ with $\alpha \leq \alpha'$, for any $\mathbf{x} \in S(\alpha)$ and $\mathbf{x}' \in S(\alpha')$, $\mathbf{x} \vee \mathbf{x}' \in S(\alpha')$. Since $\mathbf{x} \vee \mathbf{x}' \geq \mathbf{x}$, this implies that $\max\{\mathbf{y} \mid \mathbf{y} \in S(\alpha')\} \geq \mathbf{x}$ for all $\mathbf{x} \in S(\alpha)$, which implies that $\mathcal{P}$ is non-decreasing.

To show that $\mathcal{P}$ is continuous at any point α, it suffices to show that for all monotonically increasing or decreasing sequences $\{\alpha^i\}_{i=1}^{\infty}$ such that $\lim_{i\to\infty} \alpha^i = \alpha$, $\lim_{i\to\infty} \mathcal{P}(\alpha^i) = \mathcal{P}(\alpha)$. Suppose this were not the case. In particular, suppose that for some α and some increasing or decreasing sequence $\{\alpha^i\}_{i=1}^{\infty}$, $\lim_{i\to\infty} \alpha^i = \alpha$, but either $\lim_{i\to\infty} \mathcal{P}(\alpha^i)$ does not exist or $\lim_{i\to\infty} \mathcal{P}(\alpha^i) \neq \mathcal{P}(\alpha)$. Note that it cannot be the case that $\lim_{i\to\infty} \mathcal{P}(\alpha^i)$ does not exist since $\mathcal{P}$ is increasing and bounded. So suppose that $\mathbf{z} \equiv \lim_{i\to\infty} \mathcal{P}(\alpha^i) \neq \mathcal{P}(\alpha)$.

By Berge's Maximum Theorem, S is upper hemicontinuous. Since $\mathcal{P}(\alpha^i) \in S(\alpha^i)$ for all i, this implies that $\mathbf{z} \in S(\alpha)$ and therefore $\mathbf{z} < \mathcal{P}(\alpha)$ by construction of the path. (Here and throughout we use $\mathbf{a} < \mathbf{b}$ to mean that $a_i \leq b_i$ for all i, and $a_i < b_i$ for some i.) Let $\boldsymbol{\delta} = \mathcal{P}(\alpha) - \mathbf{z}$. Note that $\boldsymbol{\delta} > 0$. If the sequence $\{\alpha^i\}_{i=1}^{\infty}$ is decreasing, then for all α^i, $\mathcal{P}(\alpha^i) \geq \mathcal{P}(\alpha) = \mathbf{z} + \boldsymbol{\delta}$. But $\mathbf{z}$ cannot be bounded away from every $\mathcal{P}(\alpha^i)$ if $\mathbf{z} = \lim_{i\to\infty} \mathcal{P}(\alpha^i)$, a contradiction.

Suppose then that $\{\alpha^i\}_{i=1}^{\infty}$ is increasing. We first argue it must be the case that $\delta_0 = 0$. Suppose $\delta_0 > 0$. Since the objective F is concave, for any two points $\mathbf{x}, \mathbf{x}' \in S(\alpha)$ and any $\lambda \in [0,1]$, $\lambda\mathbf{x} + (1-\lambda)\mathbf{x}' \in S(\alpha)$. Therefore, $\mathbf{z} + \boldsymbol{\delta}/2 \in S(\alpha)$. For $\alpha \in [0,1]$, let $D(\alpha) = \{\mathbf{x} \mid \mathbf{0} \leq \mathbf{x} \leq \mathbf{n}, x_0 \leq \alpha n_0\}$ be the feasible set. Since $\mathbf{z} + \boldsymbol{\delta} \in D(\alpha)$, it must be the case that there exists some τ such that for all $i > \tau$, $\mathbf{z} + \boldsymbol{\delta}/2 \in D(\alpha^i)$. Furthermore, since $D(\alpha^i) \subseteq D(\alpha)$ for all i and $\mathbf{z} + \boldsymbol{\delta}/2$ is optimal for α, it must be the case that for all i, for all $\mathbf{x} \in D(\alpha^i)$, $F(\mathbf{x}) \leq F(\mathbf{z} + \boldsymbol{\delta}/2)$. Together these imply that for all $i > \tau$, $\mathbf{z} + \boldsymbol{\delta}/2 \in S(\alpha^i)$, and

so $\mathcal{P}(\alpha^i) \geq \mathbf{z} + \boldsymbol{\delta}/2$. But if this is true, we again cannot have $\mathbf{z} = \lim_{i\to\infty} \mathcal{P}(\alpha^i)$, another contradiction, so δ_0 must be 0.

Finally, consider the last case in which $\{\alpha^i\}_{i=1}^{\infty}$ is increasing and $\delta_0 = 0$. By the same argument above, $\mathbf{z} + \boldsymbol{\delta}/2 \in S(\alpha)$. Thus $F(\mathbf{x}) = F(\mathbf{x} + \boldsymbol{\delta}/2)$, and expanding out the definition of F, $C(\mathbf{q} + \Theta\mathbf{x} + \Theta\boldsymbol{\delta}/2) - C(\mathbf{q} + \Theta\mathbf{x}) = \boldsymbol{\ell}^\top\boldsymbol{\delta}/2$. Similarly, $F(\mathbf{x} + \boldsymbol{\delta}/2) = F(\mathbf{x} + \boldsymbol{\delta})$, so $C(\mathbf{q} + \Theta\mathbf{x} + \Theta\boldsymbol{\delta}) - C(\mathbf{q} + \Theta\mathbf{x} + \Theta\boldsymbol{\delta}/2) = \boldsymbol{\ell}^\top\boldsymbol{\delta}/2$. By Lemma 1, for any state $\mathbf{q}'$, $C(\mathbf{q}' + \Theta\boldsymbol{\delta}) - C(\mathbf{q}') = \boldsymbol{\ell}^\top\boldsymbol{\delta}$. Fix some i and let $\mathbf{y} = \mathcal{P}(\alpha_i)$. We must have $\mathbf{z} \geq \mathbf{y}$, and since $\mathbf{z} + \boldsymbol{\delta} \in D(\alpha)$ and $\delta_0 = 0$, $\mathbf{y} + \boldsymbol{\delta} \in D(\alpha_i)$. From the previous paragraph, we know that

$$F(\mathbf{y} + \boldsymbol{\delta}) = \boldsymbol{\ell}^\top(\mathbf{y} + \boldsymbol{\delta}) - C(\mathbf{q} + \Theta\mathbf{y} + \Theta\boldsymbol{\delta}) = \boldsymbol{\ell}^\top\mathbf{y} - C(\mathbf{q} + \Theta\mathbf{y}) = F(\mathbf{y}).$$

Since $\mathbf{y} \in S(\alpha_i)$, $\mathbf{y} + \boldsymbol{\delta} \in S(\alpha_i)$, and $\mathbf{y}$ could not be $\mathcal{P}(\alpha_i)$, another contradiction. This means that $\mathcal{P}$ must be continuous.

We have established that $\mathcal{P}$ is a trading path. The fact that the trading path $\mathcal{P}$ is a fair trading path follows immediately by comparing optimality conditions (2.11–2.12) with those in Definition 3. ∎

When There Is No Fair Trading Path

In the previous section, we showed that an efficient trading path can be constructed under certain favorable conditions. We now show that in general this is not the case. The following applies to a broad class of cost functions, including the LMSR cost function when traders request overlapping bundles.

Theorem 10 *Consider a complete market over $N \geq 3$ states of the world with N Arrow-Debreu securities. Let C be a cost function obtained via (2.1) with R bounded on $\mathcal{H}(\boldsymbol{\phi}(\Omega))$. There exists a set of orders $(\boldsymbol{\Theta}, \mathbf{n}, \boldsymbol{\ell})$, market state $\mathbf{q}$, and $\epsilon > 0$, such that no ϵ-fair trading path $\mathcal{P}$ for $(\boldsymbol{\Theta}, \mathbf{n}, \boldsymbol{\ell})$ at state $\mathbf{q}$ has an efficient endpoint $\mathcal{P}(1)$.*

The high-level idea behind the proof, which appears in the appendix, is as follows. Suppose that $N = 3$ and consider two traders with orders on bundles $\boldsymbol{\theta}_1 = (1/2, 1/2, 0)$ and $\boldsymbol{\theta}_0 = (0, 2/3, 1/3)$. Intuitively, purchasing $\boldsymbol{\theta}_0$ should simultaneously cause the price of security 1 to drop and the price of security 2 to (quickly) rise. Given an arbitrary cost function C, we use general properties of cost functions (*expressiveness* and *bounded loss*) to construct a specific start state $\mathbf{q}$, quantity n_0, and price ℓ_1 such that if the market state moves in a straight-line path from $\mathbf{q}$ to $\mathbf{q} + n_0\boldsymbol{\theta}_0$, the price of bundle $\boldsymbol{\theta}_1$ starts out above ℓ_1 due to a high price of security 1, temporarily drops (far) below ℓ_1 as the price of security 1 falls, and finally exceeds ℓ_1 again as the price of security 2 rises. By setting ℓ_0 very high, effectively making trader 0s order a market order, we force there to be a unique efficient fill that results in full execution of trader 0's order while trader 1's order is not executed at all. The corresponding (unique) trading path moves the market state in a straight line between $\mathbf{q}$ and $\mathbf{q} + n_0\boldsymbol{\theta}_0$, which, as described above, violates condition (2.6) for trader 1.

2.2.4. Constructing a Fair Trading Path

We now present an algorithm which, given a new order and an order book satisfying Assumption 1, constructs an approximately fair trading path terminating at a complete order fill. As in the construction of Section 2.2.3, the algorithm forms a path by gradually filling the new order, and filling standing orders when possible while respecting the fair trading path conditions (2.5–2.6).

The algorithm, given in Figure 5, consists of an outer and an inner loop. Each iteration of the outer loop solves for an efficient order fill of maximal volume, starting at the current state, subject to the requirement that the additional order volume (i.e., 1-norm) not exceed input parameter δ. The error ϵ by which the output path violates the fair trading path conditions is proportional to δ and properties of the underlying cost function, as shown below. The order fill of bounded volume is itself obtained through one or more iterations of the inner loop. At each iteration of the inner loop, a convex program is solved to find an efficient fill where the fill of the new order (order 0) is bounded according to β^t, where β is the

Parameters: $\delta > 0$, $0 < \beta < 1$.
Initialize: $\mathbf{x}^0 = \mathbf{0}$, $\mathbf{n}^0 = \mathbf{n}$, $\mathbf{q}^0 = \mathbf{q}$.
For $k = 0, 1, 2, \ldots$ do:

1. For $t = 0, 1, 2, \ldots$ repeat until $\|\mathbf{y}^k\|_1 \leq \delta$:

   (a) Formulate and solve the following convex program:

$$\begin{aligned} \max_{\mathbf{y}} \quad & F(\mathbf{y}; \mathbf{q}^k) \\ \text{s.t.} \quad & 0 \leq y_i \leq n_i^k \quad (i = 1, \ldots, m) \\ & 0 \leq y_0 \leq \beta^t n_0^k \end{aligned}$$

   From among the set of optimal solutions, set $\mathbf{y}^k$ to be a solution of maximum volume.

   (b) Set $\alpha^k = \beta^t$, update $t \leftarrow t + 1$.

2. Update $\mathbf{x}^{k+1} \leftarrow \mathbf{x}^k + \mathbf{y}^k$, $\mathbf{n}^{k+1} \leftarrow \mathbf{n} - \mathbf{x}^{k+1}$, $\mathbf{q}^{k+1} \leftarrow \mathbf{\Theta}\mathbf{x}^{k+1} + \mathbf{q}$, $k \leftarrow k + 1$.

Output: A path $\mathcal{P}$ with image $\left\{(1-s)\mathbf{x}^k + s\mathbf{x}^{k+1} \mid k \in \{0, 1, \ldots\}, s \in [0, 1]\right\}$.

Figure 5: The Fair Trading Path Algorithm (FTPA)

other input parameter. (Note the slight overload in notation: in the algorithm superscripts are used as round indices for iterates, with the exception of β^t where the t represents an exponent.) By only restricting the new order, we can guarantee the (approximate) fair trading path conditions—recall that (2.6) does not involve the new order. However, this means several inner iterations may be needed for the total order volume to fall below δ as a consequence of continuity. When it is efficient to no longer execute any trades, the algorithm has reached a complete order fill. The variable α^k is for book-keeping purposes in the proofs only.

In step (1a), we require not just an optimal solution to the convex program, but an optimal solution of maximal volume. The following result explains how to achieve this efficiently. Note that it depends on certain special properties of the class of cost functions we consider. The proofs for all results in this section are collected in the appendix.

Lemma 3 *Given an optimal solution to the convex program in (1a), an optimal solution of maximal volume can be obtained via linear programming.*

Let us provide some intuition for the lemma and describe the linear program in more detail. A consequence of Lemma 1 is that the directions along which the cost function C is linear form a subspace. We can therefore form a matrix $\mathbf{D} \in \mathbb{R}^{K \times k}$, where $k \leq K$, whose columns form a basis for this subspace. If the cost change along some combination of these directions equals the change in the limit-price term of the objective, then we obtain directions along which the objective is constant. Given an optimal solution $\mathbf{y}'$ to the convex program in step (1a), the LP for extending it to an optimal solution of maximum volume takes the following form:

$$\begin{aligned} \max_{\mathbf{y},\alpha} \quad & \|\mathbf{y}' + \mathbf{y}\|_1 \\ \text{s.t.} \quad & \mathbf{0} \leq \mathbf{y}' + \mathbf{y} \leq \mathbf{n}, \quad \boldsymbol{\Theta}\mathbf{y} = \mathbf{D}\alpha, \quad \boldsymbol{\ell}^\top \mathbf{y} = \mathbf{1}^\top \alpha \end{aligned}$$

The latter two constraints ensure that adding $\mathbf{y}$ does not change the objective value, and thus $\mathbf{y}'+\mathbf{y}$ remains an optimal solution. The fact that an LP can be formulated to maximize volume is used in proving correctness of the algorithm.

We now turn to the algorithm's properties. The first element of the proof is to show that the algorithm always progresses from one iteration to the next. For this it is important that in step (1a) we choose an optimal solution of maximal volume.

Lemma 4 *At every iteration $k = 0, 1, \ldots$ the inner loop in step (1) halts.*

From this lemma we obtain that the algorithm will generate an infinite sequence of iterates $\{\mathbf{x}^k\}$. The following gives the main property of this sequence.

Theorem 11 *The sequence of iterates $\{\mathbf{x}^k\}$ generated by the algorithm converges to a maximally complete order fill.*

There are several parts to this result. First, it states that the sequence of iterates converges. Often with numerical algorithms only convergence in objective value holds, not convergence in iterates (e.g., this is common in optimization methods when there are multiple optima).

Here we obtain convergence in iterates because we are monotonically filling the orders. Second, it states that the limit point is a complete order fill. There are two sub-cases: 1) at some point the sequence of iterates becomes constant, meaning that the algorithm has effectively converged in a finite number of steps, or 2) the algorithm generates an infinite sequence of distinct iterates. The first case occurs whenever the optimal solution obtained in step (1a) is $\mathbf{0}$, which gives one stopping criterion. The second case is not unusual for numerical algorithms (e.g., consider gradient descent), and allows for several stopping criteria: small enough change in iterates, in efficiency, etc. Under continuity of the objective's gradient, the result implies that condition (2.12) will eventually hold to any desired tolerance, which is perhaps the most natural stopping criterion for our purposes.

The next result concerns the path constructed by the algorithm. Note that even if the sequence of iterates is infinite, the output path is well-defined. Furthermore, because the iterates converge, it is straightforward to normalize the domain to [0,1] and take $\mathcal{P}(1) = \bar{\mathbf{x}}$, the limit point of the sequence.

Theorem 12 *Suppose the price function π is L-Lipschitz. Then the path $\mathcal{P}$ output by the algorithm is ϵ-fair for $\epsilon = \delta L$.*

The result requires Lipschitz continuity of the order price function. This holds if the cost function C has Lipschitz continuous gradients, and for cost functions of the form (2.1) that we consider, this is equivalent to R being strongly convex (Hiriart-Urruty and Lemaréchal, 2001, Thm 4.2.1).[10] The Lipschitz constant L for the price function depends on both the underlying constant for C, and the order bundle matrix $\mathbf{\Theta}$ (more specifically, its matrix norm). Therefore L will depend on the way bundles are normalized. Throughout, the Lipschitz parameter L refers to continuity with respect to the 1-norm.[11] The correctness of the algorithm and the previous result establish the following.

Corollary 1 *Suppose the price function π is L-Lipschitz. Then for any $\epsilon > 0$, there exists*

[10] For instance, the LMSR cost function is Lipschitz continuous with constant 1 with respect to the 1-norm.

[11] Because the number of dimensions is finite, Lipschitz continuity with respect to some norm implies the same for any norm. However, the parameter L depends on the norm chosen.

an ϵ-fair trading path for $(\boldsymbol{\Theta}, \mathbf{n}, \ell)$ at $\mathbf{q}$ terminating at a maximally complete order fill.

There are still some limitations to these results. We cannot guarantee that the algorithm will reach an efficient endpoint when an ϵ-fair trading path to one exists, let alone parametrize the algorithm to balance path fairness and endpoint efficiency. It is also not straightforward to apply a limiting argument to Corollary 1 and obtain existence of an exact ($\epsilon = 0$) and complete trading path.

Let us stress again that all our results rely on Assumption 1 holding at the outset. By Theorem 11, the assumption holds after running our algorithm upon each new order arrival. Thus we only need to ensure that it holds when the market is started, by using a call market or starting with an empty order book as suggested earlier.

2.2.5. Simulations

In this section, we investigate the empirical performance of the Fair Trading Path Algorithm (FTPA) against a simple baseline. The baseline takes as input the set of orders $(\boldsymbol{\Theta}, \mathbf{n}, \boldsymbol{\ell})$, market state $\mathbf{q}$, and a parameter $\delta > 0$, and tries to fill a small quantity of each order in cycle, as long as the trade is profitable. More specifically, for each order $i = 0, 1, \ldots, m$ that has not yet fully executed, let $\tau_i = \min\{\delta, n_i\}$ and check whether $C(\mathbf{q} + \tau_i \boldsymbol{\theta}_i) - C(\mathbf{q}) \leq \tau_i \ell_i$. If so, sell τ_i shares of $\boldsymbol{\theta}_i$ to trader i at this cost, update the current market state $\mathbf{q}$ to $\mathbf{q} + \tau_i \boldsymbol{\theta}_i$, and update n_i to $n_i - \tau_i$. Remove order i from the book whenever n_i reaches 0. Each execution of τ_i shares is filled in a straight line, forming a trading path. The baseline represents a simple and natural way to handle limit orders by executing them against the market maker in small pieces, and is similar to what some practitioners have done (Berg and Proebsting, 2009).

We evaluate the FTPA and baseline along four metrics:

- **Trade volume:** The volume (total number of executed shares) of the overall order fill after all orders have been processed.

- **Social welfare:** The efficiency after all orders have been processed.
- **Violation of condition (2.6):** The maximum amount by which the limit price of any unfilled order exceeds its market price along the trading path, potentially violating condition (2.6). Estimated by sampling points on the path.
- **Violation of condition (2.5):** The maximum amount by which the limit price of any order (excluding the new order) falls below its market price while the order is currently being executed, potentially violating condition (5). Estimated by sampling points on the path.

In our experiments we varied the maximum trade volume δ allowed at each step, as well as the liquidity parameter b. All our plots have δ on the x-axis in log scale, and one of the four metrics on the y-axis.

Two Trader Instance

We first consider one instance of the general counterexample to efficient trading paths constructed in the proof of Theorem 10. We use an LMSR cost function over three securities with liquidity parameter $b = 10$ and initial market state $q = (0, -60, -30)$. Trader 1 places a limit order with $\boldsymbol{\theta}_1 = (1/2, 1/2, 0)$, $n_1 = 100$, and $\ell_1 = 0.45$, which cannot be executed. Trader 0 then places a market order with $\boldsymbol{\theta}_0 = (0, 2/3, 1/3)$, $n_0 = 180$, and $\ell_0 = 1$. By the argument in the proof of Theorem 10, there is no ϵ-fair trading path with an efficient endpoint for small ϵ.

Results are shown in Figure 6. First examine the volume of trade. Running the FTPA with $\delta = n_0 + n_1 = 280$ (the rightmost point in the curve) is equivalent to calculating a straight-line path to the unique efficient fill. Only trader 0s order is executed in this case, resulting in a total of 180 shares executed. For all but the highest values of δ, the FTPA and the baseline execute a significant portion of trader 1s order, about 30 shares. As shown in the top right plot, this leads to very little change in efficiency. This is because trader 1

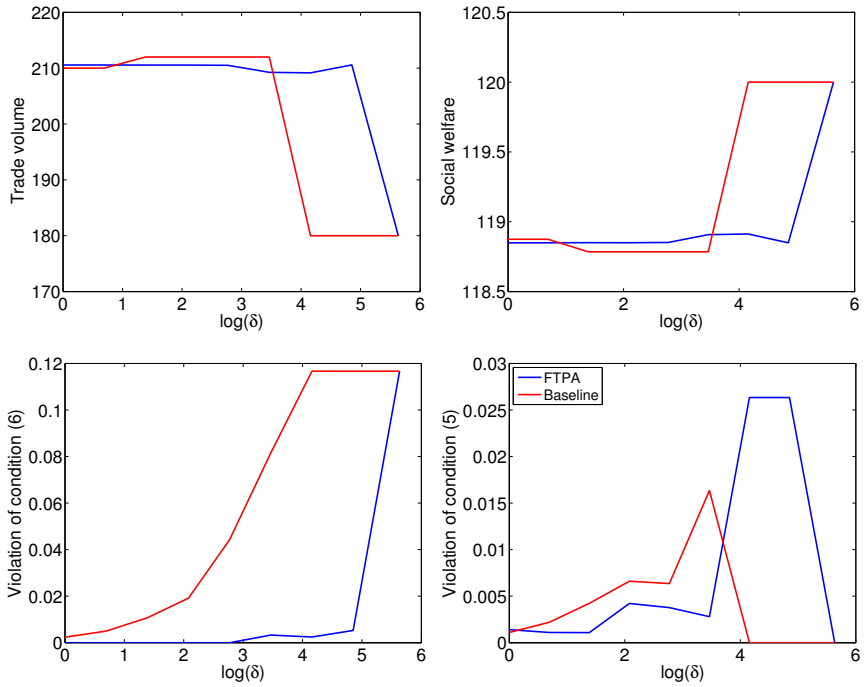


Figure 6: Algorithm performance on an instantiation of the two trader example from the proof of Theorem 10.

pays exactly his limit price for these shares. The small drop in social welfare for smaller values of δ is therefore due only to the fact that trader 0 pays slightly more when trader 1s order is partially filled.

The bottom left plot shows that for the FTPA, the maximum amount by which an order's limit price exceeds its market price along the trading path grows very slowly in δ, until δ reaches its maximum value of 280 when (as we know from the proof of Theorem 10) the market price of trader 1's bundle drops below 1/3 while $\ell_1 = 0.45$. This quantity grows more quickly in δ for the baseline. Finally, the bottom right plot shows that the maximum amount by which an order's market price exceeds its limit price during execution is relatively small for both algorithms and all values of δ.

Presidential Elections Market

We next generate trading data from survey results from the 2008 U.S. Presidential election gathered by the Princeton Election Consortium (Wang et al., 2011). This data consists of

probability forecasts on the outcome of the election in a single U.S. state (e.g., how likely it is that McCain wins in OH) or on a conjunction or disjunction of outcomes in up to three U.S. states (e.g., how likely it is that Obama wins in both OH and PA) provided by individuals across the U.S. The data was used by Dudík et al. (2012) to generate synthetic trades to evaluate their constraint-generation based market maker.

We restrict attention to forecasts that cover a set of 5 U.S. states on which there were a large number of predictions. We use the LMSR cost function for a complete market over the $N = 32\,(= 2^5)$ states of the world corresponding to the 32 possible election outcomes for the two major political parties in these five U.S. states. We generate one limit order from each forecast in the data. A probability forecast on the likelihood of an event E is converted into a request to purchase $n_i = 10$ shares of the bundle $\boldsymbol{\theta}_i$ consisting of one unit each of all securities corresponding to states $\omega \in E$. As the limit price ℓ_i we use the forecast probability of E. We discard forecasts of 0 from the data since orders with a limit price of 0 would never be executed. We run the FTPA and baseline on 1000 orders sub-sampled from this dataset.

It is well known that the performance of LMSR is sensitive to the choice of liquidity parameter b. In initial experiments, we found that with $n_i = 10$ for all i, $b = 1$ led to a reasonable amount of price movement: Prices moved quickly enough that orders were not fully executed immediately upon arrival, but slowly enough that orders in the book were often able to execute later. We chose to run our simulations with three different values of b near to this: 0.5, 1, and 2. The results, which are qualitatively similar for these three values, are shown in Figure 7. Although we plot all results together, results from runs using different liquidity parameters are not directly comparable. As for comparing the FTPA and baseline, both construct a piece-wise linear path that aims to faithfully approximate a fair trading path, and in both the parameter δ controls the maximum length of each piece, albeit in different ways. We therefore compare the two under the same setting of δ, varying δ. However, the baseline is also able to run for small values of δ where the FTPA takes

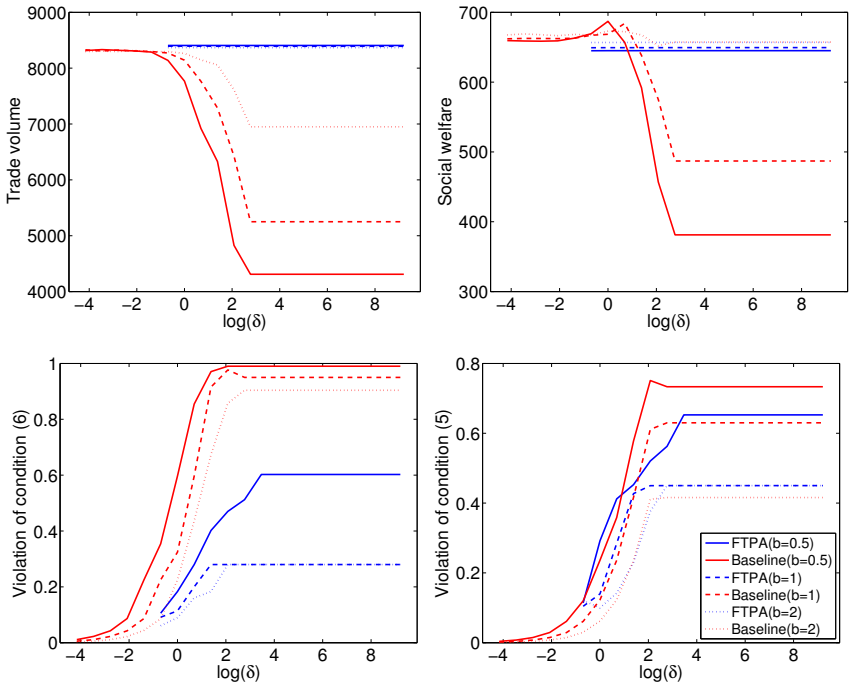


Figure 7: Algorithm performance on trades generated from survey data on the 2008 elections.

prohibitively long to finish; for completeness, we also include baseline results in this range to indicate the trends.

From the top left plot we observe that the FTPA leads to a higher trade volume for all comparable δ. Its trade volume is very stable, while trade volume for the baseline quickly degrades beyond a certain threshold. The same general pattern holds for social welfare, as seen in the top right plot. However, here we find that the baseline achieves the highest social welfare of both options at sufficiently small δ.

Observe that the baseline run with small values of δ achieves higher efficiency than the FPTA run with the largest value of δ, even though the latter executes arriving orders efficiently. This is not a contradiction: the FPTA is executing each arriving order efficiently in an online fashion, but this greedy approach is not necessarily efficient for the entire set of orders in batch.

The bottom two plots examine violations of the trading path constraints. The bottom right

plot shows that both approaches perform similarly when it comes to condition (2.5), except for larger δ where the FTPA has an advantage. The biggest discrepancy is for condition (2.6) in the bottom left plot—the baseline has a tendency to let market prices drift further below limit prices without executing the associated orders. This is arguably the most challenging condition to enforce, because it requires a global control of prices rather than just a local control of the price on the order being executed.

While the FTPA compares favorably to the baseline on the range of δ for which results are available, the trends clearly indicate that, for small enough δ, the baseline can achieve excellent social welfare and trade volume with minimal violation of the trading path conditions. Improving the FTPA so that it can scale to this range is an important avenue for future work.

2.2.6. Discussion and Future Directions

Even in a prediction market with a heavily subsidized market maker, a trader may not accept current prices. In that case, a limit order allows the trader to inject information in the market in the form of a constraint on prices, another form of liquidity. By merging cost-function based market makers with limit orders, we design a prediction market mechanism that is well suited for attracting trade. The first traders will interact mainly with the market maker at a low subsidy rate. As trade volume grows, more and more trades will effectively occur as bilateral matches between traders.

We formalized the conventions of continuous-trade markets in a general framework through the notion of a fair trading path, which monotonically fills orders and ensures that standing orders trade as the market prices reach their limit prices. We showed that arriving orders can be filled efficiently (in the sense of social welfare, or total trader utility) under supermodularity conditions, but that this is not possible more generally. We then provided a Fair Trading Path Algorithm which is guaranteed to construct a path respecting limit prices to within any required tolerance, and terminates at a state where no order is profitable.

The most immediate avenue for future work is to develop an algorithm that combines theoretical guarantees with efficient runtime in practice. It would also be worthwhile to perform more advanced simulations with traders that strategize in terms of their reported limit price or the timing of their trades. Another line of research would be to gain a deeper understanding, through theory and simulations, of the algorithm's price discovery properties when there is a common component to the traders' probability forecasts, or when there is a mix of informed and noise traders.

CHAPTER 3 : Learning with Repeated Interactions

An online market usually interacts with the same set of participants over time. The market and its participants can utilize the sequential nature of the interaction to *learn* about the other side and optimize their future actions accordingly. In this chapter, I address problems of this flavor in the context of *online labor markets* and *ad exchanges*[1].

We begin in Section 3.1 by studying how an ad exchange can take advantage of the data it accumulates over time about its bidders behavior. In an *online display ad exchanges* the exchange repeatedly interacts with quota-limited bidders, making real-time decisions about which subsets of bidders are called to participate in ad-slot-specific auctions. Given the repeated nature of the interaction with its bidders, the exchange has information about each bidders' segments of interest. This information can be utilized to design smarter callout mechanisms—with the potential of improving the exchange's revenue. Section 3.1 is a recent work by Azari et al. (2017), in which my collaborators at Google and I present an empirical framework for studying the performance of callout mechanisms in such settings.

More precisely, we present a general framework for evaluating and comparing the performance of various callout mechanisms using *historical auction data only*. To measure the impact of a callout mechanism on long-term revenue, we propose a strategic model that captures the repeated interaction between the exchange and bidders. Our model leads us to two metrics for performance: immediate revenue impact and social welfare. Next we present an empirical framework for estimating these two metrics from historical data. For the baseline to compare against, we consider random throttling, as well as a greedy algorithm with certain theoretical guarantees. We propose several natural callout mechanisms and investigate them through our framework on both synthetic and real auction data. We characterize the conditions under which each heuristic performs well and show that, in addition to being computationally faster, in practice our heuristics consistently and significantly

[1]The content of Sections 3.2 and 3.1 is taken directly from (Heidari et al., 2016a) and (Azari et al., 2017), respectively.

outperform the baselines.

Next in Section 3.2 and motivated by workers' dynamic perfomance in online labor markets, we study a setting where the behavior of market participants changes over time. In (Heidari et al., 2016a), my coauthors and I propose a stylized model of learning and performance improvement for workers in a crowdsourcing platforms. Our model is a variant of the well-studied multi-armed bandit problem in which the reward from each arm (worker) evolves monotonically in the number of times that arm is pulled (a task is assigned to that worker). We assume that the arm-dependent rates at which the rewards increase (workers' learning rates) are unknown, and propose task assignment algorithms with provably optimal *policy regret* bounds, a much stronger notion than the often-studied external regret. For the case where the rewards are increasing and concave, we give an algorithm whose policy regret is sublinear and has a (provably necessary) dependence on the time required to distinguish the optimal arm from the rest. We illustrate the behavior and performance of this algorithm via simulations. For the decreasing case, we present a simple greedy approach and show that the policy regret of this algorithm is constant and upper bounded by the number of arms.

3.1. Bidder Selection in Ad Exchanges

For online businesses advertising is a major source of monetization. Every day companies like Bing, Facebook, Google, and Yahoo run auctions — billions of auctions — to determine which advertising impressions to show. In particular, online display ad space is usually bought and sold through high-volume auction-based exchanges, of which AppNexus, Google's DoubleClick, and Microsoft Ad Exchange are examples. In these online display ad exchanges, impressions are continuously received from publishers and auctioned off among real-time bidders. Economic theory holds that such auctions allocate resources efficiently. The auction also determines the price paid by the winner. Of this payment, a fixed percentage goes to the exchange and the remainder is paid to the publisher. These transactions constitute the two revenue streams of online advertising, those of ad exchanges

and of publishers.

On the exchange side the process of running an auction usually consists of two steps: (1) Once a query arrives from the publisher side, the exchange calls a subset of buyers[2] to participate in the auction (the callout step). (2) Then, among the responding buyers the exchange runs an auction to determine the winner and price (the auction step). There are multiple reasons for the existence of the first step — the focus of this work. First, a significant percentage of bidders are limited by the number of calls per second they can respond to, their *quota* (Sel, 2015; Chakraborty et al., 2010; Devanur et al., 2011). The exchange must protect these bidders from receiving more calls than their servers can handle[3]. Furthermore, the exchange itself may need to limit the number of callouts sent to bidders to conserve its resources.

In practice, *random quota throttling (RQT)* is the principal solution by which these constraints are enforced. At a high level RQT decides which buyers to call randomly and with probabilities proportional to the quota-per-seconds (qps). Given that the exchange interacts with its bidders repeatedly and over time, it has access to *historical auction data* containing information about each bidder's segments of interest. By *learning* this information the exchange can *target* bidders more effectively. The combination of the learning and targeting is what we call a *"callout mechanism"*. An ideal callout mechanism reduces resources when bidders are unlikely to be interested, and increases calls when bidders are more likely to perceive value.

Finding the optimal callout set is equivalent to solving the following optimization problem: Subject to bidders' quota, which callouts should be sent to them for valuation and bidding so that the exchange's *long-term revenue* is maximized? Finding the revenue-maximizing callout is computationally hard (Section 3.1.1). As a consequence, the exchange has to

[2]While technically not the same, for simplicity in this work we use the terms "buyer" and "bidder" interchangeably.

[3]Beside such technological limitations, bidders may have financial constraints (see for example (Borgs et al., 2005; Dobzinski et al., 2005)) and/or specify volume limits to control exposure (see (Lahaie et al., 2008)).

rely on heuristics and approximation callout mechanisms. Different callout mechanisms can impact the long-term revenue differently, and the space of all possible callout mechanisms is extremely large. It is therefore not feasible for the exchange to evaluate every mechanism by running an actual experiment[4]. This necessitates the design of a framework that utilizes *historical data only* to compare the performance of various callout mechanisms. This is the main contribution of the present work. Rather than focusing on any particular mechanism, here we lay out a framework for evaluating any given callout mechanism in terms of its impact on long-term revenue.

The paper is organized as follows: We formalize the setting in Section 3.1.1. In Section 3.1.2 we start by observing that different callout mechanisms can impact the long-term revenue differently, mainly for the following two reasons: (1) Different mechanisms satisfy the quota differently. (2) Bidders are strategic and adapt their response to the choice of the callout mechanism. Measuring the former can be readily done using historical data, however, to measure the latter we need to have a model for the way bidders adapt their behavior to a new callout mechanism. We propose in Section 3.1.2 a game-theoretic model that captures the repeated interaction between the exchange and its bidders. This model motivates two performance metrics: *immediate revenue impact* and *social welfare*, both of which can be estimated from historical data (Section 3.1.2). To establish baselines for comparison, in Section 3.1.3 we consider two mechanisms: RQT, as well as a greedy algorithm (GRA) for which theoretical guarantees have been established, albeit under certain restrictive assumptions. In Section 3.1.3 we propose several natural callout heuristics. Finally in Section 3.1.4, we demonstrate our empirical framework, measuring the performance of these callout mechanisms on both real-world and synthetic auction data. We characterize the conditions under which each heuristic performs well and show that, in addition to being computationally faster, in practice our heuristics consistently and significantly outperform the baselines.

[4]Experiments can be unpredictably costly. In addition, there are usually restrictions in place on buyers the exchange can run experiments on.

Related Work

Microeconomics has a long line of research literature on auctions; see (Klemperer, 1999) for a survey. Repeated auctions, in which the auctioneer interacts with bidders multiple times, have received considerable attention (Bikhchandani, 1988; Ozkan and Wu; Thomas, 1996). In particular, the problem of *pricing* in repeated auctions has been studied extensively (Amin et al., 2013; Kleinberg and Leighton, 2003; Blum et al., 2004; Bar-Yossef et al., 2002; Cesa-Bianchi et al., 2013; Mohri and Medina, 2014). That said, most of the previous work has not considered strategic buyers. Some consider random bidders (Mohri and Medina, 2014), some study bidders who participate only in a single round (Kleinberg and Leighton, 2003; Blum et al., 2004; Bar-Yossef et al., 2002; Cesa-Bianchi et al., 2013), and some focus on bidders who participate in multiple rounds of the auction, but are interested in acquiring exactly one copy of a single good (Hajiaghayi et al., 2004). In none of these cases do bidders react to the seller's past behavior to gain higher payoffs in future rounds. However, in many real-world applications, the same set of buyers interacts repeatedly with the same seller. There is empirical evidence suggesting that these buyers behave strategically, adapting to their history to induce better payoffs (Edelman and Ostrovsky, 2007). Indeed, a growing literature enumerates the various strategies buyers can follow in order to improve their payoff (Cary et al., 2007; Kitts and Leblanc, 2004; Jofre-Bonet and Pesendorfer, 2000; Lucier, 2009; Gummadi et al., 2012).

More recently, several studies have focused on the strategic aspect of bidders' behavior in repeated auctions and, in particular, on the problem of setting near-optimal prices. (Acquisti and Varian, 2005; Kanoria and Nazerzadeh, 2014) study the impact of Intertemporal Price Discrimination (IPD), i.e. conditioning the price on bidder's past behavior, and examine the conditions under which IPD becomes profitable. Amin et al. (2013, 2014) investigate repeated posted-price auctions with strategic buyers, and present adaptive pricing algorithms for auctioneers. With these algorithms the auctioneer's regret increases as a function of bidders' discount factors.

In this work, rather than optimizing reserve prices, we focus on improving the callout routine. Of course, targeting bidders to call to an auction has already been reduced to practice. Consider *Selective callouts*, as implemented by Google's display ad exchange (Sel, 2015): Unlike RQT, Google's selective callout algorithm (SCA) identifies the types of impressions a bidder values, thereby increasing the number of impressions the bidder bids on and reducing the number of callouts the bidder ignores.

To our knowledge, (Chakraborty et al., 2010) is the first work to study the callout optimization problem from a purely theoretical perspective. The authors model callouts by an online recurrent Bayesian decision-making algorithm, one with bandwidth constraints and multiple performance guarantees. The optimization criteria considered in (Chakraborty et al., 2010) are different from ours — here we consider long-term revenue as the principal criterion and assert that it is the most natural choice. Also, unlike Chakraborty et al. (2010), we study the effect of strategic buyer behavior on system-wide, long-term outcomes. Relatedly, Dughmi et al. (2009) investigate the conditions under which the expected revenue of a one-shot auction is a submodular function of the set of participating bidders. While these conditions are restrictive and do not often hold in practice, we adopt Dughmi's greedy algorithm as a baseline and compare other algorithms to it.

Our work is indirectly related to the large and growing body of research on budget constraints in online auctions. Papers in this literature can be divided into two main categories: (a) those concerned with the design and analysis of auction mechanisms with desirable properties — truthfulness, optimal revenue, and so on — for budget constrained bidders (see for example (Borgs et al., 2005; Hafalir et al., 2012; Dobzinski et al., 2005; Balseiro et al., 2015)); and (b) those that present optimal or near-optimal bidding algorithms for such bidders (see for example (Chakrabarty et al., 2007; Archak et al., 2012)).

Incentive Issues: We close this section with a remark for readers familiar with the literature on incentive issues and truthfulness in mechanism design (see Chapter 23 in (Mas-Colell et al., 1995) for an overview of the main results of this topic.) While we are concerned with,

and present shortly a model for, the bidders' strategic reactions to the choice of the callout mechanism and its impact on long-term revenue, we do not claim that a bidder is better off by *bidding truthfully* when the system is in equilibrium. Indeed, in a setting as complicated as that with which we are dealing here — in which bidders have complex strategy spaces and information structures — the auction mechanism itself already fails to maintain incentive compatibility; see (Borgs et al., 2005; Gonen, 2008) for related hardness and impossibility results. Rather than setting the ambitious goal of restoring bidding truthfulness, here we consider a model in which both the action space and the information structure are simplified (see Section 3.1.2 for further details). In spite of the simplicity, the analysis of our model provides us with important insights about the performance of callout mechanisms.

3.1.1. Model and Preliminaries

Let $B = \{1, 2, \cdots, n\}$ denote the set of bidders active in the exchange. Each bidder $i \in B$ has a quota-per-second constraint denoted by $q_i > 0$. This quantity is known to the exchange and specifies the number of auctions bidder i can be called to per second. Consider a particular time interval of length one second, and assume that during this period the exchange receives a sequence of ad slots $A = \{a_t\}_{t=1}^{T}$ in an online fashion. Let v_i^t denote the value (or ROI) of ad slot a_t to bidder i. The exchange does not know the valuations in advance, but can learn about them through the bids. Let b_i^t specify bidder i's bid for the ad slot a_t.

At time $t = 1, 2, \cdots$ when ad slot a_t arrives, the exchange must choose a subset $B_t \subseteq B$ to call for participation in the auction for a_t while respecting all the quota-induced constraints. A *callout mechanism/algorithm* is any logic used to decide which subset of bidders to call at each time step t using the history of bidding behavior observed up to time t. More precisely, let the matrix $\mathbf{H}_t$ denote the bidding history observed by the exchange up to time t. Given $\mathbf{H}_{t-1}$ as the input a callout mechanism selects B_t for every $t = 1, 2, \cdots$. Once called, each bidder $i \in B_t$ decides whether to participate in the auction. The auction mechanism then specifies the winner (if any) and price. The recent bids along with $\mathbf{H}_{t-1}$ are then used to

set $\mathbf{H}_t$. Figure 8 illustrates the flow of a typical callout mechanism.

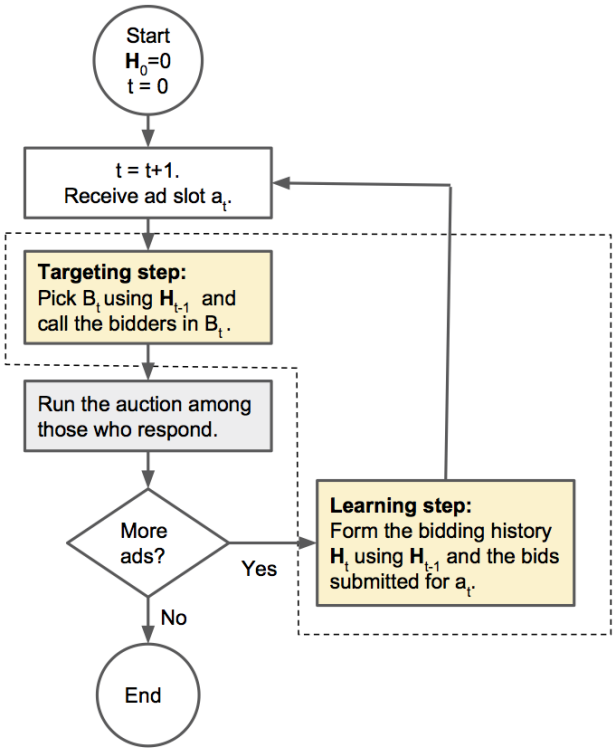


Figure 8: The high-level flowchart of the system. The two main building blocks of a callout mechanism are the components for *learning* and *targeting*. We treat the auction mechanism as fixed and given throughout.

Throughout, we treat the auction step as given and fixed; we assert no control over how the auction is run or how its parameters, e.g. the reserve price, are tuned. While much of our work extends readily to other auctions, unless otherwise specified we assume the auction mechanism is second price with reserve. Let r_t be the (given) reserve price for auction t. Bidders do not observe r_t before submitting their bids.

For the exchange, the ultimate goal is to maximize the *long-term revenue*. Therefore we define the performance of a callout mechanism by its impact on long-term revenue. Long-term revenue is simply the (possibly discounted) sum of the revenue earned at each time step $t = 1, 2, \cdots$. We denote the discount factor by δ ($0 \leq \delta \leq 1$). The optimal callout mechanism is the one that maximizes the long-term revenue. We conclude this section by demonstrating that identifying the optimal callout is computationally hard. And this

remains true even if we assume bids are sampled from fixed distributions, the exchange knows these distributions ahead of time, and can accurately predict the sequence of ad slots it receives.

Proposition 2 *Suppose when called to an auction for item t, bidder i samples their bid, b_i^t, from a distribution D_i^t ($i = 1, \cdots, n$ and $t = 1, \cdots, T$). Let variable $x_{i,t}$ be the indicator of whether bidder i is called to the auction for a_t. Solving the following optimization problem is NP-hard:*

$$\begin{aligned} \max \quad & \sum_{t=1}^{T} \delta^t \mathbb{E} R(x_{1,t}, \cdots, x_{n,t}) \\ s.t. \quad & \forall i : \sum_{t=1}^{T} x_{i,t} \leq q_i \\ & \forall i, t : x_{i,t} \in \{0, 1\} \end{aligned}$$

where the random variable $R(x_{1,t}, \cdots, x_{n,t})$ denotes the revenue of auction t with participants $\{i | x_{i,t} = 1\}$, and the expectation is taken with respect to D_i^t's.

We establish this by showing that the above class of problems includes a known NP-hard problem—maximum 3-dimensional matching—as a special case.

Proposition 2 motivates our approach in this work. Consider the hypothetical world in which we have access to a perfectly accurate *bid forecasting* model. In this setting one natural proposal is to use this model to forecast the bids, then call the two bidders with the highest bids in every auction. We emphasize that in practice this is simply not possible: Even the best forecasting model can only forecast a *bidding distribution*, not a particular number. Furthermore, even if we assume this distribution is fully compliant with the true bidding behavior, according to Propostion 2 finding the optimal callout set is computationally hard. This implies that the exchange has to rely on heuristics and approximation algorithms to construct the callout set. In this work we present a data-driven approach for evaluating and comparing various callout algorithms in terms of their average impact on long-term

revenue.

3.1.2. Measuring Performance

We start by observingthat different callout mechanisms result in different levels of long-term revenue for the following two reasons: (1) Different mechanisms satisfy quota differently. (2) Bidders are strategic and adapt their response to the choice of the callout mechanism. Measuring the first type of impact is readily possible using historical data — one simply needs to run each callout mechanism on the data and observe the change in revenue. By selling previously unfilled impressions, a smart callout mechanism can increase the exchange's revenue, and at the same time maintain, and perhaps improve, bidders' utilities.

In order to improve long-term revenue, however, it does not suffice for the exchange to find a callout mechanism with high revenue performance on historical data. In real world bidders are strategic and adapt their response to the choice of the callout mechanism. For instance, it is possible that a callout mechanism does not result in selling previously unfilled part of the inventory. Rather, it merely increases the competition and drives the prices up. While this does not hurt the revenue immediately, it can reduce the utility that certain bidders earn. Bidders potentially react to this change in their payoffs in future rounds. To quantify this type of impact on long-term revenue we need to have a *model* for bidders' reactions to the choice of the callout mechanism.

A Two-Stage Game Formulation

There are usually many options (i.e exchanges) available to bidders to choose from (i.e. participate in). We make the following simple, natural assumption about the bidder's reaction to the choice of the callout mechanism in one particular exchange: Bidder seek to maximize their *utility* (ROI), that is, they always choose to participate in the exchange that provides them with the highest utility (value minus cost). In what follows, we make two simplifying assumptions: First we limit the bidder's action space to the choice of which exchange to *participate* in. Second, we assume by participating in an exchange, all that

the bidder can observe is their utility from that exchange. We acknowledge that compared to real-world, these assumptions drastically simplify the action space and the information structure for both the bidders and the exchange. Nonetheless as the following analysis shows, this simplified model suffices to capture some of the most important aspects of the interaction between the exchange and bidders.

We follow the standard modeling approach in microeconomic theory for repeated interactions among strategic agents (see (Mailath and Samuelson, 2006)). Consider the following two-stage game between two players, the exchange and one bidder. The action space for the exchange consists of all possible callout mechanisms, denoted by the set E. The action space for the bidder consists of two actions: participating or taking the outside option. At the beginning of the first stage the exchange commits to a callout mechanism $e \in E$. (Note that our analysis does not rely on a particular choice of e.) Next, the bidder, without the knowledge of e, decides whether to participate in the exchange in the first round. If they do, then their expected utility is equal to

$$u^e = v^e - c^e$$

where v^e is the bidder's average valuation for the items they win and c^e is the average cost they pay for those items under callout mechanism e. If the bidder chooses to participate, the expected revenue that the exchange receives equals c^e. If they do not participate in the auction, they take their outside option, which provides them with utility u and the exchange with an incremental revenue of 0.

In the second stage of the game, which captures the future interaction of the bidder with the exchange, the bidder again chooses whether to participate or not. This time, however, the bidder is aware of the utility they earns from participation — if they chose to participate in the first stage. Note that this is in line with the anecdotal evidence suggesting that bidders do in practice run experiments to estimate the utility they can earn from different exchanges and adjust their rates of participation accordingly. Utilities for this stage are

defined similarly to the first stage. Denote by δ the players' discount factor for the future (i.e. the second stage) payoffs.

Proposition 3 *Among all callout mechanisms $e \in E$ for which $u^e \geq u$, let e^* be the one for which c^e is maximized. If $\frac{\max_e c^e}{c^{e^*}} - 1 \leq \delta$, the following strategy profile is the unique sub-game perfect equilibrium of the above game: The exchange chooses e^*, and the bidder chooses to participate in each stage if and only if according to their beliefs at that stage $\mathbb{E}u^e \geq u$.*

The above proposition suggests that when players value their future utilities — or more precisely, when the discount factor δ is large enough — the ideal callout mechanism increases the immediate revenue as much as possible (i.e. maximizes c^e) while providing the bidder with a utility level at least as large as what they can earn from their outside option (i.e. maintaining the constraint $u^e \geq u$). In other words, when choosing a callout mechanism, the exchange faces a trade-off between immediate- and long-term revenue: Of course, for a given callout mechanism, increasing its callouts can boost the revenue. However, unless the exchange induces sufficiently high value for bidders, such increases in callouts ultimately discourage their participation in future rounds — they find their outside option more profitable. This in turn translates into less revenue for our exchange in the long run.

Performance Metrics and Estimators

The argument above leads us to two metrics for evaluating the performance of a callout mechanism: immediate revenue impact and social welfare. Next we propose ways for estimating these metrics, c^e, u^e, as well as the outside option utility u, from historical bidding data. Throughout we assume we know and can therefore simulate the auction mechanism on any given set of bids.

More precisely, suppose we have access to the bidding data over a period of length S for n bidders (i.e. b_i^t for all $t = 1, \cdots, S$ and $i = 1, \cdots, n$). We will need the following notation: Let $\tilde{c}_i$ be the average cost bidder i pays for the items they win over time period

$t = 1, \cdots, S$—before a new callout mechanism is implemented. Similarly, let $\tilde{b}_i$ be the average bid submitted by bidder i for the items they win over this time period. Let $\tilde{c}_i$ be the average cost bidder i pays for the items they win over time period $t = 1, \cdots, S$—under callout mechanism e. Note that this can be easily computed by simulating the auction mechanism on the historical bidding data and the new callout set. Similarly, let $\tilde{b}_i$ be the average bid submitted by bidder i for the item they wins over this time period—under callout mechanism e.

Immediate revenue impact The following is an unbiased estimator of c^e: $\bar{c}^e = \frac{1}{n}\sum_{i=1}^{n} \tilde{c}_i^e$.

Social welfare To estimate the social welfare, we need some proxy of the bidders' valuations. Since we do not have access to actual valuations, for practical reasons we are constrained to rely on bids as a proxy for value. In our setting, the assumption of bid as a proxy for valuation is relatively benign: any bias in measuring the utility of winning auctions in one exchange is likely the same bias for winning auctions in any other exchange. Further, the choice of bid-as-value enables bid-minus-cost as the residual value for the buyer, one that is visible both to each buyer and to the exchange. In that sense, bid-minus-cost represents the good-faith estimate of the residual value, one that the exchange can actively work to preserve over the set of buyers. Assuming that a bidder's average bid reflects their true valuation, the following is an unbiased estimator for u^e: $\bar{u}^e = \frac{1}{n}\sum_{i=1}^{n}(\tilde{b}_i^e - \tilde{c}_i^e)$.

Outside option utility The following is an estimator for u: $\bar{u}^e = \frac{1}{n}\sum_{i=1}^{n}(\tilde{b}_i - \tilde{c}_i)$. We argue that the above is an unbiased estimator assuming that bidders are rational and participate at positive rates in both the exchange and their outside option. Here is why: Consider a strategic bidder who participate at positive rates in both our exchange and the outside option. This means that both of these options provide them with equal utilities on average, otherwise, they would have been better of by only participating in the higher paying exchange.

3.1.3. Proposed Callout Algorithms

Baselines

We propose two baselines, RQT and GRA, against which we compare other callout mechanisms[5].

Random Quota Throttling (RQT) As a naive comparison baseline, we consider random quota throttling, RQT: we drop each bidder with a fixed probability p.

The Greedy Algorithm (GRA) In settings where the auction revenue is monotone and submodular[6] a simple greedy algorithm, which greedily adds bidders by their marginal impact on revenue, is guaranteed to obtain revenue at least as large as $(1 - 1/e)$ of the (one-shot) optimal solution (Dughmi et al., 2009).

Algorithm 2 describes our baseline, GRA.

ALGORITHM 2: The Greedy Algorithm (GRA)

Data: $K \in \mathbb{N}$, $\epsilon > 0$
Start with $t = 0$;
Let $\hat{D}_i$ be the estimated bidding distribution for bidder i. Start with equal estimations for all bidders;
while *there exist more ad slots* **do**
    $t = t + 1$;
    Receive ad slot a_t;
    $B_t = \emptyset$;
    for $i = 1, 2, \cdots, K$ **do**
        Approximate the marginal revenue impact of adding bidder i to B_t using $\frac{1}{\epsilon}$ repetitions;
        Add that bidder with highest marginal revenue impact to B_t;
    end
    Run the auction among bidders in B_t;
    Update $\hat{D}_i$ for all i using $\mathbf{b}_t$;
end

[5]Note that SCA (Sel, 2015) is not among our baselines, because the details of the algorithm have not been published.

[6]This is for example the case for revenue maximizing auction mechanisms in matroid markets with bidders whose bids are drawn independently (see (Dughmi et al., 2009) for the details).

Our Heuristics

Throughout this section we focus on callout mechanisms with two natural properties, *symmetricity* and *myopicity*. We call a callout heuristic *symmetric* if two bidders with exactly the same bidding history up to time t have the same chance of being called to the auction of a_t. This basic property must be satisfied to ensure that the mechanism is fair. A *myopic* callout heuristic disregards the effect of today's targeting on the future bidding behavior. The targeting in all of our algorithm is done via *thresholding*: the algorithm calculates a score vector $\mathbf{s}_t$ from the (relevant part of the) history vector $\mathbf{H}_t$ for all bidders; by designating a suitable thresholds $\boldsymbol{\theta}$, it selects which bidders to call out. See Algorithm 3 for details. Next we present several natural choices for the metric $m(.)$ and specify how to update the scores with it.

ALGORITHM 3: A Myopic Symmetric Thresholding Algorithm

Data: metric $\mathbf{m}(.)$ and threshold θ
Start with $t = 0$ and equal scores for all bidders ($\mathbf{s}_0 = \mathbf{0}$);
while *there exist more ad slots* **do**
  $t = t + 1$;
  Receive ad slot a_t;
  Set $B_t = i : s^i_{t-1} \geq \theta$;
  Run the auction for a_t among bidders in B_t;
  Update $\mathbf{m}_t$ by including the bids for a_t;
  Update $\mathbf{s}_t$ using $\mathbf{s}_{t-1}$ and $\mathbf{m}_t$;
end

We end this section with a remark: we restrict ourselves to thresholding mechanisms—as opposed to the broader class of *randomized* bidder targeting—because the expected revenue earned from any randomized targeting can be written as the weighted sum of revenue earned by different $\boldsymbol{\theta}$ vectors. This sum is maximized when all the weight is put on a single (best) threshold vector $\boldsymbol{\theta}$.

A Linear Heuristic

Consider an auction with n bidders and let b_i denote the bid of bidder i. Let $\mathbf{b} = (b_{(1)}, b_{(2)}, \cdots, b_{(n)})$ denote the ordered bid vector where the subscript (j) denotes the j-th order statistic: $b_{(1)} \geq b_{(2)} \geq \cdots \geq b_{(n)}$. For any $S \subseteq B$, let $R(S)$ denote the revenue that the exchange earns if it calls all the bidders in S to the auction. We want to attribute to each bidder $i \in S$ part of the total revenue, which we denote by R_i, such that the attribution satisfies the following properties for any set S:[7]

1. *Symmetry:* bidders with equal bids are attributed the same revenue: $i, j \in S$ and $b_i = b_j \Rightarrow R_i = R_j$.

2. *Linearity:* $\mathbf{R}_S = \mathbf{A}\mathbf{b}_S$ for some fixed matrix $\mathbf{A}$. Throughout, for any vector $\mathbf{x}$ and any set S, $\mathbf{x}_S$ is a vector that is equal to 0 on any component $i \notin S$ and equal to $\mathbf{x}$ everywhere else.

3. *Conservation of Revenue:* the sum of attributions equals the total revenue: $\sum_{i \in S} R_i = R(S)$.

Proposition 4 *For a second-price auction properties 1–3 uniquely identify* $\mathbf{A}$.

Shapley's Linear Heuristic (ShA) works by computing and thresholding on the average value of the above metric for each bidder. We next argue that the above heuristic estimates the expected counterfactual revenue impact of adding a new bidder to an auction with respect to a particular distribution. Let S denote the subset of bidders called to the auction, but without bidder i. Consider two almost-duplicate worlds, one with bidders $S \cup i$ called to the auctions (the observable) and the other without the bidder in question, S (not observable, i.e. counterfactual). If everybody participates, the impact on revenue of the intervention — including i — is $(R(S \cup i) - R(S))$. However, the set of bidders who actually end up participating in the auction is a random variable. Suppose the probability of the subset

[7]The axioms introduced here bear similarity to those leading to Shapley values in cooperative game theory (Shapley, 1952; Neyman, 1989), hence the naming.

$T \subseteq S$ of bidders ending up participating is $\Pr(T)$. Then we can write the expected revenue impact of adding i to S as follows:

$$R(S \cup i) - R(S) = \sum_{T \subseteq S} \Pr(T) \left(R(T \cup i) - R(T) \right)$$

It is easy to see that the above is exactly equivalent to the calculation in Proposition 4 if $\Pr(T) = \frac{|T|!(|S|-|T|-1)!}{|S|!}$. The above distribution is uniquely imposed due to the symmetry property[8].

Non-linear Heuristics

We now turn to propose and discuss the following (nonlinear) heuristics.

History of bidding above reserve (BAR) We call bidder i to an auction if the number of times she has in the past bid above the reserve price for similar items exceeds some threshold θ_i. Obviously, as we increase θ_i, the number of bidders called to the auction decreases. As we will see in Section 3.1.4, the performance of this heuristic depends heavily on the reserve price setting algorithm. The more accurate this algorithm is — in predicting the auction winner, in predicting the winner's bid — the better this heuristic performs. In the ideal case where the pricing algorithm can predict exactly the winner's bid, the BAR heuristic maximizes the revenue: we only need to call to the auction the person who is willing to pay the most and set the reserve price to a level just below her bid. Conversely, consider the extreme case when the reserve price is 0 and therefore contains no information about bidder interest: the bid-above-reserve metric is equal for all bidders, and the heuristic BAR therefore performs poorly.

History of winning (WIN) We call bidder i to an auction if the average number of times she has won in the past for similar items exceeds some threshold θ_i. This algorithm

[8] As a future generalization, one can discard this property and consider a more general linear approach in which $A^{(P)} \times \vec{e}_k = \vec{p}_k$, where $\vec{p}_k = (p_{k1}, .., p_{kk}, 0, .., 0)$, $\sum_{l=1}^{k} p_{kl} = 1$, and $p_{kl} \geq 0$.

performs well in the absence of an accurate reserve price setting algorithm for the following reason: The number of times a bidder wins in a segment indicates how interested she is in similar impressions. So by calling these interested bidders, competition (and, therefore, the second price) increases. The problem with this approach is that when multiple bidders have equal interest in an impression segment. Instead of splitting the impressions among them, this heuristic calls them simultaneously, driving competition up and dissipating bidders' resources. To the extent that the price setting algorithm is accurate, WIN is wasteful. In addition to the above drawback, a bidder may not win often in a segment, but still succeed in setting a high second price for the winner. The WIN heuristic ignores this effect and does not call such price-pressuring bidders to the auction.

Total spend (SPD) We call bidder i to an auction if her total spend for similar items so far exceeds some threshold θ_i. This heuristic can be thought of as an extension of WIN, one weighted not only by how many times a bidder wins in a segment, but also by how much she spends upon winning.

Average ranking (RNK) We call bidder i to an auction if her average rank in past auctions for similar items lies below some threshold θ_i. This heuristic can be thought of as a generalized and smoothed version of WIN. With this heuristic the winner (i.e. the first ranked bidder) is not the only one who receives credit. Rather, every bidder increases her score proportional to the placement of where her bids stand relative to others.

Total bid (BID) We call bidder i to an auction if her total past bids for similar items exceeds some threshold θ_i. The problem with this heuristic is the following: Consider a bidder who bids low most of the time, but every once in a while submits an unreasonably high bid to raise their average. This heuristic cannot distinguish this bidder from one that consistently submits reasonably high bids.

Total attributed revenue (RVC) We call bidder i to an auction if her total attributed revenue for similar items exceeds some threshold θ_i. Note that a bidder's revenue impact

manifests not only when she directly provides the winning bid, but also indirectly when she influences the price of any other winners. The problem with this heuristic is that it completely disregards the role bidders other than the first and second-highest could have played in the auction. When the number of repetitions is not high, we expect ShA to outperform this heuristic. As the number of repetitions increase, this heuristic converges to ShA.

Shortly in Section 3.1.4 we see not only that our heuristics are faster, but also that they outperform both baselines. One heuristic that deserves particular attention is ShA. ShA does not suffer from the problems pointed out for non-linear heuristics above; we therefore expect it to outperform them in practice. In Section 3.1.4 we see that this is indeed the case.

3.1.4. Simulations

In this section we demonstrate our framework on both synthetic and real-world auction data. By simulating each mechanism on such data, we estimate its immediate revenue impact and social welfare impact using the estimators proposed in Section 3.1.2, and compare them with the baselines in Section 3.1.3. As predicted earlier, we see that most of our heuristics consistently outperform the baselines.

At a high level, our simulator receives the auction data and processes it one auction at a time using the heuristic specified. For any given item, the simulator decides which subset of bidders to call to the auction for that item (by setting the threshold value), simulates the auction mechanism among those bidders, and finally calculates the revenue and social welfare of the auction. By changing the threshold value θ, the percentage of called-out bidders varies. That allows us to obtain a range of values for the performance metrics for each heuristic as a function of the percentage of bidders called out.

In our simulations, we assume the qps rates are constant across bidders ($q_i = c$ for all i and some constant c). This not only simplifies the simulation, but also, by enlarging the number

of potential buyers in each auction, it implicitly increases the scale of each simulation. In practice, when different bidders have different qps's, one can designate different threshold values for each one of them; these thresholds can be set to guarantee no bidder is called out to more than their qps. More importantly, the above choice allows us to see how each mechanism's performance evolves as the percent of bidders it can keep in the auction increases (i.e as we vary p). Based on the argument in Section 3.1.2, a callout mechanism outperforms the baselines if: (1) By calling the same percentage p of bidders to auctions, it results in revenue higher than both RQT and GRA. (2) By calling the same percentage p of bidders to auctions, it results in social welfare at least as large as RQT. For example, in Figure 9 the hypothetical callout mechanism outperforms both baselines.

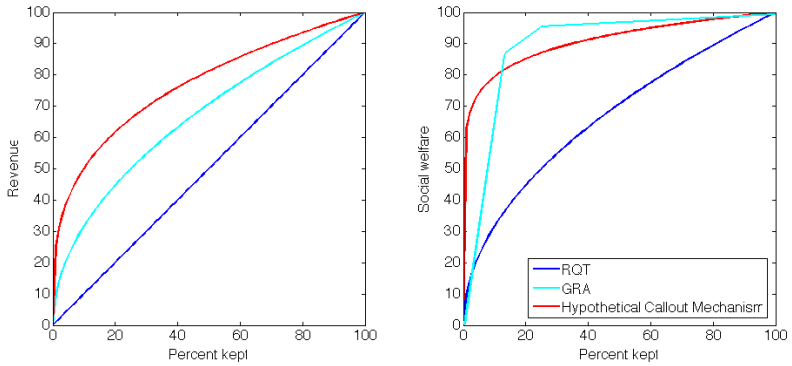


Figure 9: A good callout mechanism (red) must exceed the baselines RQT (blue) and GRA (cyan) in terms of revenue (left panel) while maintaining social welfare at least as large as RQT (right panel).

Datasets

Synthetic auction data In this dataset each bidder's bid is sampled from a fixed distribution specific to that bidder. We generate the data as follows: We assume there is a total of T items that arrive at time steps $1, 2, \ldots, T$. We have n bidders and each one of them samples their bid from a *log-normal* distribution with a fixed but bidder-specific median and variance. Note that the assumption that bids follow a log-normal distribution is standard in the literature and is backed by multiple empirical studies. See for instance (Wilson, 1998; Xiao et al., 2009; Ostrovsky and Schwarz, 2011). For each bidder the mean bid and

variance is sampled from the *uniform* distribution with support on $[0, \mu]$ and $[0, \sigma]$, respectively. For simplicity, we assume the reserve price is *fixed* and equal to r across all auctions. We generate M datasets with this specifications. By repeating out calculations on these M datasets, we obtain confidence intervals for our empirical results. Throughout we set $n = 100, \mu = 1, \epsilon = 0.05, M = 10$.

Real auction data This dataset consists of the bids observed on Google's DoubleClick Ad Exchange for a random set of 100 buyers, submitted over three consecutive weekdays for a set of similar auctions. We ensure that the items are similar by restricting the queries to a small industrialized upper-middle-income country and one type of device. For ease of interpretation, we scale the observed bids so they are in units of reserve price, i.e. on the same scale as the simulated auction data above with $r = 1$. For each bidder we generate the missing/unobserved bids by resampling from the empirical distribution of her observed bids.

Findings

Figures 10, 11, 12 illustrate the performance of our callout mechanisms along with the baselines on the synthetic dataset for different setting of the parameters. The first row in each figure depicts the revenue and the second row depicts the social welfare versus the percentage of bidders called out to each auction. The third row depicts the average revenue earned by each heuristic across all threshold values. This can be thought of as an overall score for each algorithm. Figure 13 illustrates the performance of our callout mechanisms on the real dataset.

Figure 10 illustrates the effect of the reserve price r on the performance of each callout mechanism when the average variance σ across bidders is 1 and the number of items to be auctioned off, T, is 100. We observe that regardless of the reserve price, the relative rank of the heuristics in terms of the revenue integral, remains unchanged. For example ShA always outperforms the other algorithms by an statistically significant margin. Also see

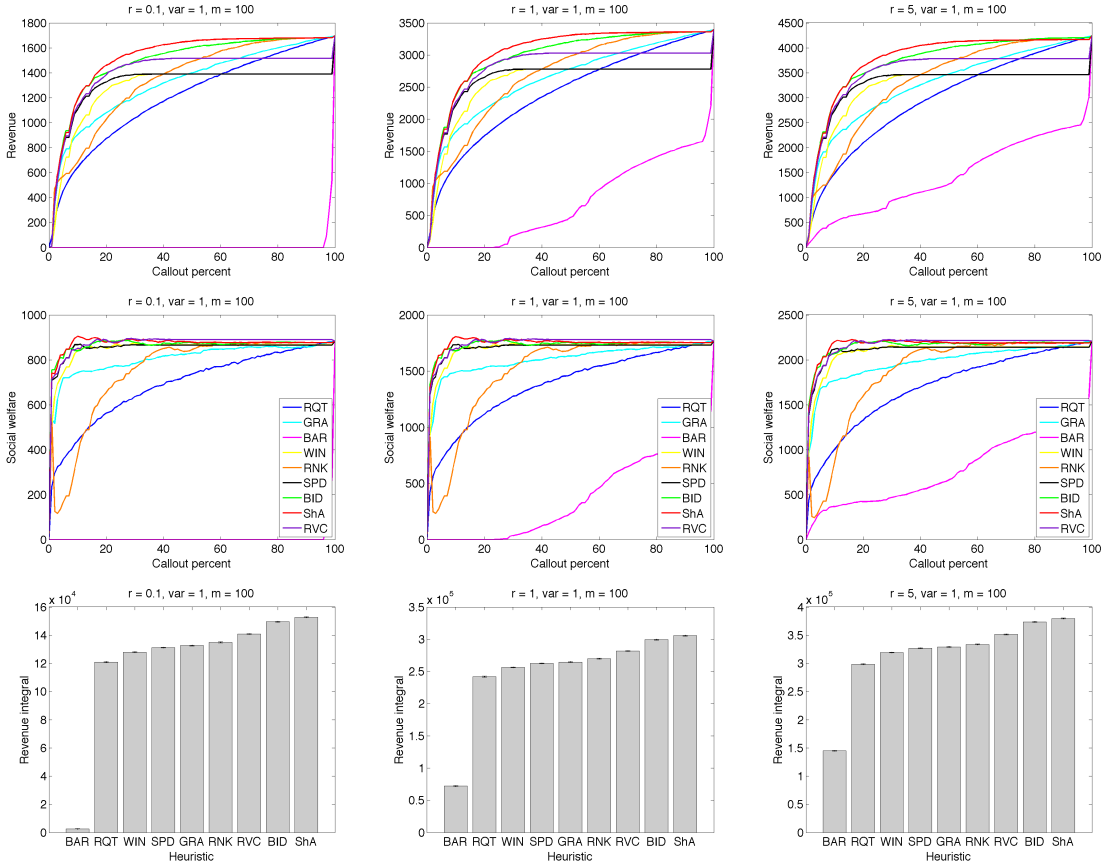


Figure 10: The effect of reserve price on the performance of each callout mechanism. Here $T = 100$ and $\sigma = 1$.

Figure 13 for a similar trend. As we expect, the performance of BAR improves as r grows larger. Note that when the percentage of bidders called to each auction is high, WIN, SPD and RVC fail to beat the RQT baseline. The reason is that these metrics give high scores to only a small subset of bidders and assign the remainder of bidders scores that are about equal. This induces the flat interval in the corresponding curves. In terms of social welfare, our heuristics always outperform RQT, except for BAR and RNK. RNK does not maintain sufficient social welfare when the percentage of bidders called to the auction is low. The reason is obvious: RNK calls bidders with lowest (best) ranks, making the winner pay more and degrading the social welfare as the result.

Figure 11 illustrates the effect of average bidding variance σ on the performance of each callout mechanism when the reserve price r is fixed and equal to 1 and the number of items

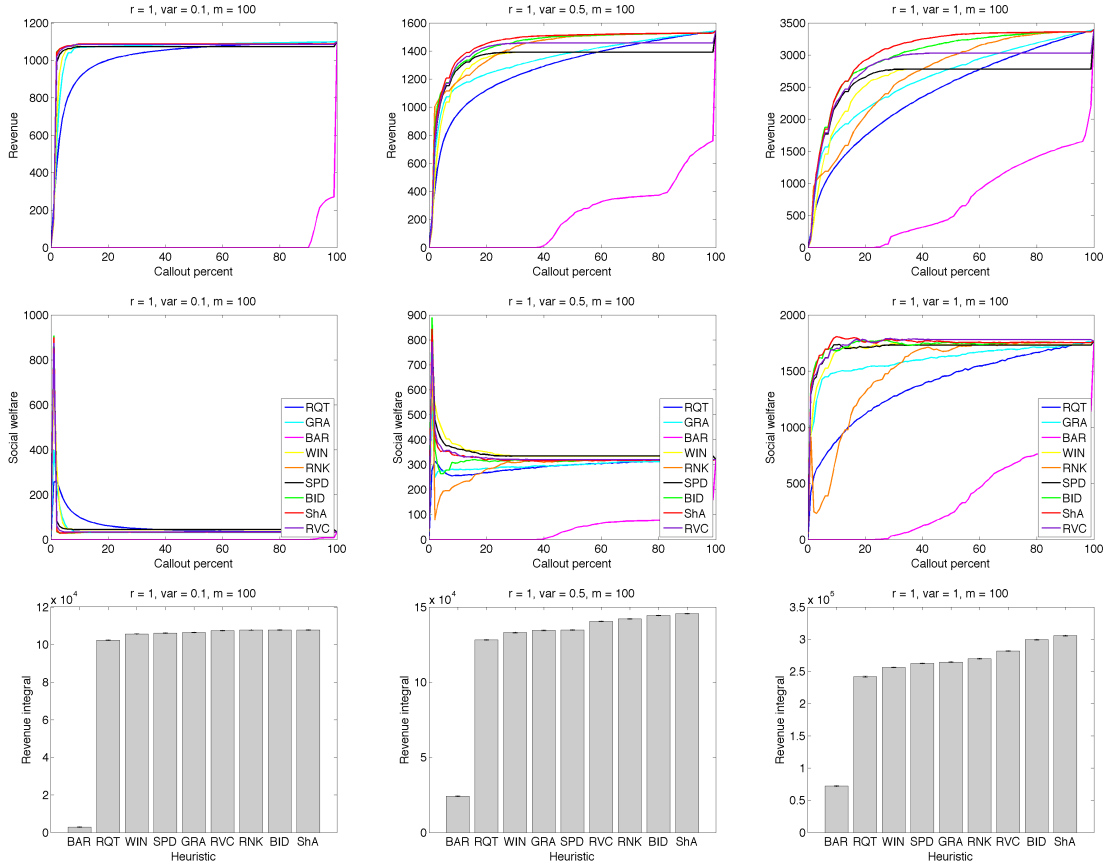


Figure 11: The effect of variance on the performance of each callout mechanism. Here $T = 100$ and $r = 1$.

T is 100. As variance increases, we see a divergence in the performance of various heuristics. In particular, the performances of GRA, BAR, RNK, and BID all start to deteriorate. Also, when the percentage of called bidders is small, we observe a sudden jump in social welfare. The heuristics exhibiting this phenomenon fail to maintain the auction pressure, dropping the winner's close competitors. As the result, the winner pays less, which boosts the social welfare.

Figure 12 illustrates the effect of the number of items T on the performance of each callout mechanism when the reserve price r is fixed and equal to 1 and the average variance σ across bidders is 1. We see that as T increases, the difference between the performance of different heuristics begins to vanish: the performance of all algorithms (expect RQT) converge to that of ShA, even WIN.

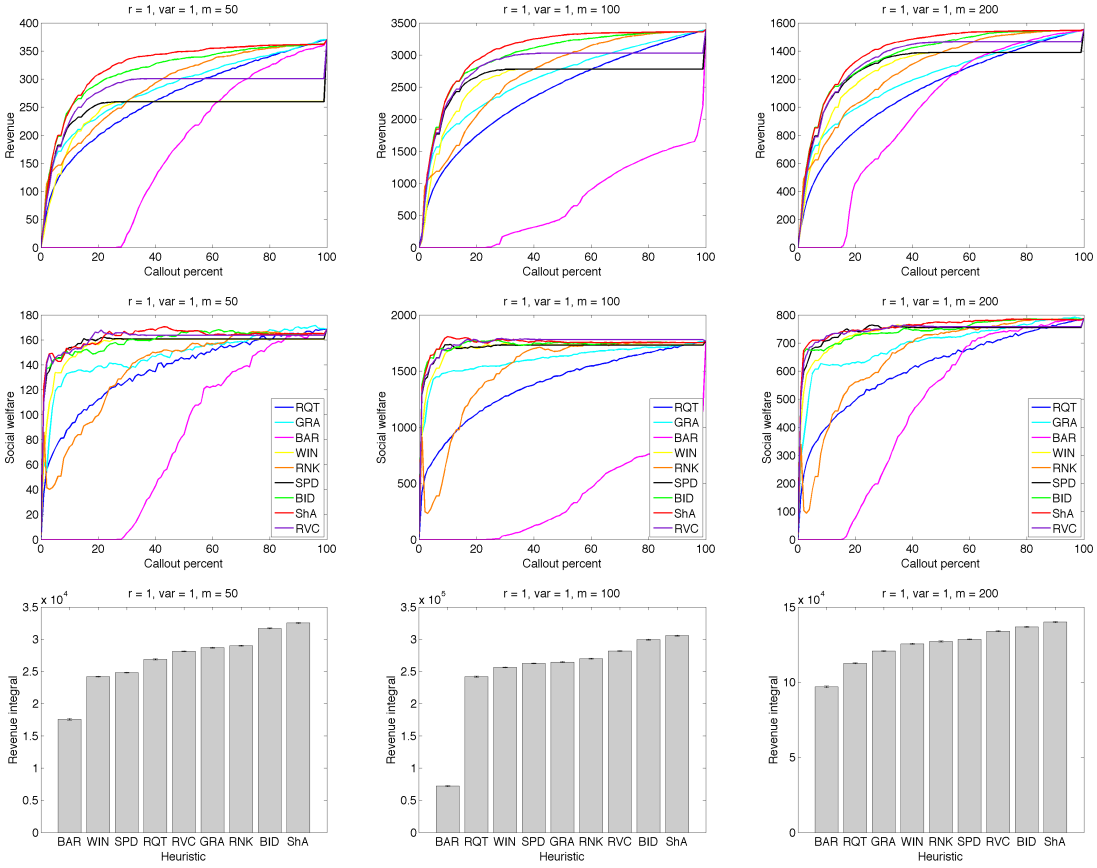


Figure 12: The effect of T on the performance of each callout mechanism. Here $r = 1$ and $\sigma = 1$.

The main takeaway messages from the empirical results presented above are:

- *It is easy to beat the RQT baseline.* Even our crudest heuristics, WIN and SPD, outperform RQT most of the time.

- *Some of our heuristics outperform both baselines.* More sophisticated heuristics, e.g. RVC, RNK, BID, and ShA, consistently outperform the baselines.

- *A good callout mechanism can significantly improve revenue.* For example in certain settings, ShA results in revenue up to 50% more than that of RQT and 25% more than that of GRA.

- *ShA* > *BID*> { *RNK, RVC, GRA*} > {*SPD, WIN, RQT, BAR* }. And these results are statistically significant across the settings investigated here.

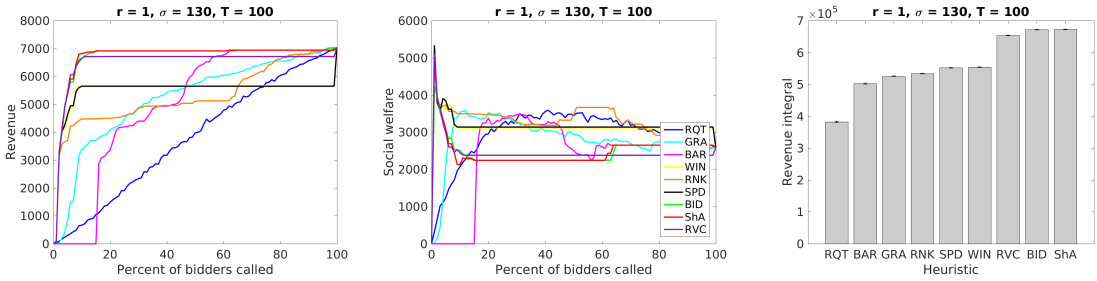


Figure 13: The performance of each callout mechanism on the real auction dataset. Here $n = 100$, $\sigma = 130$, and $r = 1$.

- *The more information a heuristic contains about the revenue impact of bidders, the better it performs.* We believe this is why ShA outperforms all the other heuristics: As we noted earlier, ShA estimates the counterfactual revenue impact of each bidder, and as a result improves revenue the most. Overall better heuristics call out more specifically to bidders with greater revenue impact.

3.1.5. Discussion and Future Directions

We presented a framework for evaluating the performance of callout mechanisms in repeated auctions using historical data only. Our framework is general enough to enable the study of other heuristics in settings beyond those considered here (e.g. alternative auction mechanisms, bidding distributions, etc.). In future, we intend to investigate the performance of more complicated callout mechanisms, including ones with more sophisticated learning steps; ones that combine multiple heuristics in a single score; ones that target bidders by means not easily represented by single-metric thresholding; and mechanisms that use online dynamic (as opposed to myopic) targeting.

3.2. Improving Workers in Online Labor Markets

In the well-studied multi-armed bandit setting, a decision maker is faced with the problem of choosing which arm to play over a sequence of trials. Each time the decision maker pulls an arm, he receives a reward, and his goal is to minimize some notion of *regret*. The majority of previous studies consider the so-called *external regret* as the objective. External regret is the difference between the total reward collected by an online algorithm

and the maximum reward that could have been collected by pulling a single arm *on the same sequence of rewards as the one generated by the algorithm.* Past solutions to this problem can be divided into two main categories: first, solutions that rely on statistical assumptions about the underlying reward process (see for instance(Robbins, 1985; Gittins et al., 2011; Auer et al., 2002b)); and second, solutions for the setting where an adversary, who is capable of reacting to the choices of the decision maker, determines the sequence of rewards (see e.g. (Auer et al., 2002b)).

Our goal in this work is to minimize a stronger notion of regret, namely *policy regret*, in a setting where the reward from each arm changes monotonically[9] every time the decision maker pulls that arm. More precisely, in the model we propose, there exist n arms, each with a *reward curve* that is unknown to the decision maker. Every time the decision maker pulls an arm, the corresponding reward of that arm monotonically changes according to its underlying reward curve.

Unlike traditional statistical approaches, we do not make probabilistic assumptions on the behavior of the arms; and unlike traditional adversarial approaches, we do not assume a fixed sequence of payoffs against which we measure our regret. In particular, under our assumptions the algorithm itself is actually *generating* the sequence of payoffs via its decisions. The right notion of regret is thus not comparing to the best single arm in hindsight on the sequence generated (external regret), but to the *best sequence that could have been generated* — that is, to the optimal policy that knows the reward functions. This stronger notion is what is called *policy regret* in the literature (Arora et al., 2012). The following simple example further illustrates the difference between policy and external regret.

Example 1 *Consider a setting in which there are two arms and time horizon $T \gg 10$. Arm 1 returns a reward of $\frac{i}{T}$ when pulled for the ith time, and arm 2 always returns a reward of 0.1. Consider the algorithm that always pulls arm 2. The external regret of this algorithm is zero because at every time step, it pulls the arm that would give it the largest*

[9]See (Slivkins, 2011) for other motivating examples and a different formalization of monotone bandits.

reward on that time step. But the policy regret of this algorithm grows linearly with T, as the best policy in hindsight indeed pulls arm 1 at every time step. [10]

We are motivated by settings in which the available actions correspond to the assignment of a sequence of relatively homogeneous tasks to one of a finite set of workers, as is often the case in crowdsourcing systems (see for example (Tran-Thanh et al., 2014; Heidari and Kearns, 2013)). In such settings, it is reasonable to expect that workers' performance may improve or decay with repeated trials, depending on the nature of the task. For example, tasks that are unfamiliar and challenging (such as segmenting brain images into individual neurons (Seung, 2015)) may require a training period during which performance gradually improves. In contrast, in a task primarily requiring only human perception (such as the transcription of license plates in images (Barowy et al., 2012; Du et al., 2013)), subjects may immediately be able to perform the task at a high level, but its tedious and taxing nature may lead to performance decay with increased workload. In both cases, different subjects may have different rates of improvement or decay.

The rest of this paper is organized as follows: In Section 3.2.1, we introduce the model. In Section 3.2.2, we study the case of increasing and concave reward functions and present an optimal online algorithm whose policy regret depends on a parameter we call τ. This parameter quantifies the time required to distinguish the optimal arm from the others, and we prove the dependence is necessary. We prove that the policy regret of this algorithm is sublinear and further illustrate the range of behaviors of τ and the performance of the algorithm via simulations. Finally, in Section 3.2.3 we investigate the case of decreasing rewards and present a provably optimal algorithm whose policy regret is constant and upper bounded by the number of arms.

[10] It is easy to adapt this example to show that specific algorithms with no external regret, such as EXP3, will indeed fail to have small policy regret.

Related Work

As discussed in (Arora et al., 2012), the notion of external regret fails to capture the actual regret of an online algorithm compared to the optimal sequence of actions it *could* have taken when the adversary is adaptive. Policy regret is defined to address this counterfactual setting. The authors show in (Arora et al., 2012) that if the adaptive adversary has *bounded memory*, then a variant of traditional online learning algorithms can still guarantee no policy regret. In our work, the rewards depend on *all* the actions taken so far and as the result our model is not captured by the bounded memory setting.

While our model is not captured by any of the previous papers in the bandit literature, the following papers are conceptually related. Tekin and Liu (2012) study a setting in which the reward from each arm is modeled as a finite-state Markov chain. Authors consider two cases: *rested* and *restless* arms. In the rested case, the state of each underlying Markov chain remains frozen unless the corresponding arm is played. While in our setting the arms are indeed rested, the reward process we study cannot be modeled by a *finite*-state Markov chain (see also (Neu et al., 2014)). Gabillon et al. (2013) study the problem of picking a subset of arms at each step in a setting where the reward is a submodular function of the chosen subset. Streeter et al. (2009) study the problem of assigning items to K positions such that a submodular utility function is maximized. Finally, our model is somewhat related to the line of research on best arm identification, however, previous studies on this topic mainly rely on stochastic assumptions on the rewards (see for example (Audibert and Bubeck, 2010),(Chandrasekaran and Karp, 2012)), and they do not apply to our setting.

There is much work in the psychology literature studying the human learning process on cognitive tasks (see e.g. (Atkinson et al., 1965; Mangal, 2009)), but to our knowledge ours is the first to model it in a bandits setting. There are however, a limited number of papers addressing relevant questions from a heuristic or empirical viewpoint (see (Basu and Christensen, 2013; Singla et al., 2013)).

3.2.1. Model and Preliminaries

The decision maker has access to n arms denoted $\{1, 2, ..., n\}$. At each time step t ($t = 1, 2, ..., T$) he has to decide which arm to pull. Every time the decision maker pulls an arm, he collects a reward and his goal is to pull the arms in such a way that his policy regret (to be formally defined shortly) is minimized. We assume the decision maker knows the time horizon T in advance.

We model the reward process of the arms as follows: Each arm k has a fixed underlying reward function denoted by $f_k(.)$. When the decision maker pulls arm k for the mth time ($m \geq 1$), he receives an instantaneous reward equal to $f_k(m)$. The cumulative reward from pulling arm k for m times is denoted by $F_k(m)$ and is equal to $f_k(1) + f_k(2) + ... + f_k(m)$. We assume that all the reward functions are bounded from below by 0 and from above by 1[11].

A *deterministic* policy π of length T is a sequence of mappings $\langle \pi_1, \pi_2, ..., \pi_T \rangle$ from the histories to the arms. That is,

$$\pi_t : \{1, 2, ..., n\}^{t-1} \times [0, 1]^{t-1} \longrightarrow \{1, 2, ..., n\}$$

prescribes the arm that must be pulled at step t given the history of actions and rewards observed so far. Given the reward functions $f_1(.), ..., f_n(.)$ and a deterministic policy π, the arm that the policy picks at each time step t is deterministic and is denoted by i_t. Let $b_k^t(\pi)$ be the instantaneous reward of arm k under policy π. More precisely, suppose the decision maker has followed policy π up until time $(t-1)$. $b_k^t(\pi)$ denotes the reward of playing arm k at time t. (As it is usually clear from the context what policy we are referring to, for simplicity we drop π in our notation.) Note the difference between $f_k(t)$ and b_k^t: unlike $f_k(t)$, b_k^t depends on the history of actions taken so far, in particular, the number of times arm k has been pulled before time t by policy π. We denote by $r(\pi)$ the total reward that

[11] It is easy to see that without this assumption no algorithm can guarantee sublinear policy regret.

a policy π collects, so $r(\pi) = \sum_{t=1}^{T} b_{i_t}^t$. Note that the total reward of a policy only depends on the number of times each arm is pulled and not the order.

Given the reward functions $f_1(.), ..., f_n(.)$ and T, let OPT be the policy that maximizes the total reward. In the online setting, the decision maker does not know the reward functions in advance and seeks to design a (possibly randomized) algorithm $\mathcal{A}$ so that the total reward he collects is as close as possible to that of OPT. In other words, the decision maker's goal is to minimize his *policy regret* which is defined as $r(\text{OPT}) - \mathbb{E}r(\mathcal{A})$. We say that an online algorithm $\mathcal{A}$ has sublinear policy regret if

$$\lim_{T\to\infty} \frac{r(\text{OPT}) - \mathbb{E}r(\mathcal{A})}{T} = 0.$$

3.2.2. Increasing Reward Functions

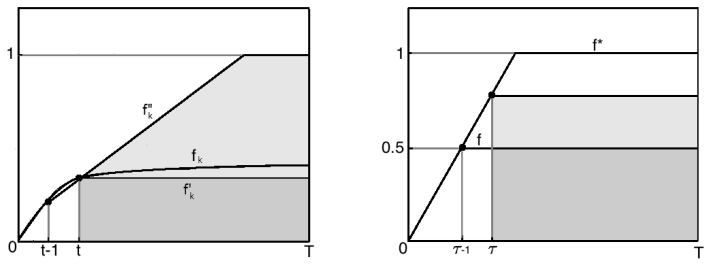
The Offline Setting

We first show that when the reward curves are all increasing, there exists a single best arm that the optimal policy must repeatedly pull. Despite this fact, merely having the guarantee of no external regret would not be sufficient to guarantee no policy regret here (see Example 1).

Proposition 5 *Suppose for all $1 \leq k \leq n$, $f_k(.)$ is increasing. Then there exists an arm k_T^* such that the optimal offline policy OPT consists of pulling k_T^* for all T trials.*

Proof Assume OPT pulls arm k, T_k times (so $\sum_{i=1}^{n} T_i = T$). Suppose there exist arms i, j for which $T_i, T_j > 0$. We claim that $f_i(0) = f_j(0) = f_i(T_i) = f_j(T_j)$. In other words, f_i, f_j are both flat and identical. If this holds, the policy OPT remains optimal if we replace every occurrence of i in it with j. Repeating this for any two arms i', j' with $T_{i'}, T_{j'} > 0$, we see that OPT consists of pulling a single arm T times.

To prove the above claim, we first note that since all the f_ks are increasing, it must be



(a) (b)

Figure 14: (a) The optimistic (light grey) and pessimistic (dark grey) estimate. (b) The lower bound example.

that $f_i(T_i) \geq f_j(T_j)$. Otherwise one could collect a reward larger than OPT by never pulling arm i and instead pulling arm j for $T_j + T_i$ times. But this is a contradiction with optimality of OPT. Similarly $f_i(T_i) \leq f_j(T_j)$ and therefore, we can conclude $f_i(T_i) = f_j(T_j)$. Next we observe that $f_i(0) \geq f_j(T_j)$. Otherwise, given that $f_i(T_i) \leq f_j(T_j)$, one could collect a reward larger than OPT by never pulling arm i and instead pulling arm j for T_i additional times. Combining this with the previous equality, we can conclude that $f_i(T_i) \geq f_i(0) \geq f_j(T_j) = f_i(T_i)$ and therefore, $f_i(0) = f_i(T_i)$. Similar argument holds for j as well. Therefore, $f_i(T_i) = f_i(0) = f_j(T_j) = f_j(0)$. This proves our claim and finishes the proof. ∎

The Online Setting

In addition to being increasing and bounded, we assume the learning curves satisfy *decreasing marginal returns* (see Section 3.2.4 for a discussion of why this assumption is needed). More precisely, for any arm k we assume

$$\forall t \geq 1 : f_k(t+1) - f_k(t) \leq f_k(t) - f_k(t-1)$$

If we think of the reward curves as continuous functions, the concavity of f_k would give us decreasing marginal returns. Therefore we slightly abuse the terminology and refer to this property as "concavity". We emphasize that the concavity assumption is very natural and common in the context of human learning (see for example (Son and Sethi, 2006))

and in particular, concave learning curves have been shown to arise in various laboratory environments (see for example (Jovanovic and Nyarko, 1995; Anderson and Schooler, 1991)).

For the case where the reward functions are all increasing, bounded and concave, we introduce an online algorithm whose policy regret bound, as we shall prove, is sublinear and optimal. Our algorithm is motivated by the following observation: Suppose we initially pull arm k, t times and observe $f_k(1), f_k(2), ..., f_k(t)$. After this, given that f_k is increasing, we can be sure that for any $t < s \leq T$, $f_k(s) \geq f_k(t)$. In other words, the additional reward that can be collected from arm k in the remaining $(T-t)$ steps is minimized if f_k flattens at t. We define the *pessimistic estimate* of the total reward from arm k (denoted by $q_k^t(T)$) to be equal to the total reward of a function f_k' that is identical to f_k up to t, and then flattens out[12] (see Figure 14 (a)).

In addition to the above lower bound, concavity yields an upper bound on future payoffs. The additional reward that can be collected from arm k in the remaining $(T-t)$ steps is maximized if the function continues to grow linearly with rate $(f_k(t) - f_k(t-1))$. We define the *optimistic estimate* of the total reward from arm k (denoted by $p_k^t(T)$) to be equal to the total reward of a function f_k'' that is identical to f_k up to t, and then continues to grow linearly with rate $(f_k(t) - f_k(t-1))$ until it hits 1 [13](See Figure 14 (a)).

Since all the reward curves are increasing, by Proposition 5 we know that there exists a single best arm that is repeatedly pulled by the optimal policy. Therefore we seek to detect this arm. Our algorithm operates as follows: it maintains a set of candidate arms in which the best arm is guaranteed to lie. At each round, it pulls all the arms in the candidate set exactly once, and updates both optimistic and pessimistic estimates of the reward from these arms. If at some round the optimistic estimate of an arm k is less than or equal to the pessimistic estimate of another arm in the candidate set, then we are sure that k can not be the optimal arm, and therefore the algorithm eliminates it from the candidate set.

[12]To be more precise, $q_k^t(T) = F_k(t) + f_k(t)(T-t)$.
[13]To be more precise, $p_k^t(T) = F_k(t) + \sum_{s=t+1}^{T} \min\{1, f_k(t) + (f_k(t) - f_k(t-1))\,(s-t)\}$.

ALGORITHM 4: The online algorithm for concave and increasing reward functions ($\mathcal{A}_1$)

$t = 0$;
% t is the index of the current phase. $S = \{1, ..., n\}$;
% S is the set of remaining candidate arms. Pull every arm exactly once;
repeat
  only one arm remains in S or time runs out;
until $t = t + 1$; % Start a new phase **for** *each* $i \in S$ **do**
  Pull arm i *once;*
  $p_i^t(T) = F_i(t) + \sum_{s=t+1}^{T} \min\{1, f_i(t) + (f(t) - f(t-1))(s-t)\}$;
  % Update the optimistic estimate. $q_i^t(T) = F_i(t) + f_i(t)(T-t)$;
  % Update the pessimistic estimate.
end
for $i \neq j \in S$ **do**
  if $q_j^t(T) > p_i^t(T)$ % i.e. there exists an arm whose total reward is guaranteed to be larger than that of arm i **then**
    $S = S \setminus \{i\}$;
  end
end
;
For the remaining steps (if any), pull the only arm left in S;

See algorithm 4 for the details. We refer to this algorithm by $\mathcal{A}_1$.

For the coming theorem, we make use of a quantity that captures the time required to distinguish the optimal arm from the others. More precisely, we define $\tau(T) = \max_k \tau_k(T)$ where

$$\tau_k(T) = \arg\min_t \left\{ q_{k_T^*}^t(T) > p_k^t(T) \right\}$$

and k_T^* is the optimal arm for the time horizon T. Thus $\tau_k(T)$ specifies the smallest number of times we need to pull both arm k and k_T^* so that the optimistic estimate of the total reward from arm k falls behind the pessimistic estimate of the total reward from arm k_T^*.

We now prove that the policy regret of $\mathcal{A}_1$ is bounded by $n\tau(T)$, and show that this is optimal. Eventually in Theorem 14 we shall prove the regret of $\mathcal{A}_1$ is in fact sublinear.

Theorem 13 *Suppose for all* $1 \leq k \leq n$, $f_k(.)$ *is concave, increasing, bounded from below*

by 0 and from above by 1. Then[14]

$$r(OPT) - r(\mathcal{A}_1) \leq n\tau(T).$$

Furthermore, the policy regret bound of $\mathcal{A}_1$ is optimal: there exists a family of examples consisting of bounded, increasing and concave reward functions for which no algorithm can be guaranteed to have a policy regret bound of $o(n\tau(T))$.

Proof Observe that $\mathcal{A}_1$ detects the optimal arm after at most $\tau(T)$ phases. This is simply because a suboptimal arm k is removed from S by phase $\tau_k(T)$. In addition, note that arm k_T^* can never be removed from S due to its optimality. Therefore, we can conclude that the number of times when $\mathcal{A}_1$ pulls suboptimals arm is at most $n\tau(T)$. Combining this with the fact that the reward per step is upper-bounded by 1 yields the desired inequality.

For the lower bound, fix a constant $\tau < T$, and consider n arms that all have linear reward curves with identical slope of $\frac{1}{2(\tau-1)}$, until they reach payoff 0.5 at time $(\tau - 1)$. Then $(n-1)$ of the curves stay at 0.5 forever, whereas one of them selected at random continues to 1 (see Figure 14 (b)). It is easy to see that for any $T > \tau$, $\tau(T) = \tau$. Using Yao's min-max principle (Yao, 1977), we lower-bound the regret of any deterministic algorithm on the above randomized setting. Let us define the number of arms an algorithm *verifies* to be equal to the number of arms that it pulls at least τ times (note that this is well-defined for any deterministic algorithm). It is easy to see that in this example, no algorithm can find the optimal arm with high probability by verifying $o(n)$ arms. To see this, first note that in order to see whether an arm is optimal, the algorithm has to pull it at least τ times. Assume, without loss of generality, that the first arm that the algorithm verifies is arm 1, the second one is arm 2, and so on. Since the index of the optimal arm is chosen uniformly at random, the expected number of arms the algorithm must verify before finding the optimal arm is equal to $\sum_{i=0}^{n-1} \frac{i}{n} = \frac{1}{n}\frac{n(n-1)}{2} = \frac{(n-1)}{2}$. Every time the algorithm verifies

[14] One can easily see that with the exact same logic as the one in the proof of Theorem 13, we can get a slightly better upper bound of $\sum_{k=1}^{n} \tau_k(T)$ on the regret, but to simplify the statement of the theorem and its proof, we work with the worst case upper bound of $n\tau(T)$.

a sub-optimal arm, it fails to collect at least 0.5τ units of reward from the optimal arm, and instead obtains a reward on average equal to 0.25τ. As a result the policy regret of the algorithm is lower bounded by $\frac{\tau(n-2)}{2}$. ∎

Next we show that the policy regret of Algorithm 4 is in fact sublinear for any choice of bounded, concave and increasing reward functions $f_1(.), ..., f_n(.)$. The following notation will be useful in our argument: we denote the asymptote of $f_i(.)$ by a_i, i.e.

$$a_i = \lim_{t\to\infty} f_i(t).$$

Note that since $f_k(.)$ is increasing and bounded, the asymptote exists and is finite. Also for any arm i $\lim_{T\to\infty} \frac{F_i(T)}{T} = a_i$. We define a^* to be $\max_{1\leq i\leq n} a_i$, and

$$\Delta_i(t) = f_i(t) - f_i(t-1).$$

Theorem 14 *For any set of bounded, concave, and increasing reward function $f_1(.), ..., f_n(.)$, the policy regret of $\mathcal{A}_1$ is sublinear.*

Here is the outline of the proof: We start by observing that for large values of T, the optimal arm must have the largest asymptote among other arms (note that we don't assume there is only one arm with maximum asymptote). Therefore a suboptimal arm i can be of one of the following two types:

1. The asymptote of arm i is smaller than that of the optimal arm; For this case, we show that $\mathcal{A}_1$ dismisses arm i from its candidate set within $o(T)$ phases. As a result the regret from pulling arm i cannot not large.

2. The asymptote of arm i is equal to that of the optimal arm; For this case, we show that while $\mathcal{A}_1$ may fail to determine the suboptimality of arm i quickly, since the asymptote of i is as large as the optimal arm, pulling it does not add much to the policy regret of $\mathcal{A}_1$.

Proof Observe that $\lim_{T\to\infty} a_{k_T^*} = a^*$; in other words, if T is sufficiently large, then $k_T^* \in \arg\max_{1\le i\le n} a_i$. Throughout, we assume T is large enough so that the latter holds.

Let $W = \arg\max_k a_k$. We start by showing that if $i \notin W$ (i.e. $a_i < a^*$), then $\mathcal{A}_1$ removes arm i from its candidate set within $o(T)$ phases.

Lemma 5 *For arm i, if $a_i < a^*$, then $\lim_{T\to\infty} \frac{\tau_i(T)}{T} = 0$.*

Proof Suppose that the statement of the lemma does not hold and $\lim_{T\to\infty} \frac{\tau_i(T)}{T} \neq 0$. This means that for any unbounded and increasing sequence $\{\gamma_T\}_{T=1}^{\infty}$ for which $\lim_{T\to\infty} \frac{\gamma_T}{T} = 0$, there exists an infinite subsequence of T's such that $q_{k^*}^{\gamma_T}(T) < p_i^{\gamma_T}(T)$. Expanding this, we have[15]

$$\begin{aligned} F_{k_T^*}(\gamma) + (T-\gamma) f_{k_T^*}(\gamma) &\le F_i(\gamma) + \sum_{s>\gamma} \min\{1, f_i(\gamma) + \Delta_i(s-\gamma)\} \\ &\le F_i(\gamma) + \sum_{s>\gamma} f_i(\gamma) + \Delta_i(s-\gamma) \end{aligned}$$

The above is equivalent to

$$\begin{aligned} \left(F_{k_T^*}(\gamma) - F_i(\gamma)\right) + (T-\gamma)\left(f_{k_T^*}(\gamma) - f_i(\gamma)\right) &\le \sum_{s>\gamma} \Delta_i(s-\gamma) \\ &\le \Delta_i \frac{(T-\gamma+1)^2}{2} \\ &\le \Delta_i \frac{T^2}{2} \end{aligned}$$

Therefore combined we have:

$$F_{k_T^*}(\gamma_T) - F_i(\gamma_T) + (T-\gamma_T)\left(f_{k_T^*}(\gamma_T) - f_i(\gamma_T)\right) < \Delta_i(\gamma_T)\frac{T^2}{2} \tag{3.1}$$

Now note that if T (and as a result γ_T) is large enough, then $F_{k_T^*}(\gamma_T) \ge F_i(\gamma_T)$ and $f_{k_T^*}(\gamma_T) - f_i(\gamma_T) \ge \frac{a_{k_T^*} - a_i}{2} = \frac{a^* - a_i}{2}$. Let $C = \frac{a^*-a_i}{2}$ (recall that due to our assumption about arm i, $C > 0$). Therefore from (3.1) we obtain that $C(T-\gamma_T) < \Delta_i(\gamma_T)\frac{T^2}{2}$ and as a

[15] To simplify the notation in this equation we drop the subscript T. Also by δ_i we mean $\Delta_i(\gamma)$.

result

$$\begin{aligned} & C(T-\gamma) < \Delta_i(\gamma)\frac{T^2}{2} \\ \Rightarrow \quad & \frac{C(T-\gamma)}{T} < \Delta_i(\gamma)\frac{T}{2} \\ \Rightarrow \quad & \lim_{T\to\infty} \frac{C(T-\gamma)}{T} \leq \lim_{T\to\infty} \Delta_i(\gamma)\frac{T}{2} \\ \Rightarrow \quad & C \leq \lim_{T\to\infty} \Delta_i(\gamma)\frac{T}{2} \end{aligned}$$

where the last inequality follows from the fact that $\lim_{T\to\infty} \frac{\gamma_T}{T} = 0$. So we have

$$C \leq \lim_{T\to\infty} \Delta_i(\gamma_T)\frac{T}{2} \tag{3.2}$$

Next, one can easily verify that $\lim_{T\to\infty} \Delta_i(\gamma_T)T = 0$. To see this note that

$$\begin{aligned} & \sum_{t=1}^{\infty} \Delta_i(t) \leq 1 \\ \Rightarrow \quad & \sum_{t=1}^{\infty} \Delta_i(t)T \leq T \\ \Rightarrow \quad & \lim_{t\to\infty} \Delta_i(t)T = 0 \\ \Rightarrow \quad & \lim_{t\to\infty} \Delta_i(\gamma_T)T = 0 \end{aligned}$$

where the last inequality follows from the fact that $\{\gamma_T\}_{T=1}^{\infty}$ is positive, increasing and unbounded. This combined with (3.2) yields $C \leq 0$, which is a contradiction. ∎

Using Lemma 5, we can conclude that for any arm $i \notin W$ our algorithm can distinguish i from k^* within $o(T)$ phases. In other words, after a prefix of at most $o(T)$ many phases, $\mathcal{A}_1$ eliminates every arm $i \notin W$.

Second we show that if $i \in W$ (i.e. $a_i = a^*$), while $\mathcal{A}_1$ may fail to detect the suboptimality of arm i quickly, pulling arm i does not add much to the policy regret of $\mathcal{A}_1$.

Lemma 6 *Let T_i denote the number of times $\mathcal{A}_1$ pulls arm $i \in W$. Then*

$$\lim_{T\to\infty} \frac{F_{k_T^*}(T) - \sum_{i\in W} F_i(T_i)}{T} = 0.$$

Proof If $i \in W$ gets eliminated within $o(T)$ phases, it can not cause the algorithm to suffer from a linear policy regret. Now consider a subset W' of W consisting of any suboptimal arm i for which it takes $\Theta(T)$ steps for the algorithm to eliminate i.

Given that the arms not in W' can all be detected in $T' = o(T)$ time steps, we have that $T - \sum_{i\in S'} T_i = T' = o(T)$. Let $w \in W'$ be the arm for which $\frac{F_i(T_i)}{T_i}$ is the smallest. We have

$$\begin{aligned}
F_{k_T^*}(T) - \sum_{i\in W'} F_i(T_i) &= F_{k_T^*}(T) - \sum_{i\in W'} T_i \frac{F_i(T_i)}{T_i} \\
&< F_{k_T^*}(T) - \sum_{i\in W'} T_i \frac{F_w(T_w)}{T_w} \\
&= T\frac{F_{k_T^*}(T)}{T} - (T - T')\frac{F_w(T_w)}{T_w} \\
&= T(\frac{F_{k_T^*}(T)}{T} - \frac{F_w(T_w)}{T_w}) + T'\frac{F_w(T_w)}{T_w} \\
&\leq T\left(\frac{F_{k_T^*}(T)}{T} - \frac{F_w(T_w)}{T_w}\right) + o(T)
\end{aligned}$$

That is,

$$F_{k_T^*}(T) - \sum_{i\in W'} F_i(T_i) \leq T\left(\frac{F_{k_T^*}(T)}{T} - \frac{F_w(T_w)}{T_w}\right) + o(T) \tag{3.3}$$

It only remains to show that

$$T\left(\frac{F_{k_T^*}(T)}{T} - \frac{F_w(T_w)}{T_w}\right) = o(T) \tag{3.4}$$

But this is certainly true as $\lim_{T\to\infty} \frac{F_{k_T^*}(T)}{T} = \lim_{T_w\to\infty} \frac{F_w(T_w)}{T_w} = a^*$. Combining (3.3), (3.4), we have the desired result. ∎

Combining Lemma 5 and 6 we obtain the sublinearity of the policy regret for $\mathcal{A}_1$. ∎

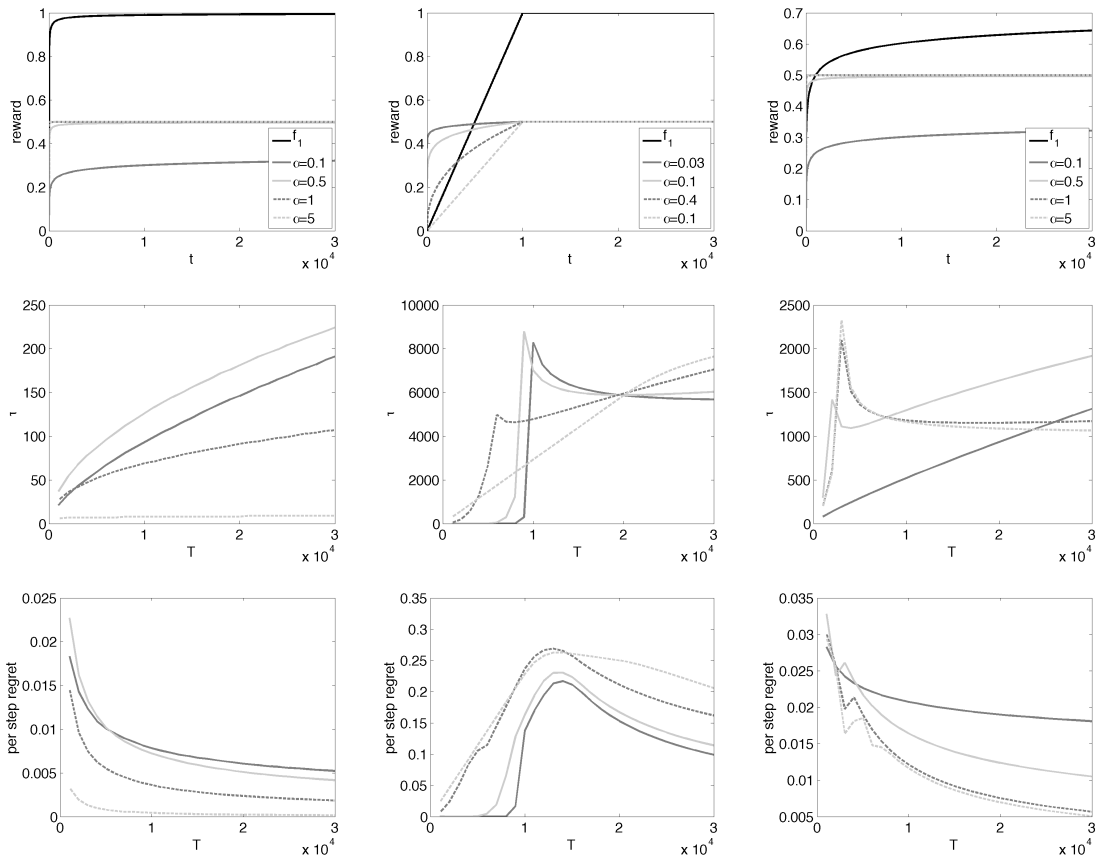


Figure 15: The first row illustrates f_1 (black) and f_2 (grey curves); the second and third rows illustrate τ and the regret respectively.

Finally, we remark that if the decision maker only gets to observe a corrupted version of the rewards and the corruption is bounded by some $\epsilon > 0$, then $\mathcal{A}_1$ is guaranteed to have a total reward at least equal to $r(\text{OPT}) - n\tau^{\epsilon}(T)$. $\tau^{\epsilon}(T)$ is the corrupted version of $\tau(T)$ in which the optimistic and pessimistic estimate from an arm are computed by taking the presence of noise into account [16] .

Simulations

So far we have established two upper bounds on the regret of $\mathcal{A}_1$: in Theorem 13 we gave an upper bound of $n\tau(T)$, and in Theorem 14 we showed that the regret of is always sub-linear. In this section we empirically investigate the performance of $\mathcal{A}_1$ on several illustrative reward

[16] More precisely, $\tau^{\epsilon}(T) = \max_k \tau_k^{\epsilon}(T)$ and $\tau_k^{\epsilon}(T) = \arg\min_t \{q_{k_T^*}^{t,\epsilon}(T) > p_k^{t,\epsilon}(T)\}$ where $p_k^{t,\epsilon}(T) = F_k(t) + \sum_{s=t+1}^{T} \min\{1, f_k(t) + \epsilon + (f(t) - f(t-1) + 2\epsilon)\,(s-t)\}$ and $q_k^{t,\epsilon}(T) = F_k(t) + (f_k(t) - \epsilon)(T-t)$.

curves, and observe that $\tau(T)$ (and the regret of our algorithm) are typically *significantly* sublinear in T.

Throughout, for simplicity we set n to 2 and consider two arms $1, 2$ where the asymptote of f_1 is 1 and that of f_2 is 0.5. In Figure 15, we report the value of regret and τ versus $T = 500, ..., 30000$ for three different sets of examples.

In the first column of Figure 15, $f_1(t) = 1 - t^{-0.5}$ and $f_2(t) = 0.5 - 0.5t^{-\alpha}$ where $\alpha = 0.1, 0.5, 1, 5$. Each of these values corresponds to a different rate of increase for f_2. The larger α is, the faster f_2 converges to its asymptote. In this example for all values of α, τ increases slowly (sub-linearly) with T and as a result regret consistently decreases with T. The reason for the slow growth of τ is that f_1 converges to its asymptote very quickly; in addition, the rate with which f_2 increases, approaches 0 fast. This enables the algorithm to find and dismiss the suboptimal arms early on.

In the second column, $f_1(t) = \min\{1, \frac{t}{30000}\}$ and $f_2(t) = \min\{0.5, 0.5(\frac{t}{30000})^{\alpha}\}$ where $\alpha = 0.03, 0.1, 0.4, 1$. The smaller α is, the faster f_2 converges to its asymptote. Here when τ increases linearly or faster with T, the regret increases as well. Note that this does not contradict Theorem 14 as the theorem holds for large enough values of T only. Notice that for $\alpha = 0.03, 0.1, 0.4$, τ spikes at around $T = 10000$ and then drops. The reason for this behavior is that at that point, the optimal arm changes from arm 2 to arm 1.

Finally in the third column of Figure 15, $f_1(t) = 1 - t^{-0.1}$ and $f_2(t) = 0.5 - 0.5t^{-\alpha}$ where $\alpha = 0.1, 0.5, 1, 5$. It might come as a surprise that in this example, at the points when τ peaks, regret actually drops. Of course this does not contradict Theorem 13. While this theorem guarantees small regret when τ grows slowly with T, if τ increases linearly or faster, it does not necessarily imply that Algorithm 4 must suffer a large regret. For example, when $\alpha = 5$ and $T \leq 2500$, arm 1 is the sub-optimal arm, however, its reward is just slightly worse than that of the optimal arm (arm 2). Given that the rate of increase of

the sub-optimal arm is sufficiently large, the algorithm fails to detect the optimal arm and instead keeps pulling the two an equal number of times. However, since the reward from the sub-optimal arm is getting closer to that of the optimal arm, the regret decreases. Note that this was not the case for the example in column 2.

3.2.3. Decreasing Reward Functions

Optimal Offline Policy

When all the reward functions are decreasing, there exists a simple greedy approach that can compute the optimal offline policy. The proposed policy, $\mathcal{A}_0$, works as follows: At each time, pull the arm that results in the highest instantaneous reward. More precisely, if up to time step t, arm i has been pulled t_i times ($1 \leq i \leq n$), then pull an arm in $\arg\max_i f_i(t_i + 1)$.

Proposition 6 *Suppose for all* $1 \leq k \leq n$, $f_k(.)$ *is decreasing. Then* $r(\mathcal{A}_0) = r(OPT)$.

Proof We prove that $\mathcal{A}_0$ maximizes the total reward for any set of reward functions using induction on T. If $T = 1$, it is obvious that the optimal action is to pull the arm that results in the highest instantaneous reward; more precisely, in order to maximize the reward, one should pull arm k^* where $k^* \in \arg\max_k f_k(1)$ and this is exactly what $\mathcal{A}_0$ does. Suppose that our claim holds for $T \leq m$. Now consider the case of $T = m + 1$. Consider the policy OPT of length $(m + 1)$ that maximizes the total reward for the given set of reward functions $f_1, ..., f_n$. Let $k^* \in \arg\max_k f_k(1)$ be the arm that $\mathcal{A}_0$ initially pulls. We first show that, without loss of generality, we can assume OPT pulls arm k^* at least once. Suppose it does not. Consider the last action that OPT takes. Suppose it pulls arm k for the T_k^*'th times. Due to the fact that f_k is decreasing and due to the definition of k^*, we have $f_k(T_k) \leq f_k(1) \leq f_{k^*}(1)$. In other words if instead of k, in the last step OPT pulls k^*, the total reward it collects can only improve. Therefore without loss of generality we can assume OPT pulls arm k^* at least once.

Now let $g_{k^*}(t) = f_{k^*}(t+1)$. Remove the first occurrence of k^* from OPT, and call the remaining policy π. Obviously, π, which is of length m, must maximize the total reward from arms with reward curves $f_1, ..., g_{k^*}, ..., f_n$, otherwise the total reward of OPT could be increased and this contradicts the optimality of OPT. Now applying the induction hypothesis, we know that $\mathcal{A}_0$ also maximizes the total reward for $f_1, ..., g_{k^*}, ..., f_n$. Therefore we can conclude that the total reward that $\mathcal{A}_0$ collects is equal to that of π. This finishes the proof. ∎

The Online Setting

As the above theorem shows, in the case of decreasing reward functions, there does not necessarily exist a single best arm that an online algorithm should track. Rather, the optimal algorithm needs to quickly react and switch to another arm when the reward from the current one drops.

We introduce a greedy online algorithm that can guarantee constant (and therefore sublinear) policy regret when all the reward curves are decreasing. The algorithm, which we call $\mathcal{A}_2$, does not need to know T in advance and works as follows: Intuitively, after the initial prefix of n actions, $\mathcal{A}_2$ chooses a policy that is *identical* to the optimal greedy policy. The only difference is that for each action, it is getting a reward that is diminished by one additional time unit compared to the offline optimal benchmark. In sum, compared to the optimal algorithm, it loses at most one unit of reward from each action. But each instantaneous reward is bounded in $[0, 1]$, so the loss is at most n. See Algorithm 5 for further details.

Theorem 15 *Suppose for all $1 \leq k \leq n$, $f_k(.)$ is decreasing, bounded from below by 0 and from above by 1. Then*

$$r(OPT) - r(\mathcal{A}_2) \leq n.$$

Furthermore, the policy regret bound of $\mathcal{A}_2$ is optimal: there exists a family of examples con-

ALGORITHM 5: A no policy regret algorithm for decreasing reward functions ($\mathcal{A}_2$)

for $1 \leq k \leq n$ **do**
  Pull arm k once;
  $m_k = 1$;
  %m_k is the number of times arm k has been pulled so far $\mathcal{R}_k = f_k(1)$;
  %$\mathcal{R}_k$ is the most recent instantaneous reward collected from arm k
end
for $n+1 \leq t \leq T$ **do**
  Pull arm j where $j \in \arg\max_k \mathcal{R}_k$;
  $\mathcal{R}_j = f_j(m_j + 1)$;
  $m_j = m_j + 1$;
end

sisting of bounded and decreasing reward functions for which no algorithm can be guaranteed to have a policy regret bound of $o(n)$.

Proof Suppose for all $1 \leq k \leq n$, OPT pulls arm k, T_k^* times and $\mathcal{A}_2$ pulls it T_k times. First note that for any arm k and any $t \leq \min\{T_k^*, T_k\}$, both OPT and $\mathcal{A}_2$ receive an instantaneous reward equal to $f_k(t)$ when they pull arm k for the t'th time. Thus these two instantaneous rewards cancel each other out in $r(\text{OPT}) - r(\mathcal{A}_2)$. It only remains to investigate the actions that $\mathcal{A}_2$ and OPT differ on.

Let U be the subset of arms for which $T_k > T_k^*$ and V be the subset of arms for which $T_k^* > T_k$. Note that if one of U or V is non-empty, the other one has to be non-empty as well. For any arm $i \in V$, let ℓ_i be the last instantaneous reward $\mathcal{A}_2$ collects from arm i, i.e. $\ell_i = f_i(T_i)$. Let $i^* = \arg\max_{i \in V} \ell_i$. Now consider the time when $\mathcal{A}_2$ pulls arm $j \in U$ for the tth time where $(T_j^* + 1) < t \leq T_j$ (if no such t exists, then $T_j^* + 1 = T_j$ which means $\mathcal{A}_2$ pulls arm j just once more than OPT). Suppose that so far, $\mathcal{A}_2$ has pulled arm i^*, $s \leq T_{i^*}$ times. Given that $\mathcal{A}_2$ chooses to pull arm j, we know that at that time $\mathcal{R}_j \geq \mathcal{R}_{i^*}$, or equivalently, $f_j(t-1) \geq f_{i^*}(s)$. Also, given that f_k's are all decreasing, we have that $f_{i^*}(s) \geq f_{i^*}(T_{i^*})$. Due to the way i^* has been chosen, we know that for all $i \in V$, $\ell_{i^*} \geq \ell_i$, or equivalently, $f_{i^*}(T_{i^*}) \geq f_i(T_i)$. Given that f_k's are all decreasing, we have that for all $i \in V$ and $r \geq T_i$, $f_i(T_i) \geq f_i(r)$. Combining the last four inequalities, we have that for all $i \in V$ and $r \geq T_i$, $f_j(t-1) \geq f_i(r)$. This inequality means that except for at most 1 step (i.e. the T_jth time

$\mathcal{A}_2$ pulls arm $j \in U$), those extra times at which $\mathcal{A}_2$ pulls arm j results in a reward larger than any of the instantaneous rewards that OPT collects and $\mathcal{A}_2$ doesn't. In other words $\mathcal{A}_2$ *wastes* at most 1 time step on arm j. Given that the rewards are all upper bounded by 1, we can conclude that $r(OPT) - r(\mathcal{A}_2) \leq |U| \leq n$.

For the lower bound, consider n arms, $(n-1)$ of which have a reward function equal $f(t) = \mathbf{1}[t = 1]$; and one of them (which we call k^*) chosen uniformly at random has a reward function always equal to 1. Note that in order to verify whether an arm is the optimal arm, any online algorithm has to pull it at least twice. If a suboptimal arm is pulled for the second time, we say the algorithm made a *mistake*. It is easy to see that in this example, no online algorithm can be guaranteed to find the optimal arm by making $o(n)$ mistakes in expectation. Assume without loss of generality that the first arm that the algorithm verifies (i.e. pulls for the second time) is arm 1, the second one is arm 2, and so on. Since the index of the optimal arm is chosen uniformly at random, the expected number of mistakes the algorithm makes before detecting k^* is equal to $\sum_{i=0}^{n-1} \frac{i}{n} = \frac{(n-1)}{2}$. This means that the algorithm makes $\Theta(n)$ mistakes in expectation. Given that every time the algorithm makes a mistake, it loses 1 unit of reward, the policy regret of the algorithm is lower bounded by $\frac{(n-1)}{2}$. This finishes the proof. ∎

We conclude with a few remarks: First, one can easily see that if the decision maker only gets to observe a corrupted version of the rewards and the magnitude of the corruption is bounded by $\epsilon > 0$, then $\mathcal{A}_2$ is guaranteed to have a total reward at least equal to $r(\text{OPT}) - n - \epsilon T$.

Second, we note that the greedy approach presented here fails to perform well in the increasing setting, since there the optimal arm can have the *lowest* payoff at time 1, and therefore never gets pulled by the greedy algorithm.

3.2.4. Discussion and Future Directions

We presented no-policy regret algorithms for two settings: one in which all the reward functions were concave and increasing, and another where all the rewards were decreasing. We saw that even this simplified setting leads to non-trivial questions. We consider our work as a first step towards designing no-regret algorithms in settings where the rewards are neither stochastic nor fully adversarial.

Here are some interesting open questions raised by the results presented here.

- **Non-concave improving bandits.** The concavity assumption allows one to infer upper and lower bounds on the total reward of each arm with certainty. This is what our analysis relied on heavily to guarantee no policy regret. The same idea cannot be readily applied to settings where the rewards are increasing but non-concave. The existence of a no-policy regret algorithm for such settings remains an interesting open question.

- **Bandits that improve initially, then decay.** We assumed either all the reward functions are increasing or they are all decreasing. A natural and interesting question is whether it is possible to guarantee no policy regret in the case where the learning curves are concave and increasing at first, but then start decaying from some point on. It is easy to see that this is not simply achievable by naively combining the algorithms presented here.

- **Statistical noise.** Our algorithms can handle small amounts of adversarially generated noise. An important question is whether a similar regret bound can be shown to hold if the noise is generated stochastically. One natural idea to extend our work to this setting is to *estimate* the gradient of the reward functions and the corresponding confidence intervals using past observations, then follow a logic similar to what we proposed here.

CHAPTER 4 : Competition for Market Share

A major challenge for any online business is to ensure that a sufficient proportion of potential customers prefers their service to that of a competitor. The analysis of this competition is the subject of this chapter[1].

Chapter 4 consists of two sections. In Section 4.1, we address market competition in the context of *ad exchanges*. As the number of exchanges has grown, sellers have turned to low-regret learning mechanisms to decide which exchange has the best price for their inventory. This in turn raises the following question for the exchange: how to set reserve prices to attract a large market share and maximize revenue? My collaborators and I formulate this as a learning problem, and present algorithms showing that simply knowing that sellers use low-regret learning to choose among their options is enough for the exchange to have a low-regret algorithm for the optimal price (Heidari et al., 2016b).

Next and motivated by the word-of-mouth and viral marketing, in Section 4.2 my coauthors and I consider a setting in which two firms compete for the consumers located in a social network. Firms have budgets to "seed" the initial adoption of their products/services, and their goal is to maximize their market share. This defines a game among firms. In (Goyal et al., 2014), we identify general properties of the adoption dynamics—namely, decreasing returns to local adoption—which determine whether the inefficiency of resource use at equilibrium (quantified by *the price of anarchy* and *stability*) is bounded across all networks. We also test the sensitivity of these results to the changes in the structure of the utility functions (see (Draief et al., 2014)). Building on the framework introduced in (Goyal et al., 2014), we first introduce a new model in which the payoff to each firm comprises not only the number of vertices who adopt its product, but also the network connectivity among those nodes. For a general class of stochastic dynamics driving the local adoption process, we derive upper bounds on (1) the (pure strategy) Price of Anarchy and Budget Multiplier.

[1]The content of Sections 4.2 and 4.1 is taken directly from Goyal et al. (2014); Draief et al. (2014) and Heidari et al. (2016b), respectively.

Here the bounds we obtain depend on the budgets and the maximum degree of the network, but no other structural properties. We also introduce a model in which budgeting decisions are endogenous, rather than externally given as is typically assumed in most viral marketing models. In sharp contrast to the previous results, we show that for almost any local adoption dynamics, there exists a family of graphs for which the price of anarchy is unbounded.

4.1. Pricing a Low-regret Seller in Ad Exchanges

A display ad exchange (e.g. DoubleClick, AdECN, and AppNexus) is a platform that facilitates buying and selling of display advertising inventory connecting multiple publishers and advertisers. Publishers can select an exchange to serve an impression each time a user visits one of their websites. Upon receiving an ad slot, the exchange sells it to one of their advertisers—often by running an auction among real-time bidding agents—and pays the publisher an amount based on the revenue generated from the ad.

With the recent growth in the number of ad exchanges, one important decision a publisher has to make is which one of these exchanges to enlist in order to sell their inventory for the highest price. Unlike traditional settings where prices are posted, in display advertising the publisher cannot simply observe the offered prices in advance and choose the highest paying exchange. There are multiple reasons behind this constraint: First, on the exchange side each price check often involves running an auction and allocating the impression to the winner. As the result the publisher cannot send the same item to multiple exchanges at the same time. This combined with the fact that there is very limited time—on the order of a few milliseconds—to serve an ad to the user, forces the publisher to commit to using a particular exchange before observing the prices.

Given that prices cannot be observed in advance, in order to pick the highest paying exchange publishers have to rely on experimentation, utilizing different exchanges and seeing the payoffs realized from each over time. In recent years great progress has been made to

automate decision making processes in such settings. The so called *bandit algorithms* automatically explore between the multitude of available options (here exchanges) and exploit the most profitable ones. These algorithms are easy to implement, are incredibly practical, and come with strong theoretical guarantees on the *regret* of the operator (here publisher). Therefore, from the point of view of the publisher, the situation is largely resolved.

From the point of view of an exchange, however, it is far from clear what strategy it must employ to maximize revenue. In an ideal world the exchange could look at the prices offered to the publisher by its competitors, and set the offering price ever so slightly higher. Any strategic publisher (e.g. one minimizing regret) would then shift their inventory towards this exchange, rewarding them for the higher prices. In practice, however, these prices are not publicly announced and there is no easy way to discover them. For instance, because of cookie based targeting, it is not possible for the exchange to simply find a 'similar' impression on one of the competing platforms and check its price. Given that the exchange cannot observe the competing prices directly, the only way to infer and react to them is through the actions of the publisher.

Faced with a publisher who selects among exchanges using a no-regret algorithm, the operator of an exchange must carefully decide what prices to offer. If the prices are too low, the publisher will never select the exchange, and if the prices are too high, the exchange is overpaying. Our goal in this work is to design a no regret pricing algorithm for the exchange.

We assume the prices offered by the competitors is drawn from an unknown distribution. This assumption is required for the existence of a no regret pricing algorithm. Furthermore, we believe that this assumption is in fact realistic: given that each exchange has a large number of competitors, the response of an individual exchange will not have a significant impact on the aggregate distribution of the competing prices (this is similar to the reasoning behind *mean-field equilibria*).

The first solution that comes to mind is to discretize the price space and run an off-the-shelf no regret algorithm in order to find the best price. As we will show in Section 4.1.4 this approach does not solve the problem. The main result of the current paper is a binary-search pricing algorithm that guarantees the exchange pays only a little more than the best price offered by its competitors, even though it never observes these prices directly (Section 4.1.2,4.1.3).

Related Work

We study a setting in which a seller repeatedly interacts with a group of buyers, deciding at each time step which buyer to sell the next item to. In this setting a natural choice for the seller is to employ low regret bandit learning algorithms (Lai and Robbins, 1985; Auer et al., 2002a,b). Bandit algorithms are a popular solution to sequential decision making problems, as they require only limited feedback and have low regret, i.e., they guarantee performance comparable to the best single action in hindsight. While some of the earliest bandit algorithms were index-based (Lai and Robbins, 1985; Auer et al., 2002a), the EXP family of algorithms (Auer et al., 2002b) are designed for bandit problems where the feedback is generated arbitrarily, rather than stochastically. Bayesian bandit algorithms based on Thompson sampling (Thompson, 1933) have also been very successful empirically (Graepel et al., 2010; Chapelle and Li, 2011).

The focus of the present paper is to design a no-regret pricing scheme for a buyer who interacts with a strategic seller over multiple time periods. The most closely related to our work are the results in (Amin et al., 2013, 2014). These papers study a repeated posted-price auction setting consisting of a single strategic buyer and a price-setting seller. The main results in (Amin et al., 2013, 2014) are pricing algorithms for the seller that guarantee no regret if the buyer's discounting factor is small. Compared to our work, Amin et al. define regret with respect to different benchmarks. Also in contrast to our model, they assume buyer's valuation is subject to time discounting, with non-trivial regret achievable

only when the discount rate is strictly less than 1.

Our work is also related to the broad literature on repeated auctions, where an auctioneer interacts with buyers and sellers over multiple time steps. Repeated auctions have been studied extensively and from various angles (Bikhchandani, 1988; Thomas, 1996; Chouinard, 2006). Both empirical (Edelman and Ostrovsky, 2007) and anecdotal evidence have suggested that in repeated auctions agents use sophisticated algorithms to induce better payoffs for themselves in the future. Indeed a growing part of the literature has been dedicated to designing various strategies and algorithms to improve the future payoff (Jofre-Bonet and Pesendorfer, 2000; Kitts and Leblanc, 2004; Kitts et al., 2005; Cary et al., 2007; Lucier, 2009; Gummadi et al., 2012). Our work is in particular concerned with the study of pricing in repeated auctions. Some of the previous papers on this topic are (Bar-Yossef et al., 2002; Kleinberg and Leighton, 2003; Blum et al., 2004; Cesa-Bianchi et al., 2013; Mohri and Medina, 2014). These papers mostly consider a simplified setting, focus on the buyer (and not the seller) side, and assume the buyer behaves in a naive manner. Our work is also related to the study intertemporal price discrimination, i.e. conditioning the price on buyer's past behavior in a repeated auction. Previous work, for instance Acquisti and Varian (2005); Kanoria and Nazerzadeh (2014) examine the conditions under which it is profitable to engage in this form of pricing.

Finally, we remark that the current paper adds to the growing line of research in algorithmic game theory investigating the outcome of games in which players employ some form of no-regret learning (Roughgarden, 2012; Syrgkanis and Tardos, 2013; Nekipelov et al., 2015). As opposed to classic economics where players are assumed to have reached an equilibrium, this recent body of work relies on the weaker assumption that players utilize no regret learning to learn from their past observations and adjust their strategies. This idea is compelling especially in online settings, such as the one studied in this work, where players repeatedly interact with one another in a complex and dynamic environment. Our work presents an algorithmic *no-regret response* against a no-regret opponent in an auction environment.

4.1.1. Model and Preliminaries

We consider a setting where a seller repeatedly interacts with a group of price-setting buyers, deciding at each time step which buyer to sell the next item to. In the context of display advertising, sellers and buyers correspond to publishers and ad exchanges, respectively; each time step represents an instance where a user visits the publisher's website and gives the publisher an advertising opportunity to sell at any of the advertising exchanges. In practice, an ad exchange is often an intermediary who runs an auction among advertisers to allocate the ad, and then determines how much to pay the publisher (typically based on the revenue generated from this auction). Nonetheless, by modeling the exchange as a buyer we implicitly assume it has full control over how much the publisher (seller) is paid. We argue that this assumption is practical for multiple reasons: First, the amount the exchange pays the publisher does not have to be tied to the amount it receives from the advertisers[2]. Second, even if two are closely related, e.g. if the exchange decides to pay the publishers a fixed percentage of the revenue, it only needs to satisfy this constraint in sum across all impressions[3]. Third, the exchange has full control over the reserve price, which in practice often directly affects the auction revenue.

Consider a seller selling one unit of an identical good at each time step to a group of price-setting buyers. Time is assumed to be discrete and indexed by positive integers. We study the pricing problem from the perspective of a buyer interested in this good. At each time step, the seller must select whether to sell the good to us, or to one of the *outside options*. If the seller does not select us, which outside option it chooses does not affect our revenue. Therefore, without loss of generality, we represent the outside option with a single buyer. Let us denote the price offered by us (A) and by the outside option (B) for the good at time t by p_t^A and p_t^B, respectively. We assume at each time step the seller must select

[2]Of course, the amount collected from the advertisers determines the "value" that the exchange has for receiving the ad slot, but this will be captured in our model by the parameter v, the buyer's value for the good.

[3]Specifically, the exchange can take on the arbitrage risk, by promising the publisher a minimum price, and recouping the cost later if needed.

between A and B *before* seeing these prices. Once it picks a buyer, the seller will observe the price offered by that buyer. Note that while this would be an odd assumption in standard marketplaces, as noted earlier in the case of the online advertising market, it is standard practice: First, the publisher cannot send the same item to multiple exchanges at the same time. Second, once a user requests a page on the publisher's website, they must quickly be served an ad, therefore the publisher simply does not have enough time to check prices at multiple exchanges.

Since the seller cannot see the prices before selecting which buyer to choose, it employs a low-regret strategy to select the buyer that over time gives her a higher price. The *regret* of the seller up to time T is defined as

$$R(T) = \max\{\sum_{t=1}^{T} p_t^A, \sum_{t=1}^{T} p_t^B\} - \sum_{t=1}^{T} p_t^{X_t},$$

where $X_t \in \{A, B\}$ is the buyer chosen by the seller in time step t. We assume the seller uses a (possibly randomized) low-regret[4] algorithm to pick X_t's. We need to be careful about the definition of *low regret* here: in our setting we need the regret to be bounded not just in expectation, but with high probability. We follow the definition in (Bubeck and Cesa-Bianchi, 2012), and assume the seller's strategy satisfies the following: for every $\delta > 0$, with probability at least $1 - \delta$, seller's regret up to time T is

$$R(T) < cT^{\gamma} \log(\delta^{-1}), \tag{4.1}$$

where c and $\gamma < 1$ are constants (independent of T and δ). The standard adversarial multi-armed bandits algorithms (Bubeck and Cesa-Bianchi, 2012) satisfy the above bound with any $\gamma > \frac{1}{2}$.[5]

The pricing problem can be defined as follows: at each time step t, we (as a buyer) would

[4] Or sublinear regret.

[5] More precisely, the EXP3.P algorithm satisfies the regret bound with $\gamma = \frac{1}{2}$ and an additional polylog term on the right hand side.

like to set a price p_t^A. All we can observe at the end of each round is the actions of the seller, i.e. whether we are selected or not. Note that in practice we cannot directly observe when the publisher chooses our opponent (exchange A does not get a call every time the publisher sends an impression to exchange B), nonetheless it is relatively easy for exchange A to know the approximate amount of traffic the seller sends to other exchanges. This can be done with either estimating the overall traffic the publisher receives, or by randomly monitoring the publisher's website and observing the fraction of times the ads on the page are served by exchange A.

We assume the price of the outside option p_t^B is drawn i.i.d. from an unknown distribution $\mathcal{D}$ with mean $\mu \in [0, 1]$. Note that in large market places it is a common practice (see for example the literature on mean field equilibrium) to assume each player treats other players' strategies as sampled from a fixed distribution. The assumption that p_t^B's are drawn stochastically is necessary for the existence of a low regret pricing algorithm. We don't get to observe our competitor's prices.

Let v be our value for each unit of the good (v can be thought of as the value we can get from the advertisers in our exchange for an advertising opportunity on this publisher). For simplicity, we treat v as a constant value, but our results generalize to the case that v is a random variable drawn i.i.d. from a distribution.[6] A clairvoyant algorithm that knows μ can simply offer a constant price slightly higher than μ. At this price, the seller almost always selects us. So, if we value the good at $v > \mu$, the total utility earned by the clairvoyant algorithm after T rounds is asymptotically $(v - \mu)T$. Our objective is to get a total utility close to this quantity without knowing μ.

The loss of any pricing algorithm can be decomposed into two components: number of times we are not selected by the seller when we employ that algorithm, and the "extra" payment

[6]Note that we are making the assumption that v is drawn each time independently of other draws of v or other random variables in the model. In particular, v has to be independent of the price of the outside option. This assumption is realistic when the set of goods that are offered for sale are homogeneous, e.g., ad slots on a single web page on the publisher's website.

(i.e., amount of payment over μ) we pay the seller during the rounds we are selected. More formally, let's define

$$\textbf{not-selected} = \sum_{t=1}^{T} \mathbf{1}[X_t = B],$$

$$\textbf{extra-payment} = \sum_{t=1}^{T} \mathbf{1}[X_t = A](p_t^A - \mu).$$

The expected *regret* of the algorithm can be written as:

$$\textbf{not-selected} \cdot (v - \mu) + \textbf{extra-payment}. \tag{4.2}$$

Our objective is to set the prices p_t^A in such a way that both terms in the above expression are sublinear ($o(T)$). Our main result is an algorithm that achieves a bound of $\tilde{O}(T^{\frac{1+\gamma}{2}})$ for these regret terms, where $\gamma < 1$ is the exponent in the regret bound (4.1) of the seller.

4.1.2. Our Pricing Algorithm

The idea behind our algorithm is simple: note that if we offer a constant price, the lowest price at which the seller still chooses us over the outside option without incurring linear regret is μ. We run a binary search to estimate this value. The subtlety here is that since the seller does not see the prices and is allowed some regret, we need to repeat offering the same price a number of times to accurately decide whether the price is too high or too low. Furthermore, if the price we offer is too close to μ, the seller can essentially choose arbitrarily without violating the regret bound. Therefore, the binary search will need to allow for some margin of error.

For simplicity, we assume the total number of rounds T is known, and we prove that at the end of the T rounds, our regret is bounded. Our proposed algorithm is described in Algorithm 6. The algorithm uses the function $f(k)$ and constant θ that will be fixed during the analysis. Also the variable t in the algorithm is only for bookkeeping purposes.

ALGORITHM 6: Binary Search Pricing Algorithm

$l_0 \leftarrow 0, u_0 \leftarrow 1$, $k \leftarrow 0$, $t \leftarrow 0$;
while $u_k - l_k > T^{-\theta}$ **do**
  $p_k \leftarrow (l_k + u_k)/2$;
  Offer the seller a price of p_k for $f(k)$ rounds;
  $x \leftarrow$ # of times the seller accepts the price of p_k;
  $l_{k+1} \leftarrow l_k$, $u_{k+1} \leftarrow u_k$;
  if $x > f(k)/2$ **then**
    $l_{k+1} = (2l_k + u_k)/3$;
  else
    $u_{k+1} = (l_k + 2u_k)/3$;
  end
  $t \leftarrow t + f(k)$;
  $k \leftarrow k + 1$;
end
Offer a price of $u_k + T^{-\theta}$ for the remaining rounds;

4.1.3. A No-regret Guarantee

The main result of this section is the following:

Theorem 16 *Consider a run of Algorithm 6 for T steps, and assume the seller follows a strategy that satisfies the regret bound* (4.1)*. Then, with probability at least $1 - O(\frac{\log T}{T})$, both the number of times we are not selected by the seller and the extra payment to the seller are bounded by*

$$O\left(T^{\frac{1+\gamma}{2}} \log T\right).$$

Proof We start with a few notations. We call the steps during the binary search while loop (lines 2–14 of Algorithm 6) the *exploration phase*, and the steps after this loop (line 6) the *exploitation phase*. The k'th iteration of the exploration while loop (with k starting from 0) is called the *k'th exploration phase*, or simply *phase k*.

Since the length of the interval $u_k - l_k$ decreases by a factor of $2/3$ in each phase, the number of phases of the algorithm is at most $O(\log T)$. Therefore, using the regret bound (4.1) with $\delta = 1/T$ and the union bound, we know that with probability at least $1 - O(\frac{\log T}{T})$, at the

end of every phase (both exploration phases and the exploitation phase), we have

$$R(t) < ct^{\gamma} \log(T). \tag{4.3}$$

Throughout the rest of the proof, we assume the above event happens, and prove that the desired bounds on the regret of our algorithm follow from this.

The argument is in two steps. First, we show that if the function $f(k)$ is properly chosen, with high probability, the algorithm maintains the invariant that the value of μ lies in the interval $[l_k, u_k]$. In particular, this means that at the end of the exploration phases, the value of μ is at most u_k and is at least $l_k \geq u_k - T^{-\theta}$. This implies that in each of the steps in the exploitation phase, either the seller gets an expected regret of at least $T^{-\theta}$ by not accepting the price of $u_k + T^{-\theta}$, or she accepts and we make an extra payment that is at most $2T^{-\theta}$. The second step is to use this fact to bound the total regret of the algorithm.

We prove the invariant $\mu \in [l_k, u_k]$ by induction. Consider a phase k, and assume $\mu \in [l_k, u_k]$. We show that the probability that this property does not hold in the subsequent phase is small. To do this, we bound the regret of the seller in this phase, and show that if the algorithm makes the wrong decision about l_{k+1} or u_{k+1} in this phase, seller's regret must be too high.

First, consider the case that $\mu > (l_k + 2u_k)/3$. We show that in this case, with high probability the seller accepts the price p_k less than $f(k)/2$ times. Let x denote the number of times that the seller accepts the price p_k during this phase. Note that x is a random variable and can depend on the draws of the price of the outside option as well as the internal random bits of the seller's algorithm. We compare the expected total price the seller pays during phases 0 through k with the expected total price she would have gotten had she always picked the outside option. The latter value is simply $\sum_{i=1}^{k} f(i)\mu$. The total price the seller gets during phase k can be computed as follows: In x steps during this phase, the seller gets a price of p_k. In each of the remaining $(f(k) - x)$ steps, the seller gets a price that

is drawn from a distribution with mean μ. We define a martingale $0 = Y_0, Y_1, Y_2, \ldots, Y_{f(k)}$ based on this process as follows: For each i, if the seller selects us in step i of phase k, we let $Y_i = Y_{i-1}$. Otherwise, we let Y_i be Y_{i-1} plus the price of the outside option in step i minus μ. Note that this is in fact a martingale. The total price of the outside option during this phase is precisely $Y_{f(k)} + (f(k) - x)\mu$. Therefore, the total price that the seller receives during this phase is

$$xp_k + (f(k) - x)\mu + Y_{f(k)} \leq f(k)\mu - x \cdot \frac{u_k - l_k}{6} + Y_{f(k)}.$$

For each step in phase i ($0 \leq i \leq k-1$), the expected price the seller gets is at most $\max(\mu, p_i)$. Therefore, the expected total price during these phases is at most

$$\begin{aligned}
\sum_{i=0}^{k-1} f(i) \max(\mu, p_i) &= \sum_{i=0}^{k-1} f(i)\mu + \sum_{i=0}^{k-1} f(i) \max(0, \mu - p_i) \\
&\geq \sum_{i=0}^{k-1} f(i)\mu + \sum_{i=0}^{k-1} f(i) \cdot \frac{u_i - l_i}{2}
\end{aligned}$$

Therefore, the difference between the total price the seller gets and the price she would have gotten had she always picked the outside option is at least

$$x \cdot \frac{u_k - l_k}{6} - \sum_{i=0}^{k-1} f(i) \cdot \frac{u_i - l_i}{2} - Y_{f(k)}$$

The value of $u_i - l_i$ decreases by a factor of $2/3$ in each phase. Therefore, if $x > f(k)/2$, the regret of the seller is at least:

$$\text{Regret} \geq \frac{1}{12}(2/3)^k f(k) - \frac{1}{2}\sum_{i=0}^{k-1}(2/3)^i f(i) - Y_{f(k)}. \tag{4.4}$$

This means that if we select $f(k)$ in such a way that the above value is more than the regret bound (4.1), the above event cannot happen, and therefore, the algorithm makes the

right choice and maintains the property that $\mu \in [l_k, u_k]$.

First, we use martingale inequalities to bound the term $Y_{f(k)}$. Using Azuma's inequality and the fact that prices are bounded by 1, the probability that $Y_{f(k)} > \epsilon(2/3)^k f(k)$ is at most $2\exp(-O(\epsilon^2(2/3)^{2k}f(k)))$. In this case, the regret of the seller is at least $(\frac{1}{12} - \epsilon)(2/3)^k f(k) - \frac{1}{2}\sum_{i=0}^{k-1}(2/3)^i f(i)$. We need to set $f(k)$ in such a way that this value is larger than the regret bound of the seller.

Assume $f(k)$ is of the form $f(k) = \alpha\beta^k$ for values $\alpha > 0$ and $\beta > 1$ that will be fixed later. The lower bound (4.4) on the regret of the seller can be written as

$$\begin{aligned} \text{Regret} &\geq \frac{\alpha}{12}(1-\epsilon)(\frac{2\beta}{3})^k - \frac{\alpha}{2}\sum_{i=0}^{k-1}(\frac{2\beta}{3})^i \\ &= \frac{\alpha}{12}(1-\epsilon)(\frac{2\beta}{3})^k - \frac{\alpha}{2}\cdot\frac{(\frac{2\beta}{3})^k - 1}{\frac{2\beta}{3} - 1}. \end{aligned} \tag{4.5}$$

On the other hand, since the value of t at the end of the k'th phase is $\sum_{i=0}^{k} f(i)$, the upper bound (4.3) on the regret can be written as

$$\begin{aligned} \text{Regret} &< c\log(T)\left(\sum_{i=0}^{k} f(i)\right)^{\gamma} \\ &= c\alpha^{\gamma}\log(T)\left(\frac{\beta^k - 1}{\beta - 1}\right)^{\gamma}. \end{aligned} \tag{4.6}$$

If we pick $\alpha = \left(\frac{c\log(T)}{\lambda}\right)^{\frac{1}{1-\gamma}}$ for another constant λ that will be fixed later, we would have

$$c\alpha^{\gamma}\log(T) = \lambda\left(\frac{c\log(T)}{\lambda}\right)^{1+\frac{\gamma}{1-\gamma}} = \lambda\alpha.$$

Therefore, after combining lower and upper bounds (4.5) and (4.6), we can cancel α from both sides of the inequality and obtain:

$$\frac{1-\epsilon}{12}(\frac{2\beta}{3})^k - \frac{1}{2}\cdot\frac{(\frac{2\beta}{3})^k - 1}{\frac{2\beta}{3} - 1} < \lambda\left(\frac{\beta^k - 1}{\beta - 1}\right)^{\gamma}$$

Assuming $\beta > \frac{3}{2}$, the above inequality implies

$$\left(\frac{1-\epsilon}{12} - \frac{1}{2(\frac{2\beta}{3}-1)}\right)(\frac{2\beta}{3})^k < \lambda\frac{\beta^{\gamma k}}{(\beta-1)^\gamma} \tag{4.7}$$

We now fix the value of β to $\beta = (\frac{3}{2})^{\frac{1}{1-\gamma}}$. Note that this value satisfies the assumption $\beta > 3/2$. We have:

$$\frac{2\beta}{3} = (\frac{3}{2})^{\frac{1}{1-\gamma}-1} = \beta^\gamma.$$

Therefore, inequality (4.7) reduces to

$$\lambda > \left(\frac{1-\epsilon}{12} - \frac{1}{2(\frac{2\beta}{3}-1)}\right)(\beta-1)^\gamma.$$

This means that if we pick the value of λ to be the expression on the right-hand side of the above inequality, inequality (4.7) leads to a contradiction. Thus, with probability at least $1 - O(\frac{\log T}{T}) - 2\sum_k \exp(-O(\epsilon^2(2/3)^{2k}f(k)))$, the event "$\mu > (l_k + 2u_k)/3$ but $x > f(k)/2$" does not happen in any phase k. An almost identical proof shows that the event "$\mu < (2l_k + u_k)/3$ but $x < f(k)/2$" does not happen in these cases either. If these events happen, the algorithm maintains the invariant that $\mu \in [l_k, u_k]$ throughout the exploration steps. The probability that this is violated is at most

$$O(\frac{\log T}{T}) + 2\sum_k \exp(-O(\epsilon^2\alpha(\frac{4\beta}{9})^k)).$$

It is not hard to see that with the above choice of the values of α and β, the above expression tends to zero as T tends to infinity.

Given this invariant, in each of the steps in the exploitation phase (line 6), either the seller incurs a regret of at least $T^{-\theta}$ by not accepting the price of $u_k + T^{-\theta}$, or she accepts and we get a regret of at most $2T^{-\theta}$. Let y denote the number of times we are not selected by the seller during the exploitation phase. We bound the total regret of the seller compared to the strategy that always selects us using a method similar to the first part of the proof.

Since $\mu \in [l_i, u_i]$ for every i, in each step during phase i, the price of the option selected by the seller is at most u_i, i.e., at most $\frac{u_i - l_i}{2} = \frac{1}{2}(2/3)^i$ higher than our price . In each of the y steps that the seller chooses the outside option during the exploitation phase, her regret is at least $T^{-\theta}$. Therefore, the total regret of the seller is at least

$$yT^{-\theta} - \frac{1}{2}\sum_{i=0}^{k^*-1}(2/3)^i f(i),$$

where k^* is the value of k at the end of the algorithm. Using the regret bound for the seller at the end of the T steps, we get the following inequality:

$$yT^{-\theta} - \frac{1}{2}\sum_{i=0}^{k^*-1}(2/3)^i f(i) < c\log(T)T^{\gamma}.$$

Replacing $f(i) = \alpha\beta^i$, we obtain:

$$yT^{-\theta} < \frac{\alpha}{2}\left(\frac{2\beta}{3} - 1\right)^{-1}(\frac{2\beta}{3})^{k^*} + c\log(T)T^{\gamma}$$

Since $u_k - l_k = (2/3)^k$, we have $k^* = \log(T^{-\theta})/\log(2/3)$. Therefore,

$$(\frac{2\beta}{3})^{k^*} = (\frac{3}{2})^{\frac{\gamma k^*}{1-\gamma}} = T^{\frac{\theta\gamma}{1-\gamma}}$$

Therefore,

$$y < \frac{\alpha}{2}\left(\frac{2\beta}{3} - 1\right)^{-1} T^{\theta + \frac{\theta\gamma}{1-\gamma}} + c\log(T)T^{\gamma+\theta}$$

Furthermore, the total length of the exploration phases is $\alpha\sum_{i=0}^{k^*-1}\beta^i < \frac{\alpha}{\beta-1}T^{\frac{\theta}{1-\gamma}}$. Therefore, even assuming that the seller never chooses us during the exploration phase, the total number of times the seller does not chose us can be written as

$$\frac{\alpha}{\beta-1}T^{\frac{\theta}{1-\gamma}} + \frac{\alpha}{2}\left(\frac{2\beta}{3} - 1\right)^{-1} T^{\frac{\theta}{1-\gamma}} + c\log(T)T^{\gamma+\theta}.$$

Since β is a constant and $\alpha = O((\log T)^{1/(1-\gamma)})$, the above expression is at most

$$O(\log(T) T^{\max(\frac{\theta}{1-\gamma}, \gamma+\theta)}). \tag{4.8}$$

Finally, we bound the amount of extra payment (i.e., payment beyond μ) made to the seller. By the invariant $\mu \in [l_i, u_i]$, we know that in each round in the i'th exploration phase, this extra payment is at most $\frac{1}{2}(u_i - l_i)$. Also, during the exploitation phase, the extra payment is at most $2T^{-\theta}$ per round. Therefore, the total extra payment made to the seller can be bounded by

$$\frac{1}{2} \sum_{i=0}^{k^*-1} (2/3)^i f(i) + 2T^{-\theta} \cdot T = O(\alpha T^{\frac{\theta\gamma}{1-\gamma}} + T^{1-\theta}). \tag{4.9}$$

Now, if we select $\theta = \frac{1-\gamma}{2}$, both expressions (4.8) and (4.9) will be at most $O(\log(T) T^{\frac{1+\gamma}{2}})$.

■

Here, we discuss some of the assumptions we made in our model. In particular, we sketch how the assumptions that the number of rounds T is known and that μ should be in $[0, 1]$ can be relaxed. We also show that the assumption that the outside option is stochastic is necessary.

Unknown number of rounds The assumption that the number of rounds T is known can be relaxed using a standard "doubling" trick. The main observation is that Theorem 16 holds even if the number of rounds turns out to be not precisely T but a constant multiple of T. Therefore, we can start running the algorithm with a small value of T as an estimate for the number of rounds, and each time we discover that the actual number of rounds is more than the current estimate, we multiply the estimate by a constant and restart the algorithm from scratch. It is not hard to show that this algorithm satisfies the same regret bounds (with larger constants hidden in the $O(\cdot)$ notation).

Range of μ The assumption that the mean μ of the outside is between 0 and 1 can be relaxed by adding an initial "doubling" stage to the binary search algorithm to find an upper bound M on μ. A term containing the value the upper bound M will be added to the regret of the algorithm.

Arbitrary buyer values If our value for the good offered by the seller is v, the expression (4.2) gives the value of our regret, *assuming* $v > \mu$. This assumption can be relaxed with a simple modification of Algorithm 6 that caps the offered price at v. The proof is straightforward and is omitted due to space constraints.

Non-stochastic outside option Since we offer prices based on the observed behavior of the seller, it is reasonable to ask why we assume that the prices offered by the outside option are drawn i.i.d. from a fixed distribution $\mathcal{D}$. Consider an alternate model where the outside option can offer arbitrary prices, and the goal is for our expected total utility to asymptotically approach $(v - \mu_T) \cdot T$, where $\mu_T = E\left[\frac{1}{T}\sum_{t=1}^{T} p_t^B\right]$. Unfortunately, allowing the outside option this much flexibility makes our goal impossible.

To see this, consider an outside option that simulates our algorithm and offers identical prices, so that the distribution of p_t^A and p_t^B are the same (note that the outside option can also observe the seller's behavior, so this simulation is feasible). Clearly one way for the seller to ensure that her regret $R(T) = 0$ is to select between us and the outside option via an independent coin toss in every round. However, in this case our expected total utility will be

$$\begin{aligned} E\left[\sum_{t=1}^{T} \mathbf{1}[X_t = A] \cdot (v - p_t^A)\right] &= \sum_{t=1}^{T} \frac{1}{2} \cdot E[(v - p_t^A)] \\ &= \sum_{t=1}^{T} \frac{1}{2} \cdot E[(v - p_t^B)] \\ &= \frac{1}{2}(v - \mu_T) \cdot T, \end{aligned}$$

and thus the difference between $(v - \mu_T) \cdot T$ and our total utility is linear in T, disallowing

the possibility that any pricing algorithm can have low regret. One potential approach to get around this impossibility result is to assume more information about the particular no-regret algorithm the seller is using. We leave the analysis of this alternative model as an interesting direction for future work.

4.1.4. Simulations

In this section we empirically evaluate the performance of Algorithm 6 and 7, and compare them with a baseline.

A Heuristic Algorithm The idea behind Algorithm 6 was to zero in on the smallest price the seller is willing to sell her goods for. To do this, we maintained the invariant that the target price is always within a shrinking interval around the price we offered. Maintaining this invariant made it possible to theoretically analyze the regret of the algorithm: we could use a simple union bound to handle the highly-correlated error events, and get around the complexity arising from the sequential stochastic nature of the errors. This invariant, however, came at a cost: we needed to offer the same price many times to ensure that the average response of the seller gives us a reliable signal about the target price, and make the decision about the next step based on this reliable signal. An alternative approach is to forgo the invariant, and adjust the price based on signals that are unreliable on their own right, but stochastically lead us in the right direction. This is what Algorithm 7 does.

There are a few subtleties in the process of updating prices in Algorithm 7: To ensure that the prices eventually get closer to the target price we need to update them in a way that the changes become smaller and smaller as time goes on. To do this, we update the prices by multiplying or dividing the current price by a time-dependent factor. Note that to ensure our price remains above the target price significantly more often than below it, we need to use different factors for multiplication and division. So every time the price is rejected we multiply it by a factor of $(1 + t^{-\alpha})$ for some $0 < \alpha < 1$, and when it is accepted, we divide it by a *smaller* factor $(1 + t^{-\beta})$ (i.e., $\beta > \alpha$). Aside from this, we leave it to the simulation

to determine the best values for the parameters α and β.

ALGORITHM 7: Heuristic Pricing Algorithm

$t \leftarrow 0$, $p_t \leftarrow \frac{1}{2}$;
while *true* **do**
  Offer the seller a price of p_t ;
  if *the seller rejects* **then**
    $p_{t+1} = (1 + t^{-\alpha})p_t$;
  else
    $p_{t+1} = (1 + t^{-\beta})^{-1}p_t$;
  end
  $t \leftarrow t + 1$;
end

While Algorithm 7 is simple and natural, and as we will see in Section 4.1.4 performs well in practice, the fact that the sequence of errors it generates is correlated makes it difficult to analyze its performance theoretically. In the next section we evaluate the performance of the algorithm via simulations, and leave its theoretical analysis for future work.

Baseline We compare our algorithms with a naive baseline that works as follows: Given parameters $0 < \epsilon < 1$, it discretizes the price space (i.e. [0,1]) into $\frac{1}{\epsilon}$ equally spaced prices and treats each of these prices as an arm. When the algorithm offers the price p_i the seller, the reward from the corresponding arm is equal to p_i if the seller chooses our buyer, and is 0 otherwise. The baseline simply runs the algorithm EXP3.P (see (Bubeck and Cesa-Bianchi, 2012)). Note that from a theoretical stand-point we don't expect this algorithm to perform well for the following reason: Given that the seller is playing a no-regret algorithm, in order for us to observe her eventual reaction to a particular price, we need to offer the same price to them multiple times, i.e. long enough for their no-regret algorithm to realize the price change and respond to it. The baseline fails to do this, and as the result we expect its regret to be high.

Simulation setup The simulation setup is as follows: we assume the price p_t^B of the outside option comes from a uniform distribution on $[0, 2\mu]$ where $\mu = 0.3$. For this and other parameters, we experimented with other values as well and did not observe any significant

Table 1: Regret values after $T = 10^6$ steps

Algorithm	not selected	extra payment	regret
Algorithm 1	61110	32040	74817
Algorithm 2	8585	9227	15236
Baseline	840149	-908	587196

difference in the outcome. For the seller, we use the algorithm EXP3.P (see (Bubeck and Cesa-Bianchi, 2012)). We take $T = 10^6$ and run both Algorithms 6, 7 with a range of values for their free parameters (i.e. the function f and the value θ for Algorithm 6, and the values α and β for Algorithm 7). We track the number of rounds our exchange is not selected by the seller, the extra payment to the seller, and the overall regret. For the baseline $\epsilon = 0.001$. The regret values reported here use a value of $v = 1$ in the regret expression (4.2). Each simulation is repeated 100 times, and the computed values are averaged over these runs. Confidence intervals are very small, hence omitted for better readability.

Optimal setting of the parameters For Algorithm 6, we use the functional form $f(k) = a \cdot \log(T)^2 \beta^k$ (see Section 4.1.3). A grid search over the ranges $a \in [0.5, 2.5], \beta \in [1, 2.5]$, and $\theta \in [0.1, 0.3]$ reveals that the values $a = 2$, $\beta = 1.5$, and $\theta = 0.2$ result in the lowest regret. Observe that the values of β and θ are close to the values derived in the analysis. For Algorithm 7, a grid search over the range $0 < \alpha < \beta \leq 1$ finds that the combination $\alpha = 0.1$ and $\beta = 0.5$ results in the lowest regret.

Comparison of the algorithms In Table 1 we present the following quantities for each algorithm: The number of times the price is not accepted by the seller, the extra payment to the seller, and the overall regret. Figure 4.1.4 illustrates the total regret of each algorithm as a function of time in the logarithmic scale. One can see that Algorithms 6 and 7 both significantly outperform the baseline in terms of the total regret. Furthermore, the regret of Algorithms 6 and 7 are sublinear, while that of the baseline is growing linearly with time. Also, interestingly algorithm 7 incurs less regret than Algorithm 6.

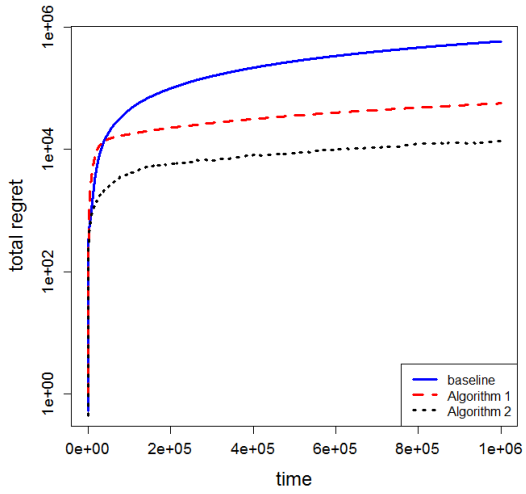


Figure 16: Regret of Algorithms 6, 7, and the baseline as a function of time.

4.1.5. Discussion and Future Directions

We presented a binary search-style pricing algorithm for a buyer facing a no-regret seller. Our main contribution was the analysis of this algorithm and showing that it guarantees the buyer vanishing regret. It remains an open question whether the regret bound presented here is asymptotically tight. Furthermore, we focused on the *buyer* side of the market only and ignored the possibility of the seller responding strategically to our proposed algorithm. We leave the equilibrium analysis and the study of the seller-side implications of the algorithm for future work.

4.2. Competing via Viral Marketing

In the traditional viral marketing problem, firms attempt to maximize the adoption of their product or service in a large social network. To this end, each of them *seeds* a set of initial "infections" in the network via product give-aways, marketing campaigns targeting the seed individuals, and so on. The product may then spread through the network via local stochastic dynamics accounting for local recommendations or influence between neighbors, known as "word of mouth" effects. Previous papers on this subject mainly focus on designing (Kempe et al., 2003, 2005; Mossel and Roch, 2010) or improving (Chen et al., 2009, 2010;

Model \ Measure	Price of Anarchy	Budget Multiplier
Goyal and Kearns model	$\Theta(1)$	$\Theta(1)$
The Connectivity model	$\Theta(K_{max}^2, d_{max})$	$\Theta(K_{max})$
The Endogenous model	unbounded	unbounded

Table 2: Summary of our upper bound results compared to the work of Goyal and Kearns (2012) for the case where the adoption dynamics exhibit decreasing returns to local adoption.

Borgs et al., 2014; Goyal et al., 2011) algorithms for finding a seed set that (approximately) maximizes the total number of vertices that ultimately adopt the product. More recently, a number of papers (Goyal and Kearns, 2012; Bharathi et al., 2007; Borodin et al., 2010; Clark and Poovendran, 2011; Carnes et al., 2007; Dubey et al., 2006; Vetta, 2002; Borodin et al., 2017; Tzoumas et al., 2012; Alon et al., 2010) have examined models of *competitive contagion* that take a game-theoretic perspective on the traditional problem: two or more players or firms compete in a large social network, each with the goal of maximizing their individual market share, possibly at the expense of others.

In this paper, we introduce and examine two new natural models for competitive contagion in a network. Existing models of influence maximization and competitive contagion assume that firms benefit merely according to the *number* of nodes that eventually adopt their product. We introduce a framework where the payoffs to firms capture both their market share and the *connectivity* within that market share in the network. In many natural settings, the goal is not simply that many nodes adopt a product, but also subsequently use a networked service requiring the product. For instance, Skype users are more valuable if they are in a densely connected subnetwork of other Skype users with whom they use the service to interact. We thus consider utility functions that combine both the size of the market share and the connectivity within that share. For a broad family of stochastic dynamics— concave switching function and linear selection function— we prove upper bounds on both the pure strategy PoA and the Budget Multiplier which depend on the firm budgets and the maximum degree of the network, but no other structural properties. We also find broad conditions under which the PoA and the Budget Multiplier can be unbounded.

Previous works on the subject of influence maximization (e.g. see (Goyal and Kearns, 2012;

Kempe et al., 2003)) assume that the budgets available to firms to seed initial infections in the network are fixed and exogenously determined. In many settings this might not be realistic, as firms are free to adjust their budgets in order to capture a larger market share. We therefore also consider a model in which budgeting decisions are *endogenous*. While the results of Goyal and Kearns (2012) establish fairly general conditions on local dynamics yielding bounded PoA and Budget Multiplier independent of network structure, we show that such robustness vanishes in the case of endogenous budgets: for almost *any* choice of dynamics, the PoA and the Budget Multiplier may be unbounded for certain network structures. The informal intuition is that firms may engage in "bidding wars" for sub-optimal seed infections that essentially eradicate subsequent market share gains. Table 2 shows a summary of our results compared to (Goyal and Kearns, 2012)[7].

Related Work

We contribute to the study of influence maximization in a networked setting (see (Kempe et al., 2003, 2005; Mossel and Roch, 2010; Chen et al., 2009, 2010; Borgs et al., 2014; Goyal et al., 2011)), where the goal is to find a small set of influential nodes in the network whose aggregated influence is maximized. We also contribute to the study of competition in networked environments (see (Gerard, 1977; Grossman and Shapiro, 1984)). We mainly build on the game-theoretic framework introduced by Goyal and Kearns (2012) for studying the competitive influence maximization in a social network. In this work, the authors identify broad conditions on the adoption dynamics — namely, decreasing returns to local adoption — under which the PoA is uniformly bounded above, across all networks. They also find sufficient conditions on the adoption dynamics — namely, proportional local adoption between competitors — for bounded pure strategy Budget Multiplier. In our work we investigate similar problems in more general settings.

To the best of our knowledge, our work is the first to take the connectivity among adopters

[7]In Table 2, K_{max} denotes the maximum number of "seeds" firms can spend to initialize the adoption of their product in the network, and d_{max} is the maximum degree of a vertex in the network.

into account in the payoff function of players. There are however several previous papers that look at a similar quantity, but with different goals. For example, Quan et al. (2012) and Chaoji et al. (2012) investigate the relationship between connection features of individuals and the popularity of a content in a social network; and Katona et al. (2011) investigate the effect of the connectivity among current adopters on a potential consumer's behavior. Our connectivity model is also remotely related to the large body of work on clustering (see (Aggarwal and Wang, 2010) for a survey).

The impact of endogenous budgets have been investigated on various economic problems, including auctions (Kotowski, 2013; Burkett, 2015; Ausubel et al., 2013) and housing markets (Pereyra, 2012).

4.2.1. Model and Preliminaries

For the underlying diffusion dynamics, we consider the switching-selection framework introduced by Goyal and Kearns (2012). Before introducing our new models, we first review this framework.

Consider a 2-player [8] game of competitive adoption on a (possibly directed) graph G over n vertices. G is known to the players, R(ed) and B(lue). The two players simultaneously choose some number of vertices to initially seed; after this seeding, the stochastic dynamics of local adoption determines how each player's seeds spread throughout G to create adoptions by new nodes. Each player seeks to maximize her payoff which is a function of her eventual adopters.

More precisely, suppose that player $p \in \{R, B\}$ has $K_p \in \mathbb{N}$ seed infections; Each player p then chooses an allocation $a_p = (a_{p1}, a_{p2}, ..., a_{pn})$ of budget across the n vertices, where $a_{pj} \in \mathbb{N}$ and $\sum_{j=1}^{n} a_{pj} = K_p$. Goyal and Kearns (2012) assume that K_p is exogenously

[8] One can also define the single-player setting by letting the opponent's strategy be the empty set. In the single-player setting the interesting problem is that of maximizing the profit. One can easily show that as is the case for other contagion models, including (Kempe et al., 2003), profit maximization is hard in our models. In fact it may even be more difficult, since in the connectivity model the payoff function is not submodular, and in the endogenous model it is not monotone.

given.We will see that this assumption is crucial for obtaining their upper bounds on the Price of Anarchy and the Budget Multiplier.

Let A_p be the set of allocations for player p, which is her pure strategy space. A mixed strategy for player p is a probability distribution σ_p on A_p. Let $\mathcal{A}_p$ denote the set of probability distributions for player p. The two players simultaneously choose their strategies (σ_R, σ_B). Consider any realized initial allocation (a_R, a_B) for the two players. Let $V(a_R) = \{v|a_{Rv} > 0\}$, $V(a_B) = \{v|a_{Bv} > 0\}$ and let $V(a_R, a_B) = V(a_R) \cup V(a_B)$. A vertex v becomes initially infected if one or more players assigns a seed to infect v. If both players assign seeds to the same vertex, then the probability of initial infection by a player is proportional to the seeds allocated by the player (relative to the other player).

Following the allocation of seeds, the stochastic contagion process on G determines how these Red and Blue infections generate new adoptions in the network. Time is considered to be discrete for this process, and the state of a vertex v at time t is denoted $s_{vt} \in \{U, R, B\}$, where U stands for Uninfected, R stands for infection by Red, and B stands for infection by Blue. We assume there is an *update schedule* which determines the order in which vertices are considered for state updates. Note that this schedule does not need to be deterministic. We also assume once a vertex is infected, it is never a candidate for updating again.

For the stochastic update of an uninfected vertex v, we will consider the *switching-selection* model. In this model, updating is determined by the application of two functions to v's local neighborhood: $f(x)$ (the *switching* function), and $g(y)$ (the *selection* function). More precisely, let α_R and α_B be the fraction of v's neighbors infected by R and B, respectively, at the time of the update, and let $\alpha = \alpha_R + \alpha_B$ be the total fraction of infected neighbors. The function f maps α to the interval $[0, 1]$ and g maps $\alpha_R/(\alpha_R + \alpha_B)$ (the relative fraction of infections that are R) to $[0, 1]$. These two functions determine the stochastic update in the following fashion:

1. With probability $f(\alpha)$, v becomes infected by *either* R or B; with probability $1 - f(\alpha)$,

v remains in state U(ninfected), and the update ends.

2. If it is determined that v becomes infected, it becomes infected by R with probability $g(\alpha_R/(\alpha_R+\alpha_B))$, and infected by B with probability $g(\alpha_B/(\alpha_R+\alpha_B))$.

We assume $f(0) = 0$ (infection requires exposure), $f(1) = 1$ (full neighborhood infection forces infection), and f is increasing (more exposure yields more infection); and $g(0) = 0$ (players need some local market share to win an infection), $g(1) = 1$. Note that since the selection step above requires that an infection takes place, we also have $g(y) + g(1-y) = 1$, which implies $g(1/2) = 1/2$.

Given a graph G and an initial allocation of seeds, the dynamics described above — determined by f, g, and the update schedule — yield a number of adopters for the two players, and as mentioned earlier, the payoff to player $p \in \{R, B\}$, which we denote by $\Pi_p(\sigma_R, \sigma_B)$, is a function of her eventual adopters. Here is one possible choice for the payoff function: let the random variable χ_p denote the *number* of infections p gets at the termination of the dynamics for the strategy profile (σ_R, σ_B); in (Goyal and Kearns, 2012) authors assume $\Pi_p(\sigma_R, \sigma_B) = \mathbb{E}[\chi_p|(\sigma_R, \sigma_B)]$, where the expectation is over any randomization in the player strategies, in the choice of initial allocations, and the randomization in the stochastic updating dynamics. Shortly in the connectivity and the endogenous budgets model we will see more general choices for the payoff function.

Given any payoff function, each player seeks to maximize her own expected payoff, and this results in competition among the players. In the resulting game a *Pure Nash Equilibrium* is a profile of pure strategies (a_R, a_B) such that a_p maximizes player p's payoff given the strategy a_{-p} of the other player.

A *Mixed Nash Equilibrium* is a pair $\sigma = (\sigma_R, \sigma_B)$ of independent probability distributions that satisfies

$$\mathbb{E}_{a\sim\sigma}[\Pi_p(a)] \geq \mathbb{E}_{a_{-p}\sim\sigma_{-p}}[\Pi_p(a'_p, a_{-p})]$$

for every p and $a'_p \in A_p$. In the above $a = (a_p, a_{-p})$.

For a fixed graph G and stochastic update dynamics, the *maximum social welfare* allocation is the (deterministic) allocation (a^*_R, a^*_B) (obeying the budget constraints if any exists) that maximizes $\Pi_R(a_R, a_B) + \Pi_B(a_R, a_B)$. For the same fixed graph and update dynamics, let (σ_R, σ_B) be the equilibrium strategies that *minimize* $\Pi_R(\sigma_R, \sigma_B) + \Pi_B(\sigma_R, \sigma_B)$ among all equilibria[9]. Then the *Price of Anarchy* (or PoA) is defined to be

$$\frac{\Pi_R(a^*_R, a^*_B) + \Pi_B(a^*_R, a^*_B)}{\Pi_R(\sigma_R, \sigma_B) + \Pi_B(\sigma_R, \sigma_B)}.$$

The Price of Anarchy is a measure of the inefficiency in resource use created due to decentralized, non-cooperative behavior by the two players.

We also study the *Budget Multiplier*, which measures the extent to which network structure and dynamics can amplify initial *resource* inequality across the players. For example, when we have external budget constraints K_R, K_B, with $K_R \geq K_B$, we define the Budget Multiplier as follows: for any fixed graph G and stochastic update dynamics, let (σ_R, σ_B) be the equilibrium that *maximizes* the ratio

$$\frac{\Pi_R(\sigma_R, \sigma_B)}{\Pi_B(\sigma_R, \sigma_B)} \times \frac{K_B}{K_R}$$

among all equilibria[10]. The resulting maximized ratio is the Budget Multiplier, and it measures the extent to which the larger budget player can obtain a final market share that exceeds her share of the initial budgets[11].

Finally, we will restate some of the results in (Goyal and Kearns, 2012) which we will make use of later in the paper.

[9] If we restrict attention to pure Nash equilibria only, the resulting ratio is called the *pure strategy* Price of Anarchy.

[10] If we restrict attention to pure Nash equilibria only, the resulting ratio is called the *pure strategy* Budget Multiplier.

[11] We will later introduce the endogenous budgets model in which budgets are not externally constrained. In that model we can either use the same definition for the Budget Multiplier, or use a similar one in which we replace the number of seeds players spend with the cost per seed for each.

Lemma 7 *Let a_R and a_B be any sets of seed vertices for the two players. Then if f is concave and g is linear,*

$$\mathbb{E}[\chi_R|(a_R, \emptyset)] \geq \mathbb{E}[\chi_R|(a_R, a_B)]$$

and

$$\mathbb{E}[\chi_B|(\emptyset, a_B)] \geq \mathbb{E}[\chi_B|(a_R, a_B)].$$

Lemma 8 *Let a_R and a_B be any sets of seeded nodes for the two players. If f is increasing,*

$$\mathbb{E}[\chi_R + \chi_B|(a_R, a_B)] \geq \mathbb{E}[\chi_R|(a_R, \emptyset)].$$

4.2.2. The Connectivity Model

In some cases, such as video conferencing and messaging applications like WhatsApp, ooVoo or Skype, not only the number of adopters matters; it is also important to the firms how well those adopters are connected to each other, as the use of these products takes place along the edges of the social network rather than at a vertex alone. Motivated by the above examples, we introduce the *connectivity* model in which a firm's goal is to maximize the number of her adopters *plus* the number of edges among those adopters; the second term here models the strength of the connectivity among the adopters.

More precisely, consider a *pure* strategy profile (a_R, a_B) where a_p denotes the strategy of player $p \in \{R, B\}$. We define the random variable γ_p to be the eventual number of edges among adopters of product p. Player p then seeks to maximize her payoff which is equal to

$$\mathbb{E}[\gamma_p + \chi_p|(a_R, a_B)].$$

Note that due to linearity of expectation $\mathbb{E}[\gamma_p+\chi_p|(a_R, a_B)] = \mathbb{E}[\gamma_p|(a_R, a_B)]+\mathbb{E}[\chi_p|(a_R, a_B)]$. To simplify the statement of our results we denote $\mathbb{E}[\gamma_p|(a_R, a_B)]$ by δ_p and $\mathbb{E}[\chi_p|(a_R, a_B)]$ by θ_p. In addition, we denote the strategy of player p in the maximum social welfare solution by a_p^* and her payoff by $\theta_p^* + \delta_p^*$. Also let $\theta = \theta_R + \theta_B$, $\delta = \delta_R + \delta_B$, $\theta^* = \theta_R^* + \theta_B^*$, and

$\delta^* = \delta^*_R + \delta^*_B$.

We will see that in this new model, when f is concave and g is linear, upper bounds on the pure PoA and Budget Multiplier still exist, but they can depend on the budget constraints and the structure of the network. In addition we will see that if f is convex and g is linear, then the PoA and Budget Multiplier can be unbounded. Note that the results and techniques presented in this section can be easily extended to the case where $\Pi_p = \mathcal{B}\delta_p + \mathcal{D}\theta_p$ with $\mathcal{B}, \mathcal{D}$ being positive constants. Detailed discussion is omitted due to space constraints.

We first illustrate the connectivity model with an example for which the equilibrium looks quite different compared to the original model of (Goyal and Kearns, 2012).

Example 2 *Consider a graph G consisting of 3 components C_1, C_2 and C_3, where C_1, C_2 are star networks of size $(N+1)$ with central nodes v_1, v_2 pointing to N peripheral vertices, and C_3 is a complete undirected network of size $3\sqrt{N}$. Suppose that both players have a budget equal to 1. Let $f(x) = x^\alpha$ for some $\alpha > 0$, and let g be linear. Suppose that the update schedule is relatively long so that any connected component containing a non-zero number of seeds eventually becomes entirely infected.*

It can be shown that in the original model, the equilibrium of the game on G is where Red and Blue put their seeds on v_1, v_2 and each get an expected number of infections equal to $(N+1)$. However, in the connectivity model, the equilibrium is where both players put their seeds on C_3 and each get an expected payoff approximately equal to $3(3N/2+\sqrt{N})/2$. Thus the connectivity objective causes the players to prefer to compete for the highly connected vertices, as opposed to each taking an isolated low-connectivity component.

In what follows, we will use the following notations: K_{max} denotes $\max\{K_R, K_B\}$, K_{min} denotes $\min\{K_R, K_B\}$, and d_{max} is the maximum degree of any vertex in the graph G.

Theorem 17 *If f is concave and g is linear, then in the connectivity model, the pure strategy Budget Multiplier is bounded from above by $8(K_{max}+1)$.*

Proof Without loss of generality suppose $K_R \geq K_B$. Let (S_R, S_B) denote the pure Nash equilibrium that maximizes the ratio $\frac{\Pi_R}{\Pi_B} \times \frac{K_B}{K_R}$. We assume that in this equilibrium, Red has the higher payoff, that is $\theta_R + \delta_R \geq \theta_B + \delta_B$ (if this is not the case the budget multiplier is bounded from above by 1). We will assume that $\delta_R \geq \theta_R$. Since if $\delta_R < \theta_R$, from Theorem 4 in (Goyal and Kearns, 2012), we know that in the equilibrium the Blue player gets at least $\frac{K_B}{2K_R}\theta_R$ infections (the Blue player can achieve this by imitating the K_B Red seeds that result in the highest *number* of infections). Therefore, we have

$$\Pi_B \geq \theta_B \geq \frac{K_B}{2K_R}\theta_R = \frac{K_B}{4K_R}(\theta_R + \theta_R) \geq \frac{K_B}{4K_R}(\theta_R + \delta_R)$$

and so we can conclude that the Budget Multiplier is at most 4.

Given the above assumptions, next we assign a unique label to each of the vertices in S_R and attribute subsequent Red infections to exactly one of these seeds. More precisely, let the solo Red process be the stochastic dynamical process on the network when only Red seeds S_R are present. Suppose $S_R = \{v_1, \ldots, v_{K_R}\}$. At time 0, label each vertex $v_i \in S_R$ by a different color C_i; also label all other vertices by U. In the subsequent steps, whenever a new vertex gets infected in the process we assign to it one of the K_R labels $\{C_i\}_{i=1}^{K_R}$ in the following manner: when updating a vertex v, we first compute the fraction α_v^R of neighbors whose current label is one of $C_1, \ldots, C_{K_R}$, and with probability $f(\alpha_v^R)$ we decide that an infection will occur (otherwise the label of v is updated to U). If an infection occurs, we simply choose an infected neighbor of v uniformly at random, and update v to have the same label (which will be one of the C_i's). It can easily be observed that at every step, the dynamics of the $(S_R, \emptyset)$ process are faithfully implemented if we drop label subscripts and simply view any label C_i as a generic Red infection R. Furthermore, at all times every infected vertex has only one of the labels C_i. Thus if we denote the expected number of edges with endpoints labeled C_i, C_j by $\Omega_{i,j}^R$, we have $\mathbb{E}[\gamma_R|(S_R, \emptyset)] = \sum_{i \leq j} \Omega_{i,j}^R$.

Next we claim that the blue player can choose K_B of the Red seeds as her strategy such that in expectation she gets at least $\frac{K_B}{4K_R(K_R+1)}$ share of the Red edges, which as we will

see, results in the desired bound on the Budget Multiplier.

To prove the above claim, observe that the Blue player can consider the $\frac{K_B}{2}$ color pairs with highest Ω^R values. If pair (i,j) is in that set, the Blue player adds both i and j to her seed set. Since there are $K_R(K_R+1)/2$ color pairs in total, the expected number of edges Blue gets by choosing this strategy, is at least $\frac{1}{4}\frac{K_B/2}{K_R(K_R+1)/2}\delta_R$ (the factor $\frac{1}{4}$ is present due to the fact that when Blue seeds v_i, v_j, the two vertices both become initially infected by Blue with probability 1/4). Therefore we have

$$\frac{\theta_R+\delta_R}{\theta_B+\delta_B} \leq \frac{2\delta_R}{\frac{K_B}{4K_R(K_R+1)}\delta_R} \leq 8(K_R+1)\frac{K_R}{K_B}$$

Next, we simply multiply the left and the the right hand side of the above inequality by $\frac{K_B}{K_R}$ and replace K_R with K_{max} to obtain the claimed bound on the Budget Multiplier. This finishes the proof. ∎

One can easily find families of examples in which the Budget Multiplier does actually grow with K_{max}.

Example 3 *Suppose $K_R = K > 2$ and $K_B = 2$. Consider the network G which is a complete graph of size K. One can easily see that in any equilibrium Red gets $\Theta(K)$ vertices while Blue gets $\Theta(1)$ vertices. Since all the Red vertices are connected to each other, the number of edges among them is $\Theta(K^2)$. Similarly the number of edges among Blue adopters is $\Theta(1)$. Therefore the Budget Multiplier is $\Theta(K)$.*

Theorem 18 *If f is concave and g is linear, then in the connectivity model, the pure strategy Price of Anarchy is bounded from above by $2+2(1+d_{max})(1+8(K_{max}+1)\frac{K_{max}}{K_{min}})$.*

Proof Let (S_R, S_B) denote the pure Nash equilibrium and (a_R^*, a_B^*) denote the maximum social welfare solution. Without loss of generality, suppose the Red player gets the higher payoff in (a_R^*, a_B^*). Since (S_R, S_B) is a Nash equilibrium, the deviation of Red player to a_R^* should not be profitable, i.e. $\theta_R+\delta_R$ must be larger than the payoff Red gets by deviating

to a_R^*. Let's denote that payoff by $(\theta'_R + \delta'_R)$. Next we find a lower bound on $(\theta'_R + \delta'_R)$. We claim the following holds:

$$\theta'_R + \delta'_R \geq (\theta_R^* - \theta) + (\delta_R^* - \theta d_{max}) \tag{4.10}$$

To prove the above, we first note that when Blue is not present, the number of infection R gets by switching to a_R^* is at least θ_R^* (Lemma 7). Also when Red is not present, the number of infection B gets by seeding S_B is at most θ (Lemma 8). Now by adding S_B to a_R^*, the total number of infections remains at least θ_R^*. In addition the number of Blue infections will decrease and become less than or equal to θ. So the number of Red infections will be at least $(\theta_R^* - \theta)$, i.e.

$$\theta'_R \geq (\theta_R^* - \theta) \tag{4.11}$$

Next, we observe that when Blue is not present, the number of edges Red gets using strategy a_R^* is at least δ_R^*. To see this, just note that by the departure of a_B^* from (a_R^*, a_B^*) all the vertices become just more likely to adopt Red. So the Red edges remain Red.

Now if the Blue player comes back with strategy S_B, this can result in at most θ Blue infections, and therefore number of Red edges decreases by at most θd_{max}, as each Blue vertex can take at most d_{max} edges away from Red, meaning that

$$\delta'_R \geq (\delta_R^* - \theta d_{max}). \tag{4.12}$$

Combining (4.11) and (4.12), we get

$$\theta'_R + \delta'_R \geq (\theta_R^* - \theta) + (\delta_R^* - \theta d_{max})$$

Combining the above with $\theta_R + \delta_R \geq \theta'_R + \delta'_R$ (which must hold because of the property of Nash equilibrium), we obtain the following:

$$\theta_R + \delta_R \geq (\theta_R^* - \theta) + (\delta_R^* - \theta d_{max}) \tag{4.13}$$

Now to prove the desired bound on PoA we can write

$$\begin{aligned}
& \theta_R + \delta_R \geq (\theta_R^* - \theta) + (\delta_R^* - \theta d_{max}) \\
\Rightarrow \quad & 1 + (1 + d_{max}) \frac{\theta}{\theta_R + \delta_R} \geq \frac{\theta_R^* + \delta_R^*}{\theta_R + \delta_R} \\
\Rightarrow \quad & 1 + (1 + d_{max}) \frac{\theta + \delta}{\theta_R + \delta_R} \geq \frac{\theta_R^* + \delta_R^*}{\theta + \delta} \\
\Rightarrow \quad & 2 + 2(1 + d_{max}) \frac{\theta + \delta}{\theta_R + \delta_R} \geq \frac{\theta^* + \delta^*}{\theta + \delta}. \qquad (4.14)
\end{aligned}$$

In the third line we used the fact that $\delta \geq 0$ and $\theta_B + \delta_B \geq 0$, and in the fourth line we used the fact that in the maximum social welfare solution the Red player gets the higher payoff, or equivalently $2(\theta_R^* + \delta_R^*) \geq \theta^* + \delta^*$. Next we note that

$$\frac{\theta + \delta}{\theta_R + \delta_R} \leq \begin{cases} 1 + 1 & \text{if } \theta_R + \delta_R \geq \theta_B + \delta_B \\ 1 + 8 & \text{if } K_R \geq K_B \\ 1 + \mathcal{M} \frac{K_B}{K_R} & \text{otherwise.} \end{cases}$$

where $\mathcal{M}$ is the Budget Multiplier[12]. Using the bound we obtained in Theorem 17 for Budget Multiplier, we get

$$\frac{\theta + \delta}{\theta_R + \delta_R} \leq 1 + 8(K_B + 1) \frac{K_B}{K_R}.$$

Combining the above with (4.14) and replacing K_B with K_{max} and K_R with K_{min}, we get

$$2 + 2(1 + d_{max})(1 + 8(K_{max} + 1) \tfrac{K_{max}}{K_{min}}) \geq \tfrac{\theta^* + \delta^*}{\theta + \delta}$$

and that finishes the proof. ∎

Finally, using constructions similar to the ones in (Goyal and Kearns, 2012), we show that the concavity of f and the linearity of g are necessary for obtaining the upper bounds.

Proposition 7 *If f is linear and $g(y) = y^s/(y^s + ((1-y)^s)$ for some $s > 1$, then in the*

[12]The discussion for the second item where $K_R \geq K_B$ is very similar to the proof of Theorem 17, and due to space constraints we do not repeat it here.

connectivity model, for any $V > 0$, there exists a graph G for which the Budget Multiplier is greater than V.

Proof (sketch) The idea, closely following Theorem 5 in (Goyal and Kearns, 2012), is to construct a layered graph that amplifies the punishment in the selection function, and as a result makes the Budget Multiplier arbitrarily large: The top layer of this amplifying graph is where in the equilibrium solution players put their seeds; as we go down the layers the share of vertices that adopt the product of the higher budget player, becomes larger and larger; and therefore in the last layer, which is a huge one that makes up for the majority of the payoff each player receives, the larger budget player receives a payoff much higher than what the opponent receives. As a result the Budget Multiplier becomes very large. By adjusting the parameters of this construction, we can make the Budget Multiplier arbitrarily large. ∎

Proposition 8 *If $f(x) = x^r$ for some $r > 1$, and g is linear, then in the connectivity model, for any $V > 0$, there exists a graph G for which the Price of Anarchy is greater than V.*

Proof (sketch) The idea, closely following Theorem 2 in (Goyal and Kearns, 2012), is to create a layered directed graph whose dynamics rapidly amplify the convexity of f. When we take two such amplification components of differing sizes, one equilibrium is the case where players coordinate on the smaller component, while the maximum social welfare solution lies in the larger component. As a result we have an example in which the PoA is large. By adjusting the parameters of this construction, we can make the PoA arbitrarily large. ∎

4.2.3. The Endogenous Budgets Model

As we have mentioned, previous works assume that there are external constraints on the maximum number of seeds players can spend to initialize the adoption of their product in the network. We argue that this assumption is not necessarily realistic, since in some settings (such as the case of product give-aways), if firms feel they can offset higher marketing

expenditures by winning greater market share, they are free to do so. This motivates us to relax this assumption and investigate the case in which firms are allowed to choose the number of seeds they want to allocate, given the constraint that each seed has a cost associated with it.

More precisely, we propose the *endogenous budgets model*, which is the following modification of the original framework: Each firm $p \in \{R, B\}$ has a cost per seed denoted by $c_p \geq 0$, and for each *new* (non-seed) vertex that adopts her product, firm p benefits b_p[13]. So if θ_p denotes the (expected) eventual number of non-seed infections that firm p obtains by initially spending K_p seeds, her payoff is equal to $b_p \times \theta_p - c_p \times K_p$. Without loss of generality we can assume $b_p = 1$, and write the payoff as:

$$\theta_p - c_p \times K_p$$

In this section we show that in the endogenous budgets model, for a broad setting of f, g and c, there are examples in which the PoA is unbounded[14].

Beside the PoA we also look at the quantity $\frac{\theta^*}{\theta}$ (recall that this quantity represented the PoA in the original model of (Goyal and Kearns, 2012)).We will see that similar to the PoA in our model, this quantity is also unbounded, showing that the unbounded PoA in the endogenous model is not merely due to the introduction of costs (i.e. the term $-c_p \times K_p$) to the payoff function.

We first illustrate the endogenous budgets model with an example where the equilibrium can look completely different compared to the original model of (Goyal and Kearns, 2012).

Example 4 *Consider a graph G consisting of a central node v pointing to N vertices $v_1, v_2, ..., v_N$, each of which points to 3 different (unimportant) vertices. Suppose $c_p =$*

[13]We assume seed vertices do not provide any payoff, only cost.

[14]Similar results hold for the Budget Multiplier. The construction—which is basically a properly tuned star-like network—is straightforward, and therefore omitted due to space constraints.

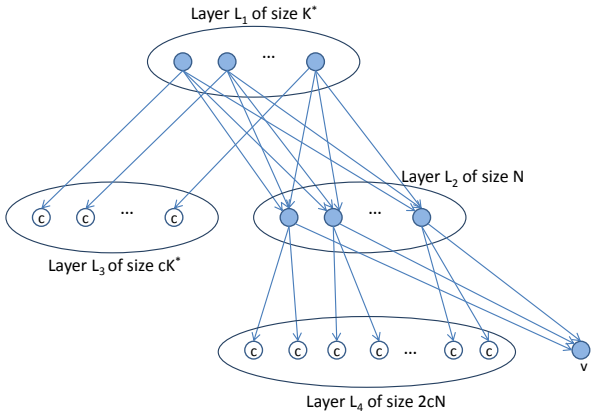


Figure 17: A graph with PoA $\geq (2c+1)N$.

$K_p = 1$ *for* $p \in \{R, B\}$. *Let* $f(x) = x^\alpha$ *for some* $\alpha > 0$, *and let* g *be linear. Suppose that the update schedule is relatively long so that a connected component containing a non-zero number of seeds eventually becomes entirely infected.*

It can be shown that in the original model of (Goyal and Kearns, 2012), the equilibrium is where both players put their only seed on v *and each get an expected payoff equal to* $(4N+1)/2$, *while in the endogenous budgets model, the equilibrium is the case where there is exactly one Red and one Blue seed on each of* $v_1, v_2, ..., v_N$, *resulting in a payoff equal to* $N/2$ *for each player.*

Theorem 19 *Suppose* $c_p = c$ *for all* $p \in \{R, B\}$ *and* $c \in \mathbb{N}$. *Then regardless of* f, g, *in the endogenous budgets model the PoA can be unbounded.*

Proof We first prove the theorem for the case of $c = 1$. Consider the graph H represented in Figure 17. We claim that in this graph the maximum social welfare solution is the case where there is a single seed on each node in layer L_1. Also we claim that the equilibrium solution is the case where players both spend exactly one seed on each node of layer L_2. If this holds then it is easy to see that the PoA is equal to $\frac{3N+1+K^*-K^*}{2N+1-2N} = \frac{3N+1}{1} = 3N+1$; And since N can be arbitrarily large, we can conclude that the PoA is unbounded. Also we have $\frac{\theta^*}{\theta} = \frac{3N+1+K^*}{2N+1}$, and since K^* can be as large as we want, $\frac{\theta^*}{\theta}$ is unbounded as well.

It only remains to prove the above claims. First note that no node in layers L_3, L_4 is ever selected as a seed in the maximum social welfare solution, because selecting those nodes does not result is any new infections (as all the edges are directed from top to bottom). Therefore since layer L_3 is empty of seeds, putting one seed on a node of layer L_1 always pays off immediately as it has a neighbor in L_3 which becomes subsequently infected with probability 1 ($f(1) = 1$). This means that if a node in L_1 is not already seeded in the maximum social welfare solution, one can seed it without decreasing the payoff. So we can assume that in the maximum social welfare solution all the nodes in layer L_1 are seeded (it is easy to see that one seed per node suffices). Now note that once all the vertices in L_1 are seeded, the rest of the network will become infected with probability 1 ($f(1) = 1$). So we can conclude that this case is indeed the maximum social welfare solution.

Now we prove our claim about the equilibrium solution. Suppose the Red player has a seed on every node of layer L_2; we compute the best response of the Blue player to this strategy. Note that adding a seed to layers L_1, L_3, L_4 does not increase Blue's payoff, so her only choice is to allocate seeds to vertices in layer L_2. Suppose Blue has $k < N$ seeds on L_2 each on a different node. This means that the expected number of Blue seeds in L_2 is equal to $k/2$. Now note that for each new seed that Blue adds to a vertex u in L_2:

1. If Blue does not have any other seed on u, by adding one, she increases her payoff by a positive amount: with probability $\frac{1}{2}$, u becomes a Blue seed. If that happens, the 2 neighbors of u in layer L_4 certainly adapt Blue; also the probability that v adopts Blue increases (as g is increasing). Since the cost per seed is equal to 1 and the change in the expected number of Blue infections is larger than $\frac{1}{2} \times 2 = 1$, this action increases Blue's payoff.

2. If Blue does already have one seed on u, by adding another, at best she changes her payoff by $\frac{1}{6}(2+1) - 1 \leq 0$ (the change in the probability of u becoming initially Blue as a result of this new seed is $\frac{1}{6}$. If this happens, in the best case scenario, both v and the two neighbors of u in L_4 become subsequently infected). Since the change is

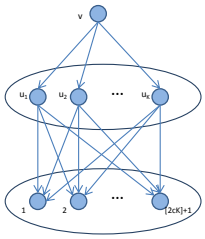


Figure 18: A graph with PoA $\geq K$.

negative this action decreases Blue's payoff.

The above shows that the best response of the Blue player is to spend exactly one seed on each vertex of L_2. Similarly, the best response of the Red player to Blue player's strategy is to put exactly one seed on every vertex of L_2, showing that this allocation is indeed an equilibrium. This finishes the proof for the case of $c = 1$.

It is now easy to see that the above example can be easily generalized to the case where the cost per seed is a positive integer c: It suffices to replace every node in layers L_3, L_4 with c vertices with the same neighbor set in L_1, L_2 as the original vertex. With a similar type of reasoning one can show that PoA is equal to $\frac{(2c+1)N+1+K^*c-K^*c}{2cN+1-2cN} = \frac{(2c+1)N+1}{1} = (2c+1)N+1$ which can be arbitrarily large by respectively choosing N sufficiently large. ∎

Corollary 2 *Suppose $c_p = c$ for all $p \in \{R, B\}$ and $c \geq 1$ ($c \in \mathbb{N}$). Then regardless of f, g, the quantity $\frac{\theta^*}{\theta}$ can be unbounded in the endogenous budgets model.*

Proof The same construction in the previous proof works here too. We have that $\frac{\theta^*}{\theta} = \frac{(2c+1)N+1+K^*c}{2cN+1}$, so it can be arbitrarily large by choosing K^* big enough. ∎

Finally we investigate the case where $c < 1$.

Theorem 20 *Suppose $c_p = c < 1$ for $p \in \{R, B\}$ and g is linear. Then regardless of f, the PoA can be unbounded in the endogenous budgets model.*

Proof Consider the graph H in Figure 18. We choose K sufficiently large. The maximum social welfare solution is obviously the case where a single seed is put on v, resulting in

payoff $K + [2cK] + 1 - c$ which is larger than K.

Next we claim that one equilibrium is the case where players put one seed on each of $u_1, ..., u_K$ resulting in total payoff of $[2cK] + 1 - 2cK < 1$, and therefore a PoA larger than K. To prove our claim, we just need to show that if the strategy of one player (say Red) is to put exactly one seed on each vertex of the second layer, then the best response of the Blue player will be to do the same thing. Suppose Blue chooses x vertices among $u_1, ..., u_K$ as her seed set. Then her expected payoff would be at most $(x/2K)([2cK] + 1) - cx$ which is increasing in x (one can easily see this by taking the derivative: $([2cK] + 1)/2K - c \geq 0$). This means that the best response is to choose $x = K$ which proves our claim.

Finally, we note that since K in the above construction can be chosen arbitrarily large, the PoA is unbounded and this finishes the proof. ∎

4.2.4. Discussion and Future Directions

We relaxed two of the main restricting assumption in previous papers on competitive influence maximization. We saw that exogenous budget constraints are crucial for obtaining upper bounds on the PoA and the Budget Multiplier. We also saw that if in addition to the number of adopters, firms take the connectivity among those adopters into account, upper bounds on the PoA and the Budget Multiplier still exist, but they can depend weakly on the network structure and the budget constraints. Our work suggests a number of additional interesting open problems, including the following:

1. In the endogenous model, we assumed that the cost per seed is a fixed value. It would be interesting to investigate the case where the cost is in fact a more complex function of the seed set.

2. In the connectivity model, we considered a simple notion of connectivity (i.e. the number of edges). Another interesting direction is to investigate more complex notions of connectivity.

CHAPTER 5 : Conclusion and Future Directions

The primary goal of this dissertation was to further our understanding of market algorithms within the *social and economic context* they operate in. As we saw earlier, society can impose various trade-offs on the market designer. I tackled multiple instances of legal, ethical, and strategic constraints in the context of online labor markets, prediction markets, ad exchanges, and beyond.

Significant future research is necessary before we have satisfactory answer to the numerous market design challenges we face in today's exceedingly online world. As the Internet becomes more and more an integrated part of modern society, the number of these challenges is inevitable to rise. In this chapter, I conclude with a brief overview of several broad, related research directions that remain largely open for future work.

5.1. Learning and Information Aggregation through Markets

One of the key components of an effective market algorithm is its ability to aggregate and react to information it accumulates from its participants. In some cases—e.g. prediction markets—this is the primary goal of the market mechanism, and in other instances—e.g. callouts in ad exchanges—information aggregation can be a powerful means to market's broader agenda.

We know that some market mechanisms outperform others in terms of their ability to learn and react to informative signals they receive from participants. For instance, in case of making predictions about future outcomes, under certain conditions cost function-based market making is known to perform significantly better than traditional methods of information aggregation, such as polling. Our current understanding ways in which information aggregates through markets is limited. Important areas for future work therefore includes:

- investigate the conditions under which one market algorithm outperforms another in terms of the speed and reliability with which it aggregates information;

- and the underlying forces driving this forward.

Obtaining a satisfactory answer to the above questions would require a clear understanding of market participants' behavior, and means to extract as much relevant information as possible from that behavior.

5.2. Modeling Market Participants' Behavior

A trivial prerequisite in designing any kind of algorithm is to know its inputs properly. Market algorithms are no exception: The first step in designing an effective market algorithm is to understand the behavior of the entities interacting with it. Unlike traditional algorithms, however, in case of market algorithms understanding this input is far from trivial. Market participants are usually sophisticated agents that use complex tools to drive their agenda forward. The challenge in modeling this behavior is, therefore, multifold:

- **Complex environment:** Market participants seldom operate in vacuum. They often interact with other market participants; participate in other, possibly competing platforms; and more broadly interact with and receive information from the outside world in numerous ways.

- **Belief formation and learning:** Given the complex environment they operate in, it is both intractable and unrealistic to model market participants as traditional Bayesian agents.

- **Evolving strategies:** Any intelligent agent is naturally expected to experiment with new strategies and methods over time, especially as its environment changes. An effective market algorithm must inevitably be capable of identifying and reacting to these changes–at least to a sufficient degree.

BIBLIOGRAPHY

Google's callout quota system. `https://developers.google.com/ad-exchange/rtb/callout-quota-system`, 2015.

J. Abernethy, Y. Chen, and J. W. Vaughan. Efficient market making via convex optimization, and a connection to online learning. *ACM Transactions on Economics and Computation*, 1(2), 2013.

A. Acquisti and H. R. Varian. Conditioning prices on purchase history. *Marketing Science*, 24(3):367–381, 2005.

M. Adler, S. Chakrabarti, M. Mitzenmacher, and L. Rasmussen. Parallel randomized load balancing. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pages 238–247. ACM, 1995.

C. C. Aggarwal and H. Wang. A survey of clustering algorithms for graph data. *Managing and Mining Graph Data*, pages 275–301, 2010.

S. Agrawal, E. Delage, M. Peters, Z. Wang, and Y. Ye. A unified framework for dynamic pari-mutuel information market design. *Operations Research*, 59(3), 2011.

N. Alon, M. Feldman, A. D. Procaccia, and M. Tennenholtz. A note on competitive diffusion through social networks. *Information Processing Letters*, 110(6):221–225, 2010.

K. Amin, A. Rostamizadeh, and U. Syed. Learning prices for repeated auctions with strategic buyers. In *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*, pages 1169–1177, 2013.

K. Amin, A. Rostamizadeh, and U. Syed. Repeated contextual auctions with strategic buyers. In *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*, pages 622–630, 2014.

J. R. Anderson and L. J. Schooler. Reflections of the environment in memory. *Psychological Science*, 2(6):396–408, 1991.

N. Archak, V. Mirrokni, and S. Muthukrishnan. Budget optimization for online campaigns with positive carryover effects. In *Proceedings of the the Workshop on Internet & Network Economics (WINE)*, 2012.

R. Arora, O. Dekel, and A. Tewari. Online bandit learning against an adaptive adversary: from regret to policy regret. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2012.

R. C. Atkinson, G. H. Bower, and E. J. Crothers. *Introduction to mathematical learning theory.* Wiley, 1965.

J.-Y. Audibert and S. Bubeck. Best arm identification in multi-armed bandits. In *Proceedings of the Conference on Learning Theory (COLT)*, pages 13–p, 2010.

P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002a.

P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002b.

L. M. Ausubel, J. E. Burkett, and E. Filiz-Ozbay. *An experiment on auctions with endogenous budget constraints*, 2013.

Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal. Balanced allocations. *SIAM Journal on Computing*, 29(1):180–200, 1999.

H. Azari, W. D. Heavlin, H. Heidari, S. Todorova, and M. Lin. A general framework for evaluating callout mechanisms in repeated auctions. *arXiv:1702.01803*, 2017.

S. R. Balseiro, O. Besbes, and G. Y. Weintraub. Repeated auctions with budgets in ad exchanges: Approximations and design. *Management Science*, 61(4):864–884, 2015.

Z. Bar-Yossef, K. Hildrum, and F. Wu. Incentive-compatible online auctions for digital goods. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 964–970. Society for Industrial and Applied Mathematics, 2002.

D. W. Barowy, C. Curtsinger, E. D. Berger, and A. McGregor. Automan: a platform for integrating human-based and digital computation. In *Proceedings of the Conference on Object-Oriented Programming Systems, Languages, and Applications (OOPSLA)*, 2012.

S. Basu and J. Christensen. Teaching classification boundaries to humans. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2013.

H. Berg and T. A. Proebsting. Hanson's automated market maker. *Journal of Prediction Markets*, 3(1):45–59, 2009.

J. Berg, R. Forsythe, F. Nelson, and T. Rietz. Results from a dozen years of election futures markets research. *Handbook of Experimental Economics Results*, 1:742–751, 2008.

S. Bharathi, D. Kempe, and M. Salek. Competitive influence maximization in social networks. In *Proceedings of the International Workshop on Web and Internet Economics (WINE)*, pages 306–311. Springer, 2007.

S. Bikhchandani. Reputation in repeated second-price auctions. *Journal of Economic Theory*, 46(1):97–119, 1988.

A. Blum, V. Kumar, A. Rudra, and F. Wu. Online learning in online auctions. *Theoretical Computer Science*, 324(2):137–146, 2004.

C. Borgs, J. Chayes, N. Immorlica, M. Mahdian, and A. Saberi. Multi-unit auctions with budget-constrained bidders. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, 2005.

C. Borgs, M. Brautbar, J. Chayes, and B. Lucier. Maximizing social influence in nearly optimal time. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 946–957. SIAM, 2014.

A. Borodin, Y. Filmus, and J. Oren. Threshold models for competitive influence in social networks. In *Proceedings of the Workshop on Internet & Network Economics (WINE)*, pages 539–550. Springer, 2010.

A. Borodin, M. Braverman, B. Lucier, and J. Oren. Strategyproof mechanisms for competitive influence in networks. *Algorithmica*, 78(2):425–452, 2017.

P. Bossaerts, L. Fine, and J. Ledyard. Inducing liquidity in thin financial markets through combined-value trading mechanisms. *European Economic Review*, (46):1671–1695, 2002.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge Uni. Press, 2009.

S. Bubeck and N. Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *arXiv:1204.5721*, 2012.

J. Burkett. Endogenous budget constraints in auctions. *Journal of Economic Theory*, 158: 1–20, 2015.

T. Carnes, C. Nagarajan, S. M. Wild, and A. Van Zuylen. Maximizing influence in a competitive social network: a follower's perspective. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, pages 351–360. ACM, 2007.

M. Cary, A. Das, B. Edelman, I. Giotis, K. Heimerl, A. R. Karlin, C. Mathieu, and M. Schwarz. Greedy bidding strategies for keyword auctions. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, pages 262–271. ACM, 2007.

N. Cesa-Bianchi, C. Gentile, and Y. Mansour. Regret minimization for reserve prices in second-price auctions. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1190–1204. Society for Industrial and Applied Mathematics, 2013.

D. Chakrabarty, Y. Zhou, and R. Lukose. Budget constrained bidding in keyword auctions and online knapsack problems. In *Proceedings of the International World Wide Web Conference (WWW)*, 2007.

M. Chakraborty, S. Das, and J. Peabody. Price evolution in a continuous double auction prediction market with a scoring-rule based market maker. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2015.

T. Chakraborty, E. Even-Dar, S. Guha, Y. Mansour, and S. Muthukrishnan. Selective

call out and real time bidding. In *Proceedings of the Workshop on Internet & Network Economics (WINE)*, 2010.

K. Chandrasekaran and R. Karp. Finding a most biased coin with fewest flips. *arXiv:1202.3639*, 2012.

V. Chaoji, S. Ranu, R. Rastogi, and R. Bhatt. Recommendations to boost content spread in social networks. In *Proceedings of the international conference on World Wide Web (WWW)*, pages 529–538. ACM, 2012.

O. Chapelle and L. Li. An empirical evaluation of thompson sampling. In *Proceedings of Advances in Neural Information Processing Systems*, pages 2249–2257, 2011.

W. Chen, Y. Wang, and S. Yang. Efficient influence maximization in social networks. In *Proceedings of the SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 199–208, 2009.

W. Chen, Y. Yuan, and L. Zhang. Scalable influence maximization in social networks under the linear threshold model. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pages 88–97. IEEE, 2010.

Y. Chen and D. Pennock. A utility framework for bounded-loss market makers. In *Proceedings of the annual Conference on Uncertainty in Artificial Intelligence (UAI)*, 2007.

H. Chouinard. *Repeated Auctions with the Right of First Refusal and Asymmetric Information*, 2006.

A. Clark and R. Poovendran. Maximizing influence in competitive environments: a game-theoretic approach. In *Proceedings of International Conference on Decision and Game Theory for Security*, pages 151–162. Springer, 2011.

J. Cremer, L. Garicano, and A. Prat. Language and the theory of the firm. *The Quarterly Journal of Economics*, 122(1):373–407, 2007.

N. R. Devanur, K. Jain, B. Sivan, and C. A. Wilkens. Near optimal online algorithms and fast approximation algorithms for resource allocation problems. In *Proceedings of the the ACM Conference on Electronic Commerce (EC)*, 2011.

D. DiPalantino and M. Vojnovic. Crowdsourcing and all-pay auctions. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, pages 119–128. ACM, 2009.

S. Dobzinski, R. Lavi, and N. Nisan. Multi-unit auctions with budget limits. In *Proceedings of the the IEEE Symposium on Foundations of Computer Science (FOCS)*, 2005.

M. Draief, H. Heidari, and M. Kearns. New models for competitive contagion. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2014.

S. Du, M. Ibrahim, M. Shehata, and W. Badawy. Automatic license plate recognition (alpr): A state-of-the-art review. *IEEE Transactions on Circuits and Systems for Video Technology*, 23(2):311–325, 2013.

P. Dubey, R. Garg, and B. De Meyer. Competing for customers in a social network: The quasi-linear case. In *Proceedings of the Workshop on Internet & Network Economics (WINE)*, pages 162–173. Springer, 2006.

M. Dudík, S. Lahaie, and D. Pennock. A tractable combinatorial market maker using constraint generation. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, 2012.

S. Dughmi, T. Roughgarden, and M. Sundararajan. Revenue submodularity. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, 2009.

D. Easley and J. Kleinberg. *Networks, crowds, and markets: Reasoning about a highly connected world*. Cambridge University Press, 2010.

B. Edelman and M. Ostrovsky. Strategic bidder behavior in sponsored search auctions. *Decision Support Systems*, 43(1):192–198, 2007.

C. Featherstone. A closer look at Topkis' theorem. Course notes, accessed at `http://stanford.io/1C9zQ00`, 2008.

D. Ferreira and R. K. Sah. Who gets to the top? generalists versus specialists in managerial organizations. *The RAND Journal of Economics*, 43(4):577–601, 2012.

H. Föllmer and A. Schied. Convex measures of risk and trading constraints. *Finance and Stochastics*, 6(4):429–447, 2002.

L. Fortnow, J. Kilian, D. M. Pennock, and M. P. Wellman. Betting Boolean-style: A framework for trading in securities based on logical formulas. *Decision Support Systems*, 39(1):87–104, 2005.

V. Gabillon, B. Kveton, Z. Wen, B. Eriksson, and S. Muthukrishnan. Adaptive submodular maximization in bandit setting. In *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*, pages 2697–2705, 2013.

L. Garicano. Hierarchies and the organization of knowledge in production. *Journal of Political Economy*, 108(5):874–904, 2000.

B. Gerard. Equilibrium distribution of prices and advertising. *Review of Economic Studies*, 44(3):465, 1977.

A. Ghosh and P. McAfee. Crowdsourcing with endogenous entry. In *Proceedings of the International World Wide Web Conference (WWW)*, pages 999–1008. ACM, 2012.

B. Ghosh, F. T. Leighton, B. M. Maggs, S. Muthukrishnan, C. G. Plaxton, R. Rajaraman, A. W. Richa, R. E. Tarjan, and D. Zuckerman. Tight analyses of two local load balancing algorithms. *SIAM Journal on Computing*, 29(1):29–64, 1999.

J. Gittins, K. Glazebrook, and R. Weber. *Multi-armed bandit allocation indices*. John Wiley & Sons, 2011.

R. Gonen. On the hardness of truthful online auctions with multidimensional constraints. In *Proceedings of the Conference on Computability in Europe*, pages 221–230. Springer, 2008.

A. Goyal, W. Lu, and L. V. Lakshmanan. Simpath: An efficient algorithm for influence maximization under the linear threshold model. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pages 211–220. IEEE, 2011.

S. Goyal and M. Kearns. Competitive contagion in networks. In *the ACM Symposium on Theory of Computing (STOC)*, 2012.

S. Goyal, H. Heidari, and M. Kearns. Competitive contagion in networks. *Games and Economic Behavior*, 2014.

T. Graepel, J. Q. Candela, T. Borchert, and R. Herbrich. Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 13–20, 2010.

G. M. Grossman and C. Shapiro. Informative advertising with differentiated products. *The Review of Economic Studies*, 51(1):63–81, 1984.

R. Gummadi, P. Key, and A. Proutiere. Repeated auctions under budget constraints: Optimal bidding strategies and equilibria. *SSRN working draft*, 2012.

I. E. Hafalir, R. Ravi, and A. Sayedi. A near pareto optimal auction with budget constraints. *Games and Economic Behavior*, 74:699–708, 2012.

M. T. Hajiaghayi, R. Kleinberg, and D. C. Parkes. Adaptive limited-supply online auctions. In *the ACM Conference on Electronic Commerce (EC)*, pages 71–80, 2004.

R. Hanson. *Book Orders for Market Scoring Rules*, 2003a.

R. Hanson. Combinatorial information market design. *Information Systems Frontiers*, 5 (1):105–119, 2003b.

R. Hanson. Logarithmic market scoring rules for modular combinatorial information aggregation. *Journal of Prediction Markets*, 1(1):3–15, 2007.

L. Harris. *Trading and Exchanges: Market Microstructure for Practitioners*. Oxford University Press, 2002.

A. C. Hax and N. S. Majluf. Organizational design: A survey and an approach. *Operations Research*, 29(3):417–447, 1981.

H. Heidari and M. Kearns. Depth-workload tradeoffs for workforce organization. In *Proceedings of Conference on Human Computation & Crowdsourcing (HCOMP)*, 2013.

H. Heidari, S. Lahaie, D. Pennock, and J. W. Vaughan. Integrating market makers, limit orders, and continuous trade in prediction markets. In *Proceedings of the ACM conference on Economics and Computation (EC)*, 2015.

H. Heidari, M. Kearns, and A. Roth. Tight policy regret bounds for improving and decaying bandits. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2016a.

H. Heidari, M. Mahdian, U. Syed, S. Vassilvistskii, and S. Yazdanbod. Pricing a low-regret seller. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2016b.

J. Hiriart-Urruty and C. Lemaréchal. *Fundamentals of Convex Analysis*. Springer Science & Business Media, 2001.

C.-J. Ho and J. W. Vaughan. Online task assignment in crowdsourcing markets. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 12, pages 45–51, 2012.

C.-J. Ho, Y. Zhang, J. W. Vaughan, and M. Van Der Schaar. Towards social norm design for crowdsourcing markets. In *Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

J. J. Horton and L. B. Chilton. The labor economics of paid crowdsourcing. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, pages 209–218. ACM, 2010.

M. Jofre-Bonet and M. Pesendorfer. Bidding behavior in a repeated procurement auction: A summary. *European Economic Review*, 44(4):1006–1020, 2000.

B. Jovanovic and Y. Nyarko. A bayesian learning model fitted to a variety of empirical learning curves. *Brookings Papers on Economic Activity. Microeconomics*, 1995:247–305, 1995.

Y. Kanoria and H. Nazerzadeh. Dynamic reserve prices for repeated auctions: Learning from bids. 2014.

D. R. Karger, S. Oh, and D. Shah. Iterative learning for reliable crowdsourcing systems. In *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*, pages 1953–1961, 2011.

Z. Katona, P. P. Zubcsek, and M. Sarvary. Network effects and personal influences: The diffusion of an online social network. *Journal of Marketing Research*, 48(3):425–443, 2011.

D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 137–146. ACM, 2003.

D. Kempe, J. Kleinberg, and É. Tardos. Influential nodes in a diffusion model for social networks. In *Proceedings of the International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 1127–1138. Springer, 2005.

B. Kitts and B. Leblanc. Optimal bidding on keyword auctions. *Electronic Markets*, 14(3): 186–201, 2004.

B. Kitts, P. Laxminarayan, B. Leblanc, and R. Meech. A formal analysis of search auctions including predictions on click fraud and bidding tactics. In *Workshop on Sponsored Search Auctions*, 2005.

R. Kleinberg and T. Leighton. The value of knowing a demand curve: Bounds on regret for online posted-price auctions. In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 594–605. IEEE, 2003.

P. Klemperer. Auction theory: A guide to the literature. *Journal of Economic Surveys*, 13 (3):227–286, 1999.

M. Kotowski. *First-Price Auctions with Budget Constraints*, 2013.

S. Lahaie, D. Parkes, and D. Pennock. An expressive auction design for online display advertising. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2008.

T. L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6(1):4–22, 1985.

J. Lange and N. Economides. A parimutuel market microstructure for contingent claims. *European Financial Management*, 11(1), 2005.

E. Law and L. v. Ahn. *Human computation*, volume 5. Morgan & Claypool Publishers, 2011.

B. Lucier. Beyond equilibria: Mechanisms for repeated combinatorial auctions. *arXiv:0909.5677*, 2009.

G. J. Mailath and L. Samuelson. *Repeated games and reputations: long-run relationships*. Oxford university press, 2006.

S. Mangal. *An Introduction to Psychology*. Sterling Publishers Pvt. Ltd, 2009.

A. Mas-Colell, M. D. Whinston, and J. R. Green. *Microeconomic Theory*. Oxford University Press, 1995.

M. Mohri and A. M. Medina. Learning theory and algorithms for revenue optimization in second price auctions with reserve. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 262–270, 2014.

E. Mossel and S. Roch. Submodularity of influence in social networks: From local to global. *SIAM Journal on Computing*, 39(6):2176–2188, 2010.

D. Nekipelov, V. Syrgkanis, and E. Tardos. Econometrics for learning agents. In *Proceedings of the ACM conference on Economics and Computation (EC)*, 2015.

G. Neu, A. Gyorgy, C. Szepesvari, , and A. Antos. Online markov decision processes under bandit feedback. *IEEE Transactions on Automatic Control*, 59, March 2014.

A. Neyman. Uniqueness of the shapley value. *Games and Economic Behavior*, 1(1):116–118, 1989.

N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani. *Algorithmic game theory*, volume 1. Cambridge University Press Cambridge, 2007.

M. Ostrovsky and M. Schwarz. Reserve prices in internet advertising auctions: A field experiment. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, pages 59–60. ACM, 2011.

A. Othman and T. Sandholm. Liquidity-sensitive automated market makers via homogeneous risk measures. In *Proceedings of the Workshop on Internet & Network Economics (WINE)*, pages 314–325. Springer, 2011.

S. Ozkan and S. D. Wu. *Competitive Equilibrium Analysis for Repeated Procurement Auctions.*

J. S. Pereyra. *Housing markets with endogenous budget constraints*, 2012.

M. Peters, A. M. So, and Y. Ye. A convex parimutuel formulation for contingent claim markets. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, 2006.

A. Prat. Hierarchies of processors with endogenous capacity. *Journal of Economic Theory*, 77(1):214–222, 1997.

H. Quan, A. Milicic, S. Vucetic, and J. Wu. A connectivity-based popularity prediction approach for social networks. In *Proceedings of the IEEE International Conference on Communications (ICC)*, pages 2098–2102. IEEE, 2012.

H. Robbins. Some aspects of the sequential design of experiments. In *Herbert Robbins Selected Papers*, pages 169–177. Springer, 1985.

D. M. Rothschild and D. M. Pennock. The extent of price misalignment in prediction markets. *Algorithmic Finance*, 3(1–2):2–20, 2014.

T. Roughgarden. The price of anarchy in games of incomplete information. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, pages 862–879, 2012.

M. Salek, Y. Bachrach, and P. Key. Hotspotting-a probabilistic graphical model for image object localization through crowdsourcing. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2013.

S. Seung. Eyewire: A game to map the brain. http://blog.eyewire.org/about/, 2015.

L. Shapley and M. Shubik. Trade using one commodity as a means of payment. *Journal of Political Economy*, pages 937–968, 1977.

L. S. Shapley. *A value for n-person games*, 1952.

A. Singla, I. Bogunovic, G. Bartók, A. Karbasi, and A. Krause. On actively teaching the crowd to classify. In *NIPS Workshop on Data Driven Education*, 2013.

A. Slivkins. Monotone multi-armed bandit allocations. In *Proceedings of the Conference on Learning Theory (COLT)*, pages 829–834. Citeseer, 2011.

L. K. Son and R. Sethi. Metacognitive control and optimal learning. *Cognitive Science*, 30 (4):759–774, 2006.

M. Streeter, D. Golovin, and A. Krause. Online learning of assignments. In *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*, pages 1794–1802, 2009.

V. Syrgkanis and E. Tardos. Composable and efficient mechanisms. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pages 211–220, 2013.

C. Tekin and M. Liu. Online learning of rested and restless bandits. *IEEE Transactions on Information Theory*, 58(8):5588–5611, 2012.

C. J. Thomas. *Market structure and the flow of information in repeated auctions.* Bureau of Economics, Federal Trade Commission, 1996.

W. R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

D. M. Topkis. Minimizing a submodular function on a lattice. *Operations Research*, 26(2), 1978.

L. Tran-Thanh, S. Stein, A. Rogers, and N. R. Jennings. Efficient crowdsourcing of unknown experts using bounded multi-armed bandits. *Artificial Intelligence*, 214:89–111, 2014.

V. Tzoumas, C. Amanatidis, and E. Markakis. A game-theoretic analysis of a competitive diffusion process over social networks. In *Proceedings of the Workshop on Internet & Network Economics (WINE)*, pages 1–14. Springer, 2012.

A. Vetta. Nash equilibria in competitive societies, with applications to facility location, traffic routing and auctions. In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 416–425. IEEE, 2002.

N. Vulkan, A. E. Roth, and Z. Neeman. *The handbook of market design*. OUP Oxford, 2013.

G. Wang, S. R. Kulkarni, H. V. Poor, and D. Osherson. Aggregating large sets of probabilistic forecasts by weighted coherent adjustment. *Decision Analysis*, 8:128–144, 2011.

R. Wilson. Sequential equilibria of asymmetric ascending auctions: The case of log-normal distributions. *Economic Theory*, 12(2):433–440, 1998.

B. Xiao, W. Yang, and J. Li. Optimal reserve price for the generalized second-price auction in sponsored search advertising. *Journal of Electronic Commerce Research*, 10(3):114, 2009.

A. C.-C. Yao. Probabilistic computations: Toward a unified measure of complexity. In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 222–227. IEEE, 1977.

H. Zhang, E. Horvitz, Y. Chen, and D. C. Parkes. Task routing for prediction tasks. In *Proceedings of the International Conference on Antonomous Agents and Multiagent Sytems (AAMAS)*, pages 889–896, 2012.