Algorithmic Game Theory

A Thesis
Presented to
The Academic Faculty

by

Aranyak Mehta

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Algorithms, Combinatorics and Optimization
Georgia Institute of Technology
August 2005

Algorithmic Game Theory

Approved by:

Richard J. Lipton, Advisor
College of Computing
Georgia Institute of Technology

Vijay V. Vazirani, Advisor
College of Computing
Georgia Institute of Technology

Milena Mihail
College of Computing
Georgia Institute of Technology

Craig Tovey
School of Industrial and Systems Engineering
Georgia Institute of Technology

Eric Vigoda
College of Computing
Georgia Institute of Technology

Date Approved: June 3, 2005

I dedicate this thesis to my parents, Anjani and Sitanshu Mehta.

ACKNOWLEDGEMENTS

I would like to thank Dick Lipton and Vijay Vazirani, for their continuous support and guidance throughout these five years. Thanks also to the rest of the theory and ACO group at Georgia Tech, faculty and students, who have created such a motivating and enriching research environment. Thanks, of course, to my collaborators and co-authors - the results presented here are products of our joint work. Finally, thanks to Swati, the fixed point of the map of my life, without whose support and patience I would never have finished this thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

The interaction of theoretical computer science with game theory and economics has resulted in the emergence of two very interesting research directions. First, it has provided a new model for algorithm design, which is to optimize in the presence of strategic behavior. Second, it has prompted us to consider the computational aspects of various solution concepts from game theory, economics and auction design which have traditionally been considered mainly in a non-constructive manner. In this thesis we present progress along both these directions.

We first consider optimization problems that arise in the design of combinatorial auctions. We provide an online algorithm in the important case of budget-bounded utilities. This model is motivated by the recent development of the business of online auctions of search engine advertisements. Our algorithm achieves a factor of $1 - 1/e$, via a new linear programming based technique to determine optimal tradeoffs between bids and budgets. We also provide lower bounds in terms of hardness of approximation in more general submodular settings, via a PCP-based reduction. Second, we consider truth-revelation in auctions, and provide an equivalence theorem between two notions of strategy-proofness in randomized auctions of digital goods. Last, we consider the problem of computing an approximate Nash equilibrium in multi-player general-sum games, for which we provide the first subexponential time algorithm.

CHAPTER I

INTRODUCTION

The interaction between game theory, economics and theoretical computer science has led to two main lines of research.

The first is the introduction of a new model of algorithms to theoretical computer science. The traditional model of algorithms is that of a black box, which takes in certain inputs and performs some computations to give an output. In this traditional model the algorithm is not supposed to question the origin of the inputs but is supposed to simply perform the optimization on the given input. In the new mechanism design model of computation, we take a wider view of what constitutes optimization. To do this, we further model the source of the inputs themselves. We assume that the different parts of the input originate from different *independent agents*, each of which has his own optimizations to perform, and is interested in only optimizing his own profit. This is modeled as certain utility functions which give an agent a certain amount of profit based on what the output of the algorithm is. Thus the agents may misreport the true value of the input so as to make the algorithm give a desirable input. In this new model, a goal of the algorithm designer would be to design an algorithm which would provide incentives to the agents to report their true utilities, so that the algorithm gives a globally optimized output.

While mechanism design is an old area of study, many known methods fail to be computationally tractable. This is where the contribution of theoretical computer science comes in. The goal of the research is to provide computationally tractable (e.g., polynomial time) algorithms or mechanisms, or to provide proofs of computational intractability of these problems. For example, the celebrated Vickrey-Clarke-Groves (VCG) mechanism [73, 12, 23] is well known to be the only truth-revealing mechanism which maximizes the social welfare in auction settings. However, a recent result (see [57]) shows that there is no polynomial time algorithm which can implement this mechanism for complex auction settings, under standard complexity assumptions. In such a scenario, what can computer science methods offer to alleviate the situation? The field of approximation algorithms provides the framework to develop *approximate mechanisms*, which approximate the maximum social welfare (e.g., [43]), or are approximately truth-revealing (e.g., [4]).

The second line of research that this interaction has brought about is the introduction of a large number of problems of economic interest to computational analysis. Solution concepts such as equilibrium prices in markets, Nash equilibria in games, fair division of goods, sharing of costs of a public good, etc. are very old ideas. But in most settings it is not known if these concepts are easy to compute. This is a second goal of this research. The computational complexity of many of these problems is yet unknown. In some cases the problems are known to be computationally intractable. This is a critique of such a concept – if one provably cannot compute the solution concept (under reasonable complexity assumptions) then the model is not a predictive model.

1.1 The contribution of this Thesis

In this thesis we present progress in both directions of research mentioned above.

The first contribution is progress towards the design of efficient auctions in complex settings. The simplest auction is the auction of a single item. In this case it is trivial to determine the bidder who maximizes social welfare (he who bids highest for the item). Furthermore it is well known how to design a truthful auction (the famous second-price auction). Here we consider a much more complicated setting in which there are a large number of goods which are desired by the bidders in specific bundles. This setting is known as a combinatorial auction. The final goal of research in such a setting is the design of computationally tractable truth-revealing auctions which also maximize certain objectives such as social welfare or auctioneer profit. This is a very difficult open problem. We take the first step towards this goal and begin with considering the simpler question of optimization when the bidders do not strategize but simply reveal their true utility. In particular we consider the optimization of social welfare (total utility of the bidders in an outcome) in certain combinatorial auctions. Upon disregarding strategic behavior of the bidders, social welfare also translates to revenue maximization. Another motivation to look at the (simpler) optimization version is that in settings in which we can actually compute the exact optimum, we can also design truthful auctions through the Vickrey-Clarke-Groves mechanism [73, 12, 23].

The particular auctions we consider are combinatorial auctions when the bidders have utilities which are submodular functions. We show how it is NP-hard to optimize social welfare up to a factor better than $1 - 1/e$ in the case of general submodular utilities. We also consider a very important special case of submodular utilities and provide an online algorithm which achieves a factor of $1 - 1/e$ for welfare/revenue maximization. This is the set of functions which are additive with an upper bound. These functions are motivated by one of the most important business models today - that of online auctions of search engine advertisements. In this setting advertisers (bidders) submit bids for various ads and also submit a daily budget, and the search engine (auctioneer) has to allocate ads in an online manner to the different advertisers in order to maximize revenue at the end of the day.

The design of strategy-proof or truthful auctions is a much more difficult goal. Indeed, it is known that in several models of combinatorial auctions it is difficult to even optimize, let alone design truthful auctions [43]. Strategy-proof auctions have been designed in very special cases only, e.g. single item auctions, k-item auctions, and unlimited item auctions (digital goods). In all these settings it is known that the only truthful auctions are those which are bid-independent. Recently, there has been much interest from the computer science community to see if randomization can help in designing truthful auctions with better properties [22]. With the power of randomization, the result that the only truthful auctions are the bid independent auctions no longer holds. One very important setting of auctions is that of digital goods, or goods in unlimited supply. We provide here an equivalence theorem for this setting: we show that any (randomized) auction which is truthful for risk-neutral bidders is in fact a randomization over (deterministic) bid-independent auctions. Hence in this setting, while randomization does help, we may restrict ourselves to top-level randomization (i.e., flip coins to choose between several deterministic bid-independent auctions).

In a second direction we consider the computation of an important game theoretic solution concept: a Nash equilibrium in a general-sum game. While it is possible to compute min-max strategies in zero-sum games (this being equivalent to linear programming), it has remained an open question if one can compute a Nash-equilibrium in non-zero sum games. The only known algorithms take a worst-case exponential time [46, 47, 75], and the exact computational complexity of this problem is not known. Here, we provide a partial answer. We show how to compute an ϵ-Nash equilibrium (strategies with at most ϵ incentive to deviate) in time $n^{O(\log n)}$. The approximate equilibria that we find also have other nice properties, e.g. they are uniform over a small support of pure strategies. We believe that this is an important step towards solving the big question of the complexity of finding Nash equilibria.

1.2 Structure of the Thesis

Having motivated the results of this thesis above, we sketch below the structure of the chapters, and the technical statements of each result.

- **Chapter 2**: We present a positive result for an important special case of optimization in combinatorial auctions under submodular utilities. This is the model of online auction of search engine advertisements. We achieve this by providing a $1-1/e$ factor online algorithm for maximizing revenue. We also provide a new linear programming based technique for finding optimal trade-offs between different relevant parameters, e.g., the bid and the budget of a bidder, in an online manner. We call this new technique a tradeoff revealing family of linear programs. This generalizes two different algorithms for special cases of the problem via this single technique.

- **Chapter 3**: Here we consider the general problem of maximizing total utility/social welfare in a combinatorial auction under general submodular utility functions. We prove that it is NP-hard to find the optimal social welfare in such a setting up to a factor better than $1 - 1/e$. Our result holds even when the number of bidders is a constant, and all the bidders have the same submodular utilities. The results of Chapters 2 and 3 do not consider strategic behavior of the bidders.

- **Chapter 4**: In this chapter we consider the notion of truth-revealing auctions in the presence of randomization. We consider two well-known definitions of truth-revelation in randomized auctions, and show how these are, in fact, equivalent in the special case of digital goods. We show that every randomized auction which is truthful in expectation (truth-revealing against risk-neutral bidders) is equivalent to one which is a randomization over deterministic bid-independent auctions. We also show how, given an auction which is truthful in expectation, to find in polynomial time, an approximately equivalent auction which is a randomization over deterministic bid-independent auctions.

- **Chapter 5**: Given a general sum game with a constant number of players, and in which each player has n pure strategies to randomize over, we prove the existence of ϵ-Nash equilibria in which each player randomizes only over $\log n/\epsilon^2$ pure strategies. This also yields the first sub-exponential algorithm to find an approximate Nash equilibrium (in time $n^{O(\log n)}$). We also prove a structural result which says that in a two-player game, if the payoff matrices have rank k, then there exists a Nash equilibrium in which both players randomize only over k pure strategies.

CHAPTER II

ADWORDS AUCTIONS AND GENERALIZED ONLINE MATCHING

How does a search engine company decide what ads to display with each query so as to maximize its revenue? This turns out to be a generalization of the online bipartite matching problem. In this chapter we introduce the notion of a tradeoff revealing linear program and use it to derive two optimal algorithms achieving competitive ratios of $1 - 1/e$ for this problem.

This setting is a special case of a combinatorial auction under submodular utilities. This is one of the most elegant and realistic auction models for which there is no truth-revealing auction known which can maximize the revenue for the search engine or which can maximize the total utility of the advertisers. In this chapter we model this search engine auction in an online setting, and provide the first online algorithms to achieve competitive factor of $1 - 1/e$. However, this is not a truth-revealing auction, since it assumes that the bidders bid their true utilities. The design of a truth-revealing auction in this setting remains open.

2.1 Introduction

Internet search engine companies, such as Google, Yahoo and MSN, have revolutionized not only the use of the Internet by individuals but also the way businesses advertise to consumers. Instead of flooding consumers with unwanted ads, search engines open up the possibility of a dialogue between consumers and businesses, with consumers typing in keywords, called Adwords by Google, that reveal what they are looking for and search engines displaying highly targeted ads relevant to the specific query.

The AdWords market[1] is essentially a large auction where businesses place bids for individual keywords, together with limits specifying their maximum daily budget. The

[1] This market dwarfs the AdSense market where the ad is based on the actual contents of the website.

search engine company earns revenue from businesses when it displays their ads in response to a relevant search query (if the user actually clicks on the ad). Indeed, most of the revenues of search engine companies are derived in this manner[2].

In this context, the following computational problem, which we call the Adwords problem, was recently posed by Henzinger [25]: assign user queries to advertisers to maximize the total revenue. Observe that the task is necessarily online – when returning results of a specific query, the search engine company needs to immediately determine what ads to display on the side.

It is easy to see that the competitive ratio of the algorithm that awards each query to the highest bidder is $1/2$; moreover, this is tight. In this chapter, we present two algorithms, one deterministic and one randomized, achieving competitive ratios of $1-1/e$ for this problem. In Section 2.9 we show that no randomized algorithm can achieve a better competitive ratio.

The offline version of the Adwords problem is **NP**-hard, and the best known approximation algorithm achieves a guarantee of $1-1/e$ [2], by solving a linear program followed by randomized rounding. Our online algorithms achieve the same approximation guarantee and are more efficient: the total running time is $O(NM)$ where N is the number of bidders and M the length of the query sequence.

In Section 2.7 we show how our algorithm and analysis can be generalized to the following, more realistic, situations, while still maintaining the same competitive ratio:

- A bidder pays only if the user clicks on his ad.
- Advertisers have different daily budgets.
- Instead of charging a bidder his actual bid, the search engine company charges him the next highest bid.
- Multiple ads can appear with the results of a query.
- Advertisers enter at different times.

[2] According to a recent New York Times article (Feb 4, 2005), the revenue accrued by Google from this market in the last three months of 2004 alone was over a billion dollars.

2.1.1 Previous work

The adwords problem is clearly a generalization of the online bipartite matching problem: the special case where each advertiser makes unit bids and has a unit daily budget is precisely the online matching problem. Even in this special case, the greedy algorithm achieves a competitive ratio of 1/2. The algorithm that allocates each query to a random interested advertiser does not do much better – it achieves a competitive ratio of $1/2 + O(\log n/n)$.

In [33], Karp, Vazirani and Vazirani gave a randomized algorithm for the online matching problem achieving a competitive ratio of $1 - 1/e$. Their algorithm, called RANKING, fixes a random permutation of the bidders in advance and breaks ties according to their ranking in this permutation. They further showed that no randomized online algorithm can achieve a better competitive ratio.

In another direction, Kalyanasundaram and Pruhs [32] considered the online b-matching problem which can be described as a special case of the adwords problem as follows: each advertiser has a daily budget of b dollars, but makes only 0/1 dollar bids on each query. Their online algorithm, called BALANCE, awards the query to that interested advertiser who has the highest unspent budget. They show that the competitive ratio of this algorithm tends to $1 - 1/e$ as b tends to infinity. They also prove a lower bound of $1 - 1/e$ for deterministic algorithms.

2.1.2 Our results

To generalize the algorithms of [32] and [33] to arbitrary bids, it is instructive to examine the special case with bids restricted to $\{0, 1, 2\}$. The natural algorithm to try assigns each query to a highest bidder, using the previous heuristics to break ties (largest remaining budget/ highest ranking in the random permutation). We give examples (in the Appendix) showing that both these algorithms achieve competitive ratios strictly smaller and bounded away from $1 - 1/e$.

This indicates the need to consider a much more delicate tradeoff between the bid versus the remaining budget in the first case, and the bid versus the position in the random permutation in the second. The correct tradeoff function is derived by a novel LP-based approach, which we outline below. The resulting algorithms are very simple, and are based on the following tradeoff function:

$$\psi(f) = 1 - e^{-(1-f)}$$

We provide two algorithms which achieve a factor of $1 - 1/e$:

Algorithm 1:

Allocate the next query to the bidder i maximizing the product of his bid and $\psi(T(i))$, where $T(i)$ is the fraction of the bidder's budget which has been spent so far, i.e. $T(i) = \frac{m_i}{b_i}$, where b_i is the total budget of bidder i, m_i is the amount of money spent by bidder i.

Algorithm 2:

Start by permuting the advertisers at random. Allocate the next query to the bidder maximizing the product of his bid and $\psi(r/n)$, where r is the rank of this bidder in the random order and n is the number of bidders.

Both algorithms assume that the daily budget of advertisers is large compared to their bids.

2.1.3 A New Technique

We now outline how we derive the correct tradeoff function. For this we introduce the notion of a *tradeoff-revealing family of LP's*. This concept builds on the notion of a *factor-revealing LP* [28]. We start by writing a factor-revealing LP to analyze the performance in the special case when all bids are equal. This provides a simpler proof of the Kalyanasundaram and Pruhs [32] result.

We give an LP, L, whose constraints are satisfied at the end of a run of BALANCE on any instance π of the equal bids case. The objective function of L gives the performance of BALANCE on π. Hence the optimal objective function value of L is a lower bound on the competitive ratio of BALANCE. How good is this lower bound? Clearly, this depends on the constraints we have captured in L. It turns out that the bound computed by our LP is $1 - 1/e$ which is tight. Indeed, for some fairly sophisticated algorithms, e.g., [28, 7], a factor-revealing LP is the only way known of deriving a tight analysis.

Next, we handle the case of arbitrary bids. We write a family of LP's $L(\pi, \psi)$, one for each input instance π and decreasing tradeoff function ψ. The objective function of $L(\pi, \psi)$ gives the performance of Algorithm 1 when run on π with tradeoff function ψ. The problem now is to choose ψ that yields a performance of at least $1 - 1/e$ on every instance π. Once the input instance and tradeoff functions are fixed, of course, the exact results achieved by the algorithm are completely determined. The right hand side of the inequalities in the LP $L(\pi, \psi)$ are based on these (unknown) parameters.

Now the constraints in each LP are obtained by relaxing a tautology, and therefore any single LP in this family does not contain any useful information. However, the entire family does express some of the structure of the problem which is revealed by considering the family of dual linear programs $D(\pi, \psi)$.

It turns out that $L(\pi, \psi)$ differs from L only in that a vector $\Delta(\pi, \psi)$ is added to the right hand side of the constraints. Therefore, the dual programs $D(\pi, \psi)$ differ from D only in the objective function, which is changed by $\Delta(\pi, \psi) \cdot \boldsymbol{y}$, where $\boldsymbol{y}$ is the vector of dual variables. Hence the dual polytope for all LP's in the family is the same as that for D. Moreover, we show that D and the each LP in the family $D(\pi, \psi)$ attains its optimal value at the same vertex, $\boldsymbol{y}^*$, of the dual polytope. Finally, we show how to use $\boldsymbol{y}^*$ to define ψ in a specific manner so that $\Delta(\pi, \psi) \cdot \boldsymbol{y}^* \leq 0$ for each instance π (observe that this function ψ does not depend on π and hence it works for all instances). This function is precisely the function used in Algorithm 1. This ensures that the performance of Algorithm 1 on each instance matches that of BALANCE on unit bid instances and is at least $1 - 1/e$.

We call this ensemble $L(\pi, \psi)$ a *tradeoff revealing family of LP's*. Once the competitive ratio of the algorithm for the unit bid case is determined via a factor-revealing LP, this family helps us find a tradeoff function that ensures the same competitive ratio for the arbitrary bids case.

The same proof outline also applies to Algorithm 2 once we suitably simplify the analysis of Karp, Vazirani and Vazirani [33] and cast it in terms of linear constraints.

2.2 Problem Definition

The Adwords problem is the following: There are N bidders, each with a specified daily budget b_i. Q is a set of query words. Each bidder i specifies a bid c_{iq} for query word $q \in Q$. A sequence $q_1 q_2 \dots q_M$ of query words $q_j \in Q$ arrive online, and each query q_j must be assigned to some bidder i (for a bid of $c_{ij} = c_{iq_j}$). The objective is to maximize the total revenue while respecting the daily budgets of the bidders.

Throughout this chapter we will make the assumption that the bids are small compared to the budgets, i.e., $\max_{i,j} c_{ij}$ is small compared to $\min_i b_i$. For the applications of this problem mentioned in the Introduction, this is a reasonable assumption.

An online algorithm is said to be α-competitive if for every instance, the ratio of the revenue of the online algorithm to the revenue of the best off-line algorithm is at least α.

While presenting the algorithms and proofs, we will make the simplifying assumptions that the budgets of all bidders are equal (assumed unit) and that the best offline algorithm exhausts the budget of each bidder. These assumptions will be relaxed in Section 2.7.

2.3 A Deterministic Algorithm

Let us first consider a greedy algorithm that maximizes revenue accrued at each step. It is easy to see that this algorithm achieves a competitive ratio of $\frac{1}{2}$ (see, e.g., [42]); moreover, this is tight as shown by the following example with only two bidders and two query words: Suppose both bidders have unit budget. The two bidders bid c and $c + \epsilon$ respectively on query word q, and they bid 0 and c on query word q'. The query sequence consists of a number of occurrences of q followed by a number of occurrences of q'. The query words q are awarded to bidder 2, and are just enough in number to exhaust his budget. When query words q' arrive, bidder 2's budget is exhausted and bidder 1 is not interested in this query word, and they accrue no further revenue.

Algorithm 1 rectifies this situation by taking into consideration not only the bids but also the unspent budget of each bidder. For the analysis it is convenient to discretize the budgets as follows: we pick a large integer k, and discretize the budget of each bidder into k equal parts (called *slabs*) numbered 1 through k. Each bidder spends money in slab j before moving to slab $j+1$.

Definition: At any time during the run of the algorithm, we will denote by slab(i) the currently active slab for bidder i.

Let $\psi : [1 \dots k] \to \mathbf{R}_+$ be the following (monotonically decreasing) function:

$$\psi(i) = 1 - e^{-(1-i/k)}$$

Algorithm 1

```
1. When a new query arrives, let the bid of bidder $i$ be $c(i)$.
2. Allocate the query to the bidder $i$ who maximizes $c(i) \times \psi(\text{slab}(i))$.
```

Note that in the special case when all the bids are equal, our algorithm works in the same way as the BALANCE algorithm of [32], for any monotonically decreasing tradeoff function.

2.4 Analyzing BALANCE using a Factor-Revealing LP

In this section we analyze the performance of Algorithm 1 in the special case when all bids are equal. This is exactly the algorithm BALANCE of [32]. We give a simpler analysis of this algorithm using the notion of a factor-revealing LP. This technique was implicit in [52, 20, 50] and was formalized and made explicit in [29, 28].

We will assume for simplicity that in the optimum solution, each of the N players spends his entire budget, and thus the total revenue is N. Recall that BALANCE awards each query to the interested bidder who has the maximum unspent budget. We wish to lower bound the total revenue achieved by BALANCE. Let us define the *type* of a bidder according to the fraction of budget spent by that bidder at the end of the algorithm BALANCE: say that the bidder is of type j if the fraction of his budget spent at the end of the algorithm lies in the range $((j-1)/k, j/k]$. By convention a bidder who spends none of his budget is assigned type 1.

Clearly bidders of type j for small values of j contribute little to the total revenue. The factor revealing LP for the performance of the algorithm BALANCE will proceed by bounding the number of such bidders of type j.

Lemma 1 *If OPT assigns query q to a bidder of type j, then BALANCE pays for q from slab k such that $k \leq j$.*

The lemma follows immediately from the criterion used by BALANCE for assigning queries to bidders.

For simplicity we will assume that bidders of type i spend exactly i/k fraction of their budget. The total error resulting from this simplification is at most N/k and is negligible. Now, for $i = 1, 2, \ldots, k-1$, let x_i be the number of bidders of type(i). Let β_i denote the total money spent by the bidders from slab i in the run of ALG. It is easy to see (Figure 1) that $\beta_1 = N/k$, and for $2 \leq i \leq k$, $\beta_i = N/k - (x_1 + \ldots + x_{i-1})/k$.

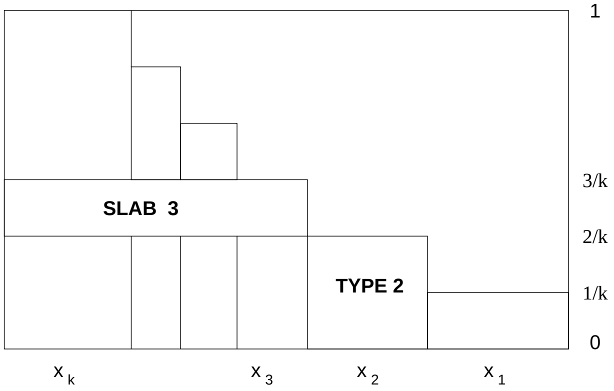

Figure 1: Definition of slabs and types: The bidders are ordered from right to left in order of increasing type. We have labeled here the bidders of type 2 and the money in slab 3.

Lemma 2

$$\forall\ i,\ 1 \leq i \leq k-1 : \qquad \sum_{j=1}^{i} (1 + \frac{i-j}{k}) x_j \leq \frac{i}{k} N$$

Proof : By Lemma 1,

$$\sum_{j=1}^{i} x_j \ \leq\ \sum_{j=1}^{i} \beta_j = \frac{i}{k} N - \sum_{j=1}^{i} (\frac{i-j}{k}) x_j$$

The lemma follows by rearranging terms. □

The revenue of the algorithm is

$$\begin{aligned} ALG &\geq \sum_{i=1}^{k-1} \tfrac{i}{k} x_i + \left(N - \sum_{i=1}^{k-1} x_i \right) - \tfrac{N}{k} \\ &= N - \sum_{i=1}^{k-1} \tfrac{k-i}{k} x_i - \tfrac{N}{k} \end{aligned}$$

To find a lower bound on the performance of BALANCE we want to find the minimum value that $N - \sum_{i=1}^{k-1} \frac{k-i}{k} x_i - \frac{N}{k}$ can take over the feasible $\{x_i\}s$. This gives the following LP, which we call L:

$$\begin{aligned} &\text{maximize} && \Phi = \sum_{i=1}^{k-1} \tfrac{k-i}{k} x_i && (1) \\ &\text{subject to} && \forall\ i,\ 1 \leq i \leq k-1: \quad \sum_{j=1}^{i} (1 + \frac{i-j}{k}) x_j \leq \frac{i}{k} N \\ & && \forall\ i,\ 1 \leq i \leq k-1: \quad x_i \geq 0 \end{aligned}$$

Let us also write down the dual LP, D, which we will use in the case of arbitrary bids.

$$\begin{aligned} &\text{minimize} && \sum_{i=1}^{k-1} \tfrac{i}{k} N y_i \\ &\text{subject to} && \forall\ i,\ 1 \leq i \leq k-1: \quad \sum_{j=i}^{k-1} (1 + \frac{j-i}{k}) y_j \geq \frac{k-i}{k} \\ & && \forall\ i,\ 1 \leq i \leq k-1: \quad y_i \geq 0 \end{aligned}$$

Define $\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}$ so the primal LP, L, can be written as

$$\max \quad \boldsymbol{c} \cdot \boldsymbol{x} \quad s.t. \quad \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \quad \boldsymbol{x} \geq 0.$$

and the dual LP, D, can be written as

$$\min \quad \boldsymbol{b} \cdot \boldsymbol{y} \quad s.t. \;\; \boldsymbol{A}^T \boldsymbol{y} \geq \boldsymbol{c} \quad \boldsymbol{y} \geq 0.$$

Lemma 3 *As $k \rightarrow \infty$, the value Φ of the above linear program goes to $\frac{N}{e}$*

Proof : On setting all the primal constraints to equality and solving the resulting system, we get a feasible solution $x_i^* \geq 0$. Similarly, we can set all the dual constraints to equality and solve the resulting system to get a feasible dual solution.

These two feasible solutions are:

$$x_i^* = \tfrac{N}{k}(1 - \tfrac{1}{k})^{i-1} \quad \text{for } i = 1, .., k-1$$

$$y_i^* = \tfrac{1}{k}(1 - \tfrac{1}{k})^{k-i-1} \quad \text{for } i = 1, .., k-1$$

But these clearly satisfy the complementary slackness conditions, hence they are also optimal solutions of the primal and dual programs.

This gives an optimal objective function value of

$$\begin{aligned} \Phi &= \boldsymbol{c} \cdot \boldsymbol{x}^* \;=\; \boldsymbol{b} \cdot \boldsymbol{y}^* \\ &= \sum_{i=1}^{k-1} (\tfrac{k-i}{k}) \tfrac{N}{k}(1 - \tfrac{1}{k})^{i-1} \\ &= N(1 - \tfrac{1}{k})^k \end{aligned}$$

As we make the discretization finer (i.e. as $k \rightarrow \infty$) Φ tends to $\frac{N}{e}$.

□

Recall that the size of the matching is at least $N - \Phi - \frac{N}{k}$, hence it tends to $N(1 - \frac{1}{e})$. Since OPT is N, the competitive ratio is at least $1 - \frac{1}{e}$.

On the other hand one can find an instance of the problem (e.g., the one provided in [32]) such at the end of the algorithm all the inequalities of the primal are tight, hence the competitive ratio of ALG is exactly $1 - \frac{1}{e}$.

2.5 A Tradeoff-Revealing Family of LPs for the Adwords Problem

Observe that even if we knew the correct tradeoff function, extending the methods of the previous section is difficult. The problem with mimicking the factor-revealing LP for constant bids is that now the tradeoff between bid and unspent budget is subtle and the basic Lemma 1 which allowed us to write the inequalities in the LP no longer holds.

Here is how we proceed instead: We fix both a monotonically decreasing tradeoff function ψ as well as the instance π of the adwords problem and write a new LP $L(\pi, \psi)$ for Algorithm 1 using tradeoff function ψ run on the instance π. Of course, once we specify the algorithm as well as the input instance, the actual allocations of queries to bidders is completely determined. In particular, the number α_i of bidders of type i is fixed. $L(\pi, \psi)$ is the seemingly trivial LP obtained by taking the left hand side of each inequality in the factor revealing LP and substituting $x_i = \alpha_i$ to obtain the right hand side. Formally:

Let $\boldsymbol{a}$ be a $k-1$ dimensional vector whose ith component is α_i. Let $\boldsymbol{Aa} = \boldsymbol{l}$. We denote the following LP by $L(\pi, \psi)$:

$$\max \quad \boldsymbol{c} \cdot \boldsymbol{x} \quad s.t. \quad \boldsymbol{Ax} \leq \boldsymbol{l} \quad \boldsymbol{x} \geq 0$$

The dual LP is denoted by $D(\pi, \psi)$ and is:

$$\min \quad \boldsymbol{l} \cdot \boldsymbol{y} \quad s.t. \quad \boldsymbol{A}^T \boldsymbol{y} \geq \boldsymbol{c} \quad \boldsymbol{y} \geq 0$$

Clearly, any one LP $L(\pi, \psi)$ offers no insight into the performance of Algorithm 1; after all the right hand sides of the inequalities are expressed in terms of the unknown number of bidders of type i. Nevertheless, the entire family $L(\pi, \psi)$ does contain useful information which is revealed by considering the duals of these LP's.

Since $L(\pi, \psi)$ differs from L only in the right hand side, the dual $D(\pi, \psi)$ differs from D only in the dual objective function; the constraints remain unchanged. Hence solution $\boldsymbol{y}^*$ of D is feasible for $D(\pi, \psi)$ as well. Recall that this solution was obtained by setting all nontrivial inequalities of D to equality.

Now by construction, if we set all the nontrivial inequalities of LP $L(\pi, \psi)$ to equality we get a feasible solution, namely $\boldsymbol{a}$. Clearly, $\boldsymbol{a}$ and $\boldsymbol{y}^*$ satisfy all complementary slackness conditions. Therefore they are both optimal. Hence we get:

Lemma 4 *For any instance π and monotonically decreasing tradeoff function ψ, $\boldsymbol{y}^*$ is an optimal solution to $D(\pi, \psi)$.*

The structure of Algorithm 1 does constraint ow the LP L differs from $L(\pi, \psi)$. This is what we will explore now.

As in the analysis of BALANCE, we divide the budget of each bidder into k equal *slabs*, numbered 1 to k. Money in slab i is spent before moving to slab $i + 1$. We say that a bidder is of type j if the fraction of his budget spent at the end of Algorithm 1 lies in the range $((j-1)/k, j/k]$. By convention a bidder who spends none of his budget is assigned type 1. As before, we make the simplifying assumption (at the cost of a negligible error term) that bidders of type j spend exactly j/k fraction of their budget. Let α_j denote the number of bidders of type j. Let β_i denote the total money spent by the bidders from slab i in the run of Algorithm 1. It is easy to see that $\beta_1 = N/k$, and for $2 \leq i \leq k$, $\beta_i = N/k - (\alpha_1 + \ldots + \alpha_{i-1})/k$.

We are interested in comparing the performance of Algorithm 1 (abbreviated as ALG) with the optimal algorithm OPT. The following definitions focus on some relevant parameters comparing how ALG and OPT treat a query q:

Definition: Let $\text{ALG}(q)$ $(\text{OPT}(q))$ denote the revenue earned by Algorithm 1 (OPT) for query q. Say that a query q is of *type* i if OPT assigns it to a bidder of type i, and say that q lies in *slab* i if Algorithm 1 pays for it from slab i.

Lemma 5 *For each query q such that* $1 \leq \text{type}(q) \leq k-1$,

$$\text{OPT}(q)\psi(\text{type}(q)) \leq \text{ALG}(q)\psi(\text{slab}(q)).$$

Proof : Consider the arrival of q during the run of Algorithm 1. Since $\text{type}(q) \leq k-1$, the bidder b to whom OPT assigned this query is still actively bidding from some slab

$j \leq \text{type}(q)$ at this time. The inequality in the lemma follows from the criterion used by Algorithm 1 to assign queries, together with the monotonicity of ψ.

□

Lemma 6

$$\sum_{i=1}^{k-1} \psi(i)(\alpha_i - \beta_i) \leq 0.$$

Proof : We start by observing that for $1 \leq i \leq k-1$:

$$\sum_{q:\text{type}(q)=i} \text{OPT}(q) = \alpha_i$$

$$\sum_{q:\text{slab}(q)=i} \text{ALG}(q) = \beta_i$$

By Lemma 5

$$\sum_{q:\text{type}(q)\leq k-1} [\text{OPT}(q)\psi(\text{type}(q)) - \text{ALG}(q)\psi(\text{slab}(q))] \quad \leq \quad 0.$$

Next observe that

$$\begin{aligned} \sum_{q:\text{type}(q)\leq k-1} \text{OPT}(q)\psi(\text{type}(q)) &= \sum_{i=1}^{k-1} \sum_{q:\text{type}(q)=i} \text{OPT}(q)\psi(i) \\ &= \sum_{i=1}^{k-1} \psi(i)\alpha_i. \end{aligned}$$

And

$$\begin{aligned} \sum_{q:\text{type}(q)\leq k-1} \text{ALG}(q)\psi(\text{slab}(q)) &\leq \sum_{q:\text{slab}(q)\leq k-1} \text{ALG}(q)\psi(\text{slab}(q)) \\ &= \sum_{i=1}^{k-1} \sum_{q:\text{slab}(q)=i} \text{ALG}(q)\psi(i) \\ &= \sum_{i=1}^{k-1} \psi(i)\beta_i. \end{aligned}$$

The lemma follows from these three inequalities. □

Let $\Delta(\pi,\psi)$ be a $k-1$ dimensional vector whose ith component is $(\alpha_1-\beta_1)+\ldots+(\alpha_i-\beta_i)$. The following lemma relates the right hand side of the LPs L and $L(\pi,\psi)$.

Lemma 7

$$\boldsymbol{l} = \boldsymbol{b} + \Delta(\pi,\psi).$$

Proof : Consider the ith components of the three vectors. We need to prove:

$$\alpha_1(1+\tfrac{i-1}{k})+\alpha_2(1+\tfrac{i-2}{k})+\ldots+\alpha_i=\tfrac{iN}{k}+(\alpha_1-\beta_1)+\ldots+(\alpha_i-\beta_i).$$

This equation follows using the fact that $\beta_i = N/k - (\alpha_1+\ldots+\alpha_{i-1})/k$. □

Theorem 8 *For function ψ defined as*

$$\psi(i) \;:=\; \sum_{j=i}^{k-1} y_j^* \;=\; 1-(1-\frac{1}{k})^{k-i+1}$$

the competitive ratio of Algorithm 1 is $(1-\frac{1}{e})$.

Proof : By Lemma 4, the optimal solution to $L(\pi,\psi)$ and $D(\pi,\psi)$ has value $\boldsymbol{l}\cdot\boldsymbol{y}^*$. By Lemma 7 this equals $(\boldsymbol{b}+\Delta)\cdot\boldsymbol{y}^* \le N/e+\Delta\cdot\boldsymbol{y}^*$ (since $\boldsymbol{b}\cdot\boldsymbol{y}^*\le N/e$, from Section 2.4).

Now,

$$\begin{aligned}\Delta\cdot\boldsymbol{y}^* &= \sum_{i=1}^{k-1}\boldsymbol{y}_i^*((\alpha_1-\beta_1)+\ldots+(\alpha_i-\beta_i)\\ &= \sum_{i=1}^{k-1}(\alpha_i-\beta_i)(y_i^*+\ldots+y_{k-1}^*)\\ &= \sum_{i=1}^{k-1}(\alpha_i-\beta_i)\psi(i)\\ &\le 0,\end{aligned}$$

where the last equality follows from our choice of the function ψ, and the inequality follows from Lemma 6. Hence the competitive ratio of Algorithm 1 is $(1-\frac{1}{e})$.

□

2.6 Direct Analysis of Algorithm 1

In the last section we derived the correct tradeoff function ψ to use in Algorithm 1, and in the process also gave a proof that the competitive ratio of the resulting algorithm is $1-1/e$. In this section abstract out the essential features of the argument and sketch a direct proof of the competitive ratio starting with this tradeoff function ψ.

Theorem 9 *The competitive ratio of Algorithm 1 is* $1-1/e$.

Proof : Recall that α_i is the number of bidders of type i, and β_i is the total amount of money spent by bidders from slab i. We have the following relations from Section 2.5:

$$\forall i : \quad \beta_i = \frac{N - \sum_{j=1}^{i-1} \alpha_j}{k}$$

$$\sum_i \psi(i)\alpha_i \ \le \ \sum_i \psi(i)\beta_i$$

Using the above equations and the choice of ψ:

$$\psi(i) = 1 - (1 - \frac{1}{k})^{k-i+1}$$

we get:

$$\sum_{i=1}^{k} \alpha_i \frac{k-i+1}{k} \le \frac{N}{e} \tag{2}$$

But the left side of (2) is precisely the amount of money left unspent at the end of the algorithm. Hence the factor of the ψ-based algorithm is at least $1 - 1/e$. □

2.7 Towards more realistic models

In this section we show how our algorithm and analysis can be generalized to the following situations:

1. Advertisers have different daily budgets.
2. The optimal allocation does not exhaust all the money of advertisers
3. Advertisers enter at different times.
4. More than one ad can appear with the results of a query. The most general situation is that with each query we are provided a number specifying the maximum number of ads.
5. A bidder pays only if the user clicks on his ad.
6. A winning bidder pays only an amount equal to the next highest bid.

1, 2, 3: We say that the current type of a bidder at some time during the run of the algorithm is j if he has spent between $(j-1)/k$ and j/k fraction of his budget at that time.

The algorithm allocates the next query to the bidder who maximizes the product of his bid and ψ(current type).

The proof of the competitive ratio changes minimally: Let the budget of bidder j be B_j. For $i = 1, .., k$, define β_i^j to be the amount of money spent by the bidder j from the interval $[\frac{i-1}{k}B_j, \frac{i}{k}B_j)$ of his budget. Let $\beta_i = \sum_j \beta_i^j$. Let α_i be the amount of money that the optimal allocation gets from the bins of final type i. Let $\alpha = \sum_i \alpha_i$, be the total amount of money obtained in the optimal allocation.

Now the relations used in Section 2.6 become

$$\forall i : \quad \beta_i \geq \frac{\alpha - \sum_{j=1}^{i} \alpha_j}{k}$$

$$\sum_i \psi(i)\alpha_i \ \leq \ \sum_i \psi(i)\beta_i$$

These two sets of equations suffice to prove that the competitive ratio is at least $1-1/e$. We also note that the algorithm and the proof of the competitive ratio remain unchanged even if we allow advertisers to enter the bidding process at any time during the query sequence.

4: If the arriving query q requires n_q number of advertisements to be placed, then allocate it to the bidders with the top n_q values of the product of bid and ψ(current type). The proof of the competitive ratio remains unchanged.

5: In order to model this situation, we simply set the effective bid of a bidder to be the product of his actual bid and his click-through rate (CTR), which is the probability that a user will click on his ad. We assume that the click-through rate is known to the algorithm in advance - indeed several search engines keep a measure of the click-through rates of the bidders.

6: So far we have assumed that a bidder is charged the value of his bid if he is awarded a query. Search engine companies charge a lower amount: the next highest bid. There are two ways of defining "next highest bid": next highest bid for this query among all bids received at the start of the algorithm or only among alive bidders, i.e. bidders who still have money.

It is easy to see that a small modification of our algorithm achieves a competitive ratio of $1 - 1/e$ for the first possibility: award the query to the bidder that maximizes next highest bid $\times\, \psi$(fraction of money spent). Next, let us consider the second possibility. In this case, the offline algorithm will attempt to keep alive bidders simply to charge other bidders higher amounts. If the online algorithm is also allowed this capability, it can also keep all bidders alive all the way to the end and this possibility reduces to the first one.

2.8 A Randomized Algorithm

In this section we define a generalization of the RANKING algorithm of [33], which has a competitive ratio of $1 - 1/e$ for arbitrary bids, when the bid to budget ratio is small.

In this algorithm we pick a random permutation σ of the n bidders right at the beginning. For a bidder i, we call $\sigma(i)$ the position or rank bidder i in σ. Again, we choose the same tradeoff function to trade off the importance of the bid of a bidder and his rank in the permutation:

$$\psi(i) = 1 - \left(1 - \frac{1}{n}\right)^{n-i+1}$$

Algorithm 2:

1. Pick a random permutation σ of the bidders.
2. For each new query, let the bid of bidder i be $b(i)$.
3. Allocate this query to a bidder with the highest value of the product $b(i) \times \psi(\sigma(i))$.

2.8.1 Analysis of Algorithm 2

In this section we prove that the competitive ratio of Algorithm 2 is also $1 - 1/e$. We follow the direct proof of Section 2.6.

We first define the notion of a Refusal algorithm based on Algorithm 2, which will *disallocate* certain money from the bidders as follows. Refusal will run identically to Algorithm 2, with the following difference: Consider a query q which arrives in the online order. Let r_q be the bidder to whom OPT allocated q, and let opt_q be the amount of money that OPT gets for q. Suppose that r_q has at least opt_q remaining budget when q arrives. Suppose further, that Refusal matches q to some bidder other than r_q (since this bidder has a higher product of bid and ψ-value). Then Refusal will disallocate opt_q money from r_q, i.e. it will artificially reduce the remaining budget of r_q by an amount opt_q.

Let b_{max} be the maximum bid value for any query from any bidder.

Lemma 10 *For any bidder, the amount of money which is not disallocated and not spent is at most b_{max}.*

Proof : Consider any bidder i and consider the queries that the optimum allocation allocates to i. The sum of the money spent by i in the optimal allocation is exactly the budget of i by assumption. For each such query q, let opt_q be the revenue of OPT on query q. When q enters during the algorithm, either it is allocated to i at the price of opt_q, or it is allocated to some other bidder. In the latter case, if i had opt_q amount of money remaining at that time, then opt_q amount is disallocated from i. Otherwise, the money remaining with i is less than $opt_q \leq b_{max}$. □

Lemma 11 *The competitive ratio of Refusal is at most the competitive ratio of Algorithm 2.*

Proof : By induction on the arrival of queries it is easy to see that the amount of money left with each bidder in Refusal is at most the amount of money with that bidder in Algorithm 2. □

We will now prove that the competitive ratio of Refusal is at least $1 - 1/e$.

Fix a query q and a permutation σ of the rows. Let r_q be the bidder to which OPT allocates q and let opt_q be the amount of money that OPT gets for q.

If Refusal matches q to r_q, then define $\alpha(q, \sigma) = n + 1$. Otherwise, we define $\alpha(q, \sigma)$ as follows: Let $A(q, \sigma)$ be the position in σ of the bidder to which Refusal matches q. Modify σ by shifting r_q upwards in the order, keeping the order of the rest of the bidders unchanged. Define $\alpha(q, \sigma)$ as the highest such position of r_q so that r_q has at least opt_q remaining budget when q arrives, and Refusal still matches q to the bidder in position $A(q, \sigma)$.

Define $x_i^q = opt_q \, Pr[\alpha(q,\sigma) = i]$, where the probability is taken over random σ. Let $x_i = \sum_q x_i^q$.

Define w_i^q to be the expected amount of money spent by the row in position i on query q. Let $w_i = \sum_q w_i^q$, the expected amount of money spent by the row in position i at the end of Refusal.

Lemma 12

$$\sum_i \psi(i) x_i \;\leq\; \sum_i \psi(i) w_i \tag{3}$$

Proof : Fix a query q and a permutation σ. Let r_q be the bidder to which OPT allocates q and let opt_q be the amount of money OPT gets for q. Let $A(q,\sigma)$ be the position in σ of the bidder to which Refusal matches q and let alg_q be the amount of money Refusal gets for q.

In the case that $\alpha(q,\sigma) \neq n+1$, the following holds by the rule used by the algorithm:

$$\psi(\alpha(q,\sigma)) opt_q \;\leq\; \psi(A(q,\sigma)) alg_q$$

In the case that $\alpha(q,\sigma) = n+1$, we simply write:

$$0 \;\leq\; \psi(A(q,\sigma)) alg_q$$

Taking expectation over random σ we get

$$\sum_i \psi(i) x_i^q \;\leq\; \sum_i \psi(i) w_i^q$$

Taking a summation over all queries q, we get

$$\sum_i \psi(i) x_i \;\leq\; \sum_i \psi(i) w_i$$

□

Lemma 13

$$\forall i: \quad w_i \;\geq\; 1 - \frac{\sum_{j=1}^{i} x_j}{n} - b_{max} \tag{4}$$

Proof : By Lemma 10, it is equivalent to prove that the expected amount of money disallocated in position i by Refusal is at most $\frac{\sum_{j=1}^{i} x_j}{n}$.

For a fixed query q and permutation σ, let r_q be the row to which OPT allocates q, and let $B(q,\sigma)$ be the position of r_q in σ. Then an opt_q amount of money is disallocated from r_q if and only if $\alpha(q,\sigma) \leq B(q,\sigma)$. In such a case, consider the following process. Start with a permutation derived from σ by shifting r_q to position $\alpha(q,\sigma)$. Replace r_q uniformly at random in each of the n positions. Then with probability $1/n$ we get back σ and opt_q amount of money is disallocated from r_q in position $B(q,\sigma)$. In this manner, we may only be overcounting the amount of disallocated money, since some of the positions for r_q below $\alpha(q,\sigma)$ may correspond to permutations σ' with a different (larger) value of $\alpha(q,\sigma')$.

Taking expectation over random σ and summing over all queries q, we get the statement of the lemma. □

Comparing to Section 2.6, we see that the constraints (3) and (4) are similar to the constraints obtained in that proof. The amount of money left unspent also has the same form, namely

$$\sum_{i=1}^{n}(1-w_i) \;\leq\; \sum_{i=1}^{n} x_i \frac{n-i+1}{n}$$

Hence we get

Proposition 14 *The competitive ratio of Refusal is at least* $1-1/e$.

From Lemma 11, we get:

Theorem 15 *The competitive ratio of the Algorithm 2 is at least* $1-1/e$.

Remark: Lemma 10 points to the reason why we assume that the largest bid is small compared to the budgets. Our analysis loses an amount of b_{max} due to fence-post errors, and it would be interesting to tighten the analysis and remove the assumption that the budget is large compared to the bids.

2.9 A Lower Bound for Randomized Algorithms

In [33], the authors proved a lower bound of $1-1/e$ on the competitive ratio of any randomized online algorithm for the online bipartite matching problem. Also, [32] proved a lower bound of $1 - 1/e$ on the competitive ratio of any online deterministic algorithm for the online b-matching problem, even for large b. We show a lower bound of $1 - 1/e$ for for online randomized algorithms for the b-matching problem, even for large b. This also resolves an open question from [31].

Theorem 16 *No randomized online algorithm can have a competitive ratio better than $1 - 1/e$ for the b-matching problem, for large b.*

Proof : We use Yao's Lemma [76], which says that the worst case expected factor (over all inputs) of the best randomized algorithm is equal to the expected factor of the best deterministic algorithm for the worst distribution over inputs. Thus it suffices for our purpose to present a distribution over inputs such that any deterministic algorithm obtains at most $1 - 1/e$ of the optimal allocation on the average. By Yao's Lemma, this would put a bound of $1 - 1/e$ on the worst case performance of any randomized algorithm.

Consider first the worst case input for the algorithm BALANCE with N bidders, each with a budget of 1. In this instance, the queries enter in N rounds. There are $1/\epsilon$ number of queries in each round. We denote by Q_i the queries of round i, which are identical to each other. For every $i = 1, .., N$, bidders i through N bid ϵ for each of the queries of round i, while bidders 1 through $i - 1$ bid 0 for these queries. The optimal assignment is clearly the one in which all the queries of round i are allocated to bidder i, achieving a revenue of N. One can also show that BALANCE will achieve only $N(1 - 1/e)$ revenue on this input.

Now consider all the inputs which can be derived from the above input by permutation of the numbers of the bidders and take the uniform distribution $\mathcal{D}$ over all these inputs. Formally, $\mathcal{D}$ can be described as follows: Pick a random permutation π of the bidders. The queries enter in rounds in the order $Q_1, Q_2, ..., Q_N$. Bidders $\pi(i), \pi(i+1), ..., \pi(N)$ bid ϵ for the queries Q_i and the other bidders bid 0 for these queries. The optimal allocation for any permutation π remains N, by allocating the queries Q_i to bidder $\pi(i)$. We wish to bound the expected revenue of any deterministic algorithm over inputs from the distribution $\mathcal{D}$.

Fix any deterministic algorithm. Let q_{ij} be the fraction of queries from Q_i that bidder j is allocated. We have:

$$E_\pi[q_{ij}] \leq \begin{cases} \frac{1}{N-i+1} & \text{if } j \geq i, \\ 0 & \text{if } j < i. \end{cases}$$

To see this, note that there are $N - i + 1$ bidders who are bidding for queries Q_i. The deterministic algorithm allocates some fraction of these queries to some bidders who bid for them, and leaves the rest of the queries unallocated. If $j \geq i$ then bidder j is a random bidder among the bidders bidding for these queries and hence is allocated an average amount of $\frac{1}{N-i+1}$ of the queries which were allocated from Q_i (where the average is taken over random permutations of the bidders). On the other hand, if $j < i$, then bidder j bids 0 for queries in Q_i and is not allocated any of these queries in any permutation.

Thus we get that the expected amount of money spent by a bidder j at the end of the algorithm is at most $\min\{1, \sum_{i=1}^{j} \frac{1}{N-i+1}\}$. By summing this over $j = 1, .., N$, we get that the expected revenue of the deterministic algorithm over the distributional input $\mathcal{D}$ is at most $N(1 - 1/e)$. This finishes the proof of the theorem. □

2.10 Discussion

Search engine companies accumulate vast amounts of statistical information which they do use in solving the Adwords problem. The main new idea coming from our study of this problem from the viewpoint of worst case analysis is the use of a tradeoff function. Blending this idea into the the currently used algorithms seems a promising possibility. One concrete way of accomplishing this is by applying the algorithm for learning from expert advice [18] to fine tuning the tradeoff function. For example, the tradeoff function should ideally be relaxed towards the end of the day. This can be accomplished by consider a set of "experts" each of whom changes the tradeoff function according to a fixed schedule over the course of the day. The learning algorithm is used over successive days to track the best such daily variation in tradeoff function.

Our first algorithm needs to keep track of the money spent by each advertiser, but the second one does not and is therefore useful if the search engine company is using a distributed set of servers which periodically coordinate the money spent by each advertiser.

Several new issues arise: Our notion of tradeoff revealing family of LP's deserves to be studied further in the setting of approximation and online algorithms. Is it possible to achieve competitive ratio of $1 - 1/e$ when the budgets of advertisers are not necessarily large? As stated in the Introduction, both our algorithms assume that daily budgets are large compared to bids. It is worth noting that an online algorithm for this problem with a competitive ratio of $1 - 1/e$ will not only match the lower bound given in [33] for online algorithms but also the best known off-line approximation algorithm [2].

Finally, this new auctions setting seems ripe with new game theoretic issues. For example, some of the search engines (e.g., Google) use a mechanism similar to a second-price auction for charging the advertisers in order to achieve some degree of incentive compatibility. However, it seems that there are still various ways for the advertisers to game these mechanisms. Designing a truthful mechanism in this setting is an important open problem. Recently, [11] provided a partial answer for this problem by showing that under some assumptions, it is impossible to design a truthful mechanism that allocates all the keywords to budget constrained advertisers.

Counterexamples for the simpler algorithms

2.11 Chapter Appendix: Counterexample 1

We present an example to show a factor strictly less than $1 - 1/e$ for the algorithm which gives a query to a highest bidder, breaking ties by giving it to the bidder with most left-over money. This example has only three values for the bids - 0, a or $2a$, for some small $a > 0$. Thus, in the case of arbitrary bids, the strategy of bucketing close enough bids (say within a factor of 2) together, and running such an algorithm does not work.

There are N bidders numbered $1, \ldots, N$, each with budget 1. We get the following query sequence and bidding pattern. Each bid is either 0, a or $2a$. Let $m = 1/a$. We will take $a \to 0$.

The queries arrive in N rounds. In each round m queries are made. The N rounds are

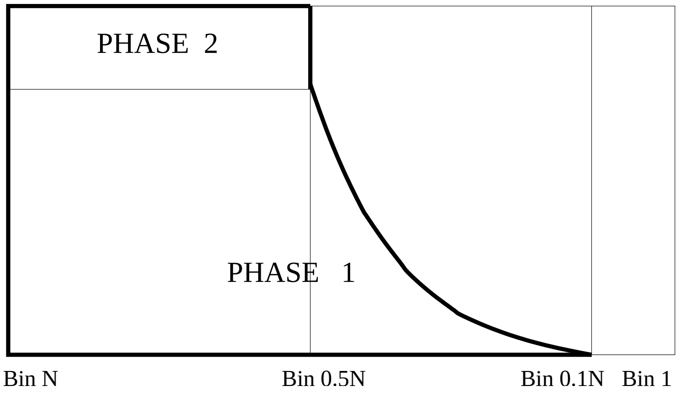

Figure 2: Counterexample 1: The bidders are ordered from right to left. The area inside the dark outline is the amount of money generated by the algorithm. The optimum allocation gets an amount equal to the whole rectangle.

divided into 3 phases.

Phase 1 **($1 \leq i \leq 0.4N$):** In the first round m queries are made, for which the bidders $0.1N + 1$ to N bid with a bid of a, and bidders 1 to $0.1N$ do not bid. Similarly, for $1 \leq i \leq 0.4N$, in the ith round m queries are made, for which bidders $0.1N + i$ to N bid with a bid of a, and for which bidders 1 to $0.1N + i - 1$ do not bid.

For $1 \leq i \leq 0.4N$, the algorithm will distribute the queries of the ith round equally between bidders $0.1N + i$ to N. This will give the partial allocation as shown in Figure 2.

Phase 2 **($0.4N + 1 \leq i \leq 0.5N$):** In the $(0.4N + 1)$th round m queries are made, for which bidder 1 bids a, and bidders $0.5N$ to N bid $2a$ (the rest of the bidders bid 0). Similarly, for $0.4N + 1 \leq i \leq 0.5N$, in the ith round m queries are made, for which bidder $i - 0.4N$ bids a, and bidders $0.5N$ to N bid $2a$.

For $0.4N + 1 \leq i \leq 0.5N$, the algorithm will distribute the queries of round i equally between bidders $0.5N$ to N.

At this point during the algorithm, bidders $0.5N + 1$ to N have spent all their money.

Phase 3 **($0.5N + 1 \leq i \leq N$):** m queries enter in round i, for which only bidder i bids at a, and the other bidders do not bid.

The algorithm has to throw away these queries, since bidders $0.5N+1$ to N have already spent their money.

The optimum allocation, on the other hand, is to allocate the queries in round i as follows:

- For $1 \leq i \leq 0.4N$, allocate all queries in round i to bidder $0.1N + i$.
- For $0.4N + 1 \leq i \leq 0.5N$, allocate all queries in round i to bidder $i - 0.4N$.
- For $0.5N + 1 \leq i \leq N$, allocate all queries in round i to bidder i.

Clearly, OPT makes N amount of money. A calculation shows that the algorithm makes $0.62N$ amount of money. Thus the factor is strictly less than $1 - 1/e$.

We can modify the above example to allow bids of 0, a and κa, for any $\kappa > 1$, such that the algorithm performs strictly worse that $1 - 1/e$.

As $\kappa \to \infty$, the factor tends to $1-1/e$, and as $\kappa \to 1$, the factor tends to $1/2$. Of course, if $\kappa = 1$, then this reduces to the original model of [32], and the factor is $1 - 1/e$.

2.12 Chapter Appendix: Counterexample 2

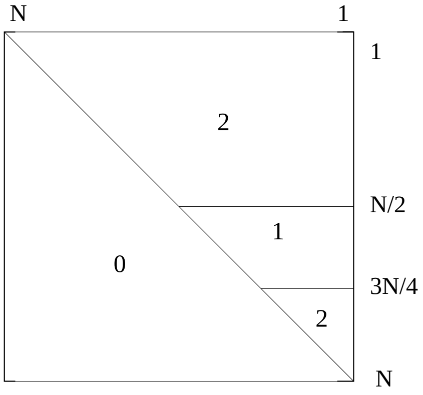

Figure 3: Counterexample 2: The rows N/2 to 3N/4 have a budget of 1. All other rows have a budget of 2.

The example consists of an $N \times N$ upper-triangular matrix, as shown in Figure 3. The columns represent the queries and are ordered from right to left. The rows represent the bidders. The entry in the ith row and jth column is the bid of bidder i for query j. The entries in the upper triangle in rows 1 to $N/2$, and rows $3N/4$ to N are all 2. These bidders have a budget of 2. The entries in the upper triangle in rows $N/2+1$ to $3N/4$ are all 1. These bidders have a budget of 1.

The optimum allocation is along the diagonal, with column i allocated to row i. This generates $2N - N/4$ amount of money.

It can be proved that the algorithm gets $1.1N$ amount of money, which is strictly less than $1 - 1/e$ of the optimum.

CHAPTER III

COMBINATORIAL AUCTIONS UNDER GENERAL SUBMODULAR UTILITIES

In this chapter we study a generalization of the problem considered in Chapter 2, although in the offline setting. Considering the advertisers of Chapter 2, we note that their utility functions (as a function of the set of queries they get allocated) are additive with an upper bound of their budget. Such a utility function is immediately seen to be monotone and submodular. Here we consider the generalization of the allocation problem to general submodular utilities.

We consider the following allocation problem arising in the setting of combinatorial auctions: a set of goods is to be allocated to a set of players so as to maximize the sum of the utilities of the players (i.e., the social welfare). The utility of each player is a monotone submodular function. We prove that there is no polynomial time approximation algorithm which approximates the maximum social welfare by a factor better than $1 - 1/e \simeq 0.632$, unless **P**= **NP**.

One motivation for studying this problem comes from the attempt to design a truth-revealing auction under this setting. If we could obtain the optimal allocation then the Vickrey-Clarke-Groves mechanism would immediately give a truth-revealing auction. However, by the results of this chapter, we cannot find the optimum or even $1 - 1/e$ of the optimum in polynomial time (unless **P**=**NP**). This leads to an even more interesting question which we leave open - how to design polynomial time truth-revealing auctions which well-approximate the social optimum.

3.1 Introduction: Optimization in Combinatorial Auctions

Auctions are becoming a very popular method of transaction, from auction services on the Internet (e.g. eBay), to large scale transactions (e.g. the FCC auctions of spectrum licenses). Recently, there has been a lot of interest in auctions with complex bidding and allocation possibilities that can capture various dependencies between a large number of items being sold. A very general model which can express such complex scenarios is that of combinatorial auctions.

In a combinatorial auction, a set of goods is to be sold to a set of players. A utility function is associated with each player specifying the happiness of the player for each subset of the goods. One natural objective for the auctioneer is to maximize the economic efficiency of the auction, which is the sum of the utilities of all the players. Formally, we have a set M of m indivisible goods and n players. Player i has a monotone utility function $v_i : 2^M \rightarrow \mathrm{R}$. The *allocation problem* is to find a partition $(S_1, \ldots, S_n)$ of the set of goods among the n players that maximizes the total utility or *social welfare*, $\sum_i v_i(S_i)$. Such an allocation is called an optimal allocation.

We are interested in the computational complexity of the allocation problem. We would like an algorithm which runs in time polynomial in n and m. However, one can see that the input representation is itself exponential in m for general utility functions. Even if the utility functions have a succinct representation (polynomial in n and m), the allocation problem may be **NP**-hard [43, 3]. In the absence of a succinct representation, it is typically assumed that the auctioneer has oracle access to the players' utilities and that he can ask queries to the players. There are 2 types of queries that have been considered. In a *value query* the auctioneer specifies a subset $S \subseteq M$ and asks player i for the value $v_i(S)$. In a *demand query*, the auctioneer presents a set of prices for the goods and asks a player for the set S of goods that maximizes his profit (which is his utility for S minus the sum of the prices of the goods in S). Note that if we have a succinct representation of the utility functions then we can always simulate value queries. Even with queries the problem remains hard. Hence we are interested in approximation algorithms and inapproximability results.

A natural class of utility functions that has been studied extensively in the literature is the class of submodular functions. A function v is submodular if for any 2 sets of goods $S \subseteq T$, the marginal contribution of a good $x \notin T$, is bigger when added to S than when added to T, i.e., $v(S \cup x) - v(S) \geq v(T \cup x) - v(T)$. Submodularity can be seen as the discrete analog of concavity and arises naturally in economic settings since it captures the property that marginal utilities are decreasing as we allocate more goods to a player. It is known that the class of submodular utility functions contains the functions with the Gross Substitutes property [24], and also that submodular functions are complement-free.

3.1.1 Previous Work

For general utility functions, the allocation problem is **NP**-hard. Approximation algorithms have been obtained that achieve factors of $O(\frac{1}{\sqrt{m}})$ ([44, 10], using demand queries) and $O(\frac{\sqrt{\log m}}{m})$ ([10], using value queries). It has also been shown that we cannot have polynomial time algorithms with a factor better than $O(\frac{\log m}{m})$ ([10], using value queries) or better than $O(\frac{1}{m^{1/2-\epsilon}})$ ([44, 70], even for single minded bidders). Even if we allow demand queries, exponential communication is required to achieve any approximation guarantee better than $O(\frac{1}{m^{1/2-\epsilon}})$ [58]. For single-minded bidders, as well as for other classes of utility functions, approximation algorithms have been obtained, among others, in [4, 9, 44]. For more results on the allocation problem see [13].

For the class of submodular utility functions, the allocation problem is still **NP**-hard. The following positive results are known: In [43] it was shown that a simple greedy algorithm using value queries achieves an approximation ratio of 1/2. An improved ratio of $1 - 1/e$ was obtained in [3] for a natural special case of submodular functions, the class of additive valuations with budget constraints. Very recently, an approximation algorithm with ratio $1 - 1/e$ was obtained in [15] using demand queries in the case when all players have the same submodular utility function. As for negative results, it was shown in [58] that an exponential amount of communication is needed to achieve an approximation ratio better than $1 - O(\frac{1}{m})$. In [15] it was shown that there cannot be any polynomial time algorithm in the succinct representation or the value query model with a ratio better than 50/51, unless **P**= **NP**. In [15] a hardness result of $1 - 1/e$ is proved for the class of XOS utilities. This class strictly contains the class of submodular utilities (for a definition of the class XOS see [43]).

3.1.2 Our Result

We show that there is no polynomial time approximation algorithm for the allocation problem with monotone submodular utility functions achieving a ratio better than $1-1/e$, unless **P= NP**. Our result is true in the succinct representation model, and hence also in the value query model. The result does not hold if the algorithm is allowed to use demand queries.

A hardness result of $1-1/e$ for the larger class XOS is obtained in [15] by a gadget reduction from the maximum k-coverage problem. Similar reductions do not seem to work for submodular functions. Instead we provide a reduction from multi-prover proof systems for MAX-3-COLORING. Our result is based on the reduction of Feige [16] for the hardness of set-cover and maximum k-coverage. The results of [16] use a reduction from a multi-prover proof system for MAX-3-SAT. This is not sufficient to give a hardness result for the allocation problem, as explained in Section 3.3. Instead, we use a proof system for MAX-3-COLORING. We then define an instance of the allocation problem and show that the new proof system enables all players to achieve maximum possible utility in the yes case, and ensure that in the no case, players achieve only $(1-1/e)$ of the maximum utility on the average. The crucial property of the new proof system is that when a graph is 3-colorable, there are in fact many different proofs (i.e. colorings) that make the verifier accept. By introducing a correspondence between colorings and players of the allocation instance, we obtain the desired result.

We note that we do not address the question of obtaining truthful mechanisms for the allocation problem. For some classes of functions, incentive compatible mechanisms have been obtained that also achieve reasonable approximations to the allocation problem (e.g. [44, 4, 9]). For submodular utilities, the only truthful mechanism known is obtained in [15]. This achieves an $O(\frac{1}{\sqrt{m}})$-approximation. Obtaining a truthful mechanism with a better approximation guarantee seems to be a challenging open problem.

The rest of the chapter is organized as follows: In the next section we define the model formally and introduce some notation. In Section 3.3 we first present a weaker hardness result of 3/4 using a 2-prover proof system to illustrate the ideas in our proof. In Section 3.4 we present the hardness of $1-1/e$ based on the k-prover proof system of [16].

3.2 Model, Definitions and Notation

We assume we have a set of players $N = \{1, ..., n\}$ and a set of goods $M = \{1, ..., m\}$ to be allocated to the players. Each player has a utility function v_i, where for a set $S \subseteq M$, $v_i(S)$ is the utility that player i derives if he obtains the set S. We make the standard assumptions that v_i is monotone and that $v_i(\emptyset) = 0$.

Definition 1 *A function $v : 2^M \rightarrow R$ is submodular if for any sets $S \subset T$ and any $x \in M \backslash T$:*

$$v(S \cup \{x\}) - v(S) \geq v(T \cup \{x\}) - v(T)$$

An equivalent definition of submodular functions is that for any sets S, T: $v(S \cup T) + v(S \cap T) \leq v(S) + v(T)$.

An allocation of M is a partition of the goods $(S_1, ..., S_n)$ such that $\bigcup_i S_i = M$ and $S_i \cap S_j = \emptyset$. The allocation problem we will consider is: Given a monotone, submodular utility function v_i for every player i, find an allocation of the goods $(S_1, ..., S_n)$ that maximizes $\sum_i v_i(S_i)$. To clarify how the input is accessed, we assume that either the utility functions have a succinct representation[1], or that the auctioneer can ask value queries to the players. In a value query, the auctioneer specifies a subset S to a player i and the player responds with $v_i(S)$. In this case the auctioneer is allowed to ask at most a polynomial number of value queries.

Since the allocation problem is **NP**-hard, we are interested in polynomial time approximation algorithms: an algorithm achieves an approximation ratio of $\alpha \leq 1$ if for every instance of the problem, the algorithm returns an allocation with social welfare at least α times the optimal social welfare.

3.3 A Hardness of 3/4

We first present a hardness result of 3/4. The reduction of this section is based on a 2-prover proof system for MAX-3-COLORING, which is analogous to the proof system of [49] for

[1] By this we mean a representation of size polynomial in n and m, such that given S and i, the auctioneer can compute $v_i(S)$ in time polynomial in the size of the representation. For example, additive valuations with budget limits [43] have a succinct representation.

MAX-3-SAT. We remark that this proof is provided here only to illustrate the main ideas of our result and to give some intuition. In the next Section we will see that by moving to a k-prover proof system we can obtain hardness of $1 - 1/e$.

In the MAX-3-COLORING problem, we are given a graph G and we are asked to color the vertices of G with 3 different colors so as to maximize the number of properly colored edges, where an edge is properly colored if its vertices receive different colors. Given a graph G, let $OPT(G)$ denote the maximum fraction of edges that can be properly colored by any 3-coloring of the vertices. We will start with a *gap* version of MAX-3-COLORING: Given a constant c, we denote by GAP-MAX-3-COLORING(c) the promise problem in which the yes instances are the graphs with $OPT(G) = 1$ and the no instances are graphs with $OPT(G) \leq c$. By the PCP theorem [6], and by [59], we know:

Proposition 17 *There is a constant* $c < 1$ *such that GAP-MAX-3-COLORING(c) is* **NP**-*hard, i.e., it is* **NP**-*hard to distinguish whether*
YES Case: $OPT(G) = 1$, *and*
NO Case: $OPT(G) \leq c$.

Let G be an instance of GAP-MAX-3-COLORING(c). The first step in our proof is a reduction to the Label Cover problem. A label cover instance L consists of a graph G', a set of labels Λ and a binary relation $\pi_e \subseteq \Lambda \times \Lambda$ for every edge e. The relation π_e can be seen as a constraint on the labels of the vertices of e. An assignment of one label to each vertex is called a *labeling*. Given a labeling, we will say that the constraint of an edge $e = (u, v)$ is satisfied if $(l(u), l(v)) \in \pi_e$, where $l(u), l(v)$ are the labels of u, v respectively. The goal is to find a labeling of the vertices that satisfies the maximum fraction of edges of G', denoted by $OPT(L)$.

The instance L produced from G is the following: G' has one vertex for every edge (u, v) of G. The vertices (u_1, v_1) and (u_2, v_2) of G' are adjacent if and only if the edges (u_1, v_1) and (u_2, v_2) have one common vertex in G. Each vertex (u, v) of G' has 6 labels corresponding to the 6 different proper colorings of (u, v) using 3 colors. For an edge between (u_1, v_1) and (u_2, v_2) in G', the corresponding constraint is satisfied if the labels of (u_1, v_1) and (u_2, v_2)

assign the same color to their common vertex. From Proposition 17 it follows easily that:

Proposition 18 *It is* **NP***-hard to distinguish between:*
YES Case: $OPT(L) = 1$*, and*
NO Case: $OPT(L) \leq c'$*, for some constant* $c' < 1$

We will say that 2 labelings L_1, L_2 are *disjoint* if every vertex of G' receives a different label in L_1 and L_2. Note that in the YES case, there are in fact 6 disjoint labelings that satisfy all the constraints.

The Label Cover instance L is essentially a description of a 2-prover 1-round proof system for MAX-3-COLORING with completeness parameter equal to 1 and soundness parameter equal to c' (see [16, 49] for more on these proof systems).

Given L, we will now define a new label cover instance L', the hardness of which will imply hardness of the allocation problem. Going from instance L to L' is equivalent to applying the parallel repetition theorem of Raz [63] to the 2-prover proof system for MAX-3-COLORING.

We will denote by H the graph in the new label cover instance L'. A vertex of H is now an ordered tuple of s vertices of G', i.e., it is an ordered tuple of s edges of G, where s is a constant to be determined later . We will refer to the vertices of H as nodes to distinguish them from the vertices of G. For 2 nodes of H, $u = (e_1, ..., e_s)$ and $v = (e'_1, ..., e'_s)$, there is an edge between u and v if and only if for every $i \in [s]$, the edges e_i and e'_i have exactly one common vertex (where $[s] = \{1, ..., s\}$). We will refer to these s common vertices as the *shared* vertices of u and v. The set of labels of a node $v = (e_1, ..., e_s)$ is the set of 6^s proper colorings of its edges ($\Lambda = [6^s]$). The constraints can be defined analogously to the constraints in L. In particular, for an edge $e = (u, v)$ of H, a labeling satisfies the constraint of edge e if the labels of u and v induce the same coloring of their shared vertices.

By Proposition 18 and Raz's parallel repetition theorem [63], we have:

Proposition 19 *It is* **NP***-hard to distinguish between:*
YES Case: $OPT(L') = 1$*, and*
NO Case: $OPT(L') \leq 2^{-\gamma s}$*, for some constant* $\gamma > 0$*.*

Remark 1 *In fact, in the YES case, there are 6^s disjoint labelings that satisfy all the constraints.*

This property will be used crucially in the remaining section. The known reductions from GAP-MAX-3-SAT to label cover, implicit in [16, 49], are not sufficient to guarantee that there is more than one labeling satisfying all the edges. This was our motivation for using GAP-MAX-3-COLORING instead.

The final step of the proof is to define an instance of the allocation problem from L'. For that we need the following definition:

Definition 2 *A Partition System $P(U, r, h, t)$ consists of a universe U of r elements, and t pairs of sets $(A_1, \bar{A}_1), ...(A_t, \bar{A}_t)$, ($A_i \subset U$) with the property that any collection of $h' \leq h$ sets without a complementary pair $A_i, \bar{A}_i$ covers at most $(1 - 1/2^{h'})r$ elements.*

If $U = \{0, 1\}^t$, we can construct a partition system $P(U, r, h, t)$ with $r = 2^h$ and $h = t$. For $i = 1, ..., t$ the pair $(A_i, \bar{A}_i)$ will be the partition of U according to the value of each element in the i-th coordinate. In this case the sets $A_i, \bar{A}_i$ have cardinality $r/2$.

For every edge e in the label cover instance L', we construct a partition system $P^e(U^e, r, h, t = h = 3^s)$ on a separate subuniverse U^e as described above. Call the partitions $(A^e_1, \bar{A}^e_1)$,...., $(A^e_t, \bar{A}^e_t)$.

Recall that for every edge $e = (u, v)$, there are 3^s different colorings of the s shared vertices of u and v. Assuming some arbitrary ordering of these colorings, we will say that the pair $(A^e_i, \bar{A}^e_i)$ of P^e corresponds to the ith coloring of the shared vertices.

We define a set system on the whole universe $\bigcup U^e$. For every node v and every label i, we have a set $S_{v,i}$. For every edge e incident on v, $S_{v,i}$ contains one set from every partition system P^e, as follows. Consider an edge $e = (v, w)$. Then A^e_j contributes to all the sets $S_{v,i}$ such that label i in node v induces the jth coloring of the shared vertices between v and w. Similarly $\bar{A}^e_j$ contributes to all the $S_{w,i}$ such that label i in node w gives the jth coloring to the shared vertices (the choice of assigning A^e_j to the $S_{v,i}$'s and $\bar{A}^e_j$ to the $S_{w,i}$'s is made arbitrarily for each edge (v, w)). Thus

$$S_{v,i} = \bigcup_{(v,w) \in E} B^{(v,w)}_j$$

where E is the set of edges of H, $B_j^{(v,w)}$ is one of $A_j^{(v,w)}$ or $\bar{A_j}^{(v,w)}$, and j is the coloring that label i gives to the shared vertices of (v,w).

We are now ready to define our instance I of the allocation problem. There are $n = 6^s$ players, all having the same utility function. The goods are the sets $S_{v,i}$ for each node v and label i. If a player is allocated a collection of goods $S_{v_1,i_1}...S_{v_l,i_l}$, then his utility is

$$|\bigcup_{j=1}^{l} S_{v_j,i_j}|$$

It is easy to verify that this is a monotone and submodular utility function. Let $OPT(I)$ be the optimal solution to the instance I.

Lemma 20 *If $OPT(L') = 1$, then $OPT(I) = nr|E|$.*

Proof : From Remark 1, there are $n = 6^s$ disjoint labelings that satisfy all the constraints of L'. Consider the ith such labeling. This defines an allocation to the ith player as follows: we allocate the goods $S_{v,l(v)}$ for each node v, to player i, where $l(v)$ is the label of v in this ith labeling. Since the labeling satisfies all the edges, the corresponding sets $S_{v,l(v)}$ cover all the subuniverses. To see this, fix an edge $e = (v,w)$. The labeling satisfies e, hence the labels of v and w induce the same coloring of the shared vertices between v and w, and therefore they both correspond to the same partition of P^e, say $(A_j^e, \bar{A}_j^e)$. Thus U^e is covered by the sets $S_{v,l(v)}$ and $S_{w,l(w)}$ and the utility of player i is $r|E|$. We can find such an allocation for every player, since the labelings are disjoint. □

For the No case, consider the following simple allocation: each player gets exactly one set from every node. Hence each player i defines a labeling, which cannot satisfy more than $2^{-\gamma s}$ fraction of the edges. For the rest of the edges, the 2 sets of player i come from different partitions and hence can cover at most 3/4 of the subuniverse. This shows that the total utility of this simple allocation is at most 3/4 of that in the Yes case. In fact, we will show that this is true for any allocation.

Lemma 21 *If $OPT(L') \leq 2^{-\gamma s}$, then $OPT(I) \leq (3/4 + \epsilon)nr|E|$, for some small constant $\epsilon > 0$ that depends on s.*

Proof : Consider an allocation of goods to the players. If player i receives sets $S_1, ..., S_l$, then his utility p_i can be decomposed as $p_i = \sum_e p_{i,e}$, where

$$p_{i,e} = |(\cup_j S_j) \cap U^e|$$

Fix an edge (u, v). Let m_i be the number of goods of the type $S_{u,j}$ for some j. Let m_i' be the number of goods of the type $S_{v,j}$ for some j, and let $x_i = m_i + m_i'$. Let N be the set of players. For the edge $e = (u, v)$, let N_1^e be the set of players whose sets cover the subuniverse U^e and $N_2^e = N \backslash N_1^e$. Let $n_1^e = |N_1^e|$ and $n_2^e = |N_2^e|$. Note that for $i \in N_1^e$, the contribution of the x_i sets to $p_{i,e}$ is r. For $i \in N_2^e$, it follows that the contribution of the x_i sets to $p_{i,e}$ is at most $(1 - \frac{1}{2^{x_i}})r$ by the properties of the partition system of this edge[2]. Hence the total utility derived by the players from the subuniverse U^e is

$$\sum_i p_{i,e} \leq \sum_{i \in N_1^e} r + \sum_{i \in N_2^e} (1 - \tfrac{1}{2^{x_i}})r$$

In the YES case, the total utility derived from the subuniverse U^e was nr. Hence the loss in the total utility derived from U^e is

$$\Delta_e \geq nr - \sum_{i \in N_1^e} r - \sum_{i \in N_2^e} (1 - \tfrac{1}{2^{x_i}})r = r \sum_{i \in N_2^e} \frac{1}{2^{x_i}}$$

By the convexity of the function 2^{-x}, we have that

$$\Delta_e \geq r \; n_2^e \; 2^{-\frac{\sum_{i \in N_2^e} x_i}{n_2^e}}$$

But note that $\sum_{i \in N_1^e} \geq 2n_1^e$, since players in N_1^e cover U^e and they need at least 2 sets to do this. Therefore $\sum_{i \in N_2^e} x_i \leq 2n_2^e$ and $\Delta_e \geq r \; n_2^e/4$. The total loss is

$$\sum_e \Delta_e \geq r/4 \sum_e n_2^e$$

Hence it suffices to prove $\sum_e n_2^e \geq (1 - \epsilon)n|E|$, or that $\sum_e n_1^e \leq \epsilon n|E|$.

For an edge (u, v), let $N_1^{e,\leq s}$ be the set of players from N_1^e who have at most s goods of the type $S_{u,j}$ or $S_{v,j}$. Let $N_1^{e,>s} = N_1^e \backslash N_1^{e,\leq s}$.

[2] To use the property of P^e, we need to ensure that $x_i \leq 3^s$. However since $i \in N_2^e$, even if $x_i > 3^s$, the distinct sets A_j^e or $\bar{A}_j^e$ that he has received through his x_i goods are all from different partitions of U_e and hence they can be at most 3^s.

$$\sum_e n_1^e = \sum_e |N_1^{e,>s}| + |N_1^{e,\leq s}| \leq \frac{2n|E|}{s} + \sum_e |N_1^{e,\leq s}|$$

where the inequality follows from the fact that for the edge e we cannot have more than $2n/s$ players receiving more than s goods from u and v.

Claim 22 $\sum_e |N_1^{e,\leq s}| < \delta n|E|$, *where* $\delta \leq c' s 2^{-\gamma s}$, *for some constant* c'.

Proof : Suppose that the sum is $\delta n|E|$ for some $\delta \leq 1$. Then it can be easily seen that for at least $\delta|E|/2$ edges, $|N_1^{e,\leq s}| \geq \delta n/2$. Call these edges *good* edges.

Pick a player i at random. For every node, consider the set of goods assigned to player i from this node, and pick one at random. Assign the corresponding label to this node. If player i has not been assigned any good from some node, then assign an arbitrary label to this node. This defines a labeling. We look at the expected number of satisfied edges.

For every good edge $e = (u, v)$, the probability that e is satisfied is at least $\delta/2s^2$, since e has at at least $\delta n/2$ players that have covered U^e with at most s goods. Since there are at least $\delta|E|/2$ good edges, the expected number of satisfied edges is at least $\delta^2|E|/4s^2$. This means that there exists a labeling that satisfies at least $\delta^2|E|/4s^2$ edges. But, since $OPT(L') \leq 2^{-\gamma s}$, we get $\delta \leq c' s 2^{-\gamma s}$, for some constant c'. □

We finally have

$$\sum_e n_1^e \leq \frac{2n|E|}{s} + \delta n|E| \leq \epsilon n|E|$$

where ϵ is some small constant depending on s. Therefore the total loss is

$$\sum_e \Delta_e \geq \frac{1}{4}(1-\epsilon) nr|E|$$

which implies that $OPT(I) \leq (3/4 + \epsilon) nr|E|$. □

Given any $\epsilon > 0$, we can choose s large enough so that Lemma 21 holds. From Lemmas 20 and 21, we have:

Theorem 23 *For any* $\epsilon > 0$*, there is no polynomial time* $(3/4+\epsilon)$*-approximation algorithm for the allocation problem with monotone submodular utilities, unless* $\mathbf{P} = \mathbf{NP}$.

3.4 A Hardness of 1-1/e

In this section we obtain a stronger result by using a k-prover proof system (i.e., a label cover problem on hypergraphs) for MAX-3-COLORING. The new proof system is obtained in a similar manner as the proof system for MAX-3-SAT by Feige [16].

Let G be an instance of GAP-MAX-3-COLORING(c). From the graph G, we will define a new label cover instance. The label cover instance is now defined on a hypergraph H instead of a graph. Let s and k be constants to be determined later. The hypergraph H consists of k layers of vertices, $V_1, ..., V_k$. To every layer V_i, we associate a binary string $C_i \in \{0,1\}^s$ of weight[3] $s/2$, with the property that the Hamming distance between any 2 strings is at least $s/3$. One can obtain such a collection of strings by using the codewords of an appropriate binary code (see [16] for more details). Notice that each C_i defines a partition of the indices $\{1, ..., s\}$ into 2 sets A_i, B_i, each of cardinality $s/2$, where an index l belongs to A_i (resp. B_i) if the l-th bit of C_i is 1 (resp. 0).

We will refer to the vertices of H as nodes. One difference from Section 3.3 is that now a node of H will contain both edges and vertices of G. To be more specific, a node v in V_i is an ordered s-tuple $v = (v^1, ..., v^s)$, where for $l \in \{1, ..., s\}$, if $l \in A_i$, then v_l is an edge of G, otherwise it is a vertex of G. Clearly there are at most $n^{O(s)}$ nodes in each layer V_i and since k and s are constants, the size of H is polynomial in the size of G.

A label of a node v in V_i will be a proper coloring of the $s/2$ edges corresponding to v and a coloring of the $s/2$ vertices corresponding to v. Hence there are $6^{s/2}3^{s/2}$ distinct labels. For technical reasons we make $6^{s/2}$ copies of each label so that in total we have 6^s labels in every node.

Edges of the hypergraph H have cardinality k and contain one node from each layer. For every ordered tuple of s edges $(e_1, ..., e_s)$, of G and a choice of s vertices $(u_1, ..., u_s)$, one from each e_i, we will have a hyperedge $(v_1, ..., v_k)$ in H, with $v_i \in V_i$ if and only if the nodes $v_1, ..., v_k$ satisfy the following:

1. $v_i^l = e_l$ if $l \in A_i$.

[3] The weight of a binary string is the number of 1's in it.

2. $v_i^l = u_l$ if $l \in B_i$.

We will call the vertices $u_1, ..., u_s$ the *shared* vertices of the hyperedge $(v_1, ..., v_k)$. Given a labeling to the nodes of H, let $(l(v_1), ..., l(v_k))$ be the labels of the hyperedge $e = (v_1, ..., v_k)$. We will say that e is *weakly* satisfied if there exists a pair of nodes v_i, v_j that agree on the coloring of the shared vertices as induced by their labeling. We will call the pair of labels $(l(v_i), l(v_j))$ a consistent pair with respect to the hyperedge e and the labeling. We will say that a hyperedge is *strongly* satisfied if for every pair v_i, v_j, $(l(v_i), l(v_j))$ is consistent. This completes the description of the label cover instance L. Let $OPT^{weak}(L)$ (resp. $OPT^{strong}(L)$) be the maximum fraction of hyperedges that can be weakly (resp. strongly) satisfied by any labeling. The following lemma is essentially Lemma 5 in [16].

Lemma 24 *It is* **NP***-hard to distinguish between:*
YES Case: $OPT^{strong}(L) = 1$
NO Case: $OPT^{weak}(L) \leq k^2 2^{-\gamma s}$*, for some constant* $\gamma > 0$.

Remark 2 *In the YES Case of Lemma 24, there are* 6^s *disjoint labelings that strongly satisfy all the hyperedges.*

This follows from a similar argument as for Remark 1.

To define the instance of the allocation problem, we will first construct a set system as in Section 3.3. For this we will need a more general notion of a partition system:

Lemma 25 ([16]) *Let* $U = [k]^n$. *We can construct a partition system on* U *of the form* $P = \{(A_1^1, ..., A_k^1), (A_1^2, ..., A_k^2), ..., (A_1^n, ..., A_k^n)\}$, *with the properties that*

1. *For* $i = 1, ..., n$, $\cup A_j^i = U$.
2. *Any collection of* $l \leq n$ *sets, one from each partition, covers exactly* $(1-(1-1/k)^l)|U|$ *elements.*

For every hyperedge e, we will have a separate subuniverse U^e. Let $n = 6^s$ be the number of labels of each node. For each hyperedge e we construct a partition system P^e on the subuniverse U^e as in Lemma 25. Let $P^e = \{(A_{1,1}^e, ..., A_{1,k}^e), (A_{2,1}^e, ..., A_{2,k}^e), ..., (A_{n,1}^e, ..., A_{n,k}^e)\}$.

Notice that for a hyperedge $e = (v_1, ..., v_k)$, we can always find n disjoint labelings of the nodes $v_1, ..., v_k$ that strongly satisfy the hyperedge e. This follows from the fact that there are 6^s proper colorings of an s-tuple of edges of G and for each such coloring we can pick a label from each node v_i that respects this coloring. Due to the multiple copies of each distinct label, we in fact have more than n labelings that strongly satisfy e. We arbitrarily pick n of these disjoint labelings (note that any other labeling that strongly satisfies e is "isomorphic" to one of the n labelings picked). Assuming some arbitrary ordering among the n labelings, we associate the jth partition of P^e with the jth labeling of e, for every e. If $(l_1^j, ..., l_k^j)$ is the jth labeling of e and $(A_{j,1}^e, ..., A_{j,k}^e)$ is the jth partition of P^e we will also say that the set $A_{j,i}^e$ corresponds to the label l_i^j of v_i.

We can now define our set system. We will have one set $S_{v,i}$ for every node v and label i. Let $v \in V_l$ for some $l \in [k]$. For an edge e that contains node v, suppose label i is in the jth labeling of e. We will then include the set $A_{j,l}^e$ from the jth partition in $S_{v,i}$. Hence $S_{v,i}$ is the following union of sets:

$$S_{v,i} = \bigcup_{e:v\sim e} A_{j_e(i),l}^e$$

where $j_e(i)$ is the labeling of edge e that contains i.

As in Section 3.3, the instance of the allocation problem contains $n = 6^s$ players with the same submodular utility function. The goods are the sets $S_{v,i}$ and the utility of a player for a collection of sets is the cardinality of their union. Let I denote the instance of the allocation problem and let $OPT(I)$ be the optimal solution of I. Let $r = |U^e|$ and let E be the set of the hyperedges of H.

Lemma 26 *If $OPT^{strong}(L) = 1$, then $OPT(I) = nr|E|$.*

Proof : Since $OPT^{strong}(L) = 1$, consider a labeling that strongly satisfies all the hyperedges. By the discussion above, we can always pick a labeling such that when restricted to the nodes of an edge, it corresponds to one of the n disjoint labelings of that edge. Let $l(v)$ be the label of each node. Pick a player and allocate to him all the sets $\{S_{v,l(v)}\}$. We claim that the sets cover the subuniverse U^e for every edge e and the utility of the player is therefore $r|E|$. To see this, fix an edge $e = (v_1, ..., v_k)$. Since the labeling strongly satisfies the edge, it corresponds to some partition of the partition system P^e, say the jth partition. Hence for $i = 1, ..., k$, the set $A^e_{j,i}$ which corresponds to label $l(v_i)$ is contained in $S_{v_i,l(v_i)}$. Thus the player covers the entire subuniverse U^e with the sets $S_{v_i,l(v_i)}$. Since this is true for every edge, his utility is exactly $r|E|$. By Remark 2 we can repeat the above for all the 6^s players.

□

Lemma 27 *If $OPT^{weak}(L) \leq k^2 2^{-\gamma s}$, then $OPT(I) \leq (1 - 1/e + \epsilon)nr|E|$, where $\epsilon > 0$ is some small constant depending on s and k.*

Proof : Consider an allocation of the goods to the players, i.e., an allocation of the labels of each node. We decompose the utility p_i of player i as: $p_i = \sum p_{i,e}$, where $p_{i,e}$ is as in Section 3.3. For a node v and a player i, let m^v_i be the number of sets of the type $S_{v,j}$ that player i has received. Fix an edge $e = (v_1, ..., v_k)$. Let $x^e_i = \sum_{l=1}^{k} m^{v_l}_i$. Define the set of players:

$$N^e_1 = \{i : \exists v_j, v_l \text{ such that } i \text{ has a pair of consistent labels for these 2 nodes}\}$$

Let $N^e_2 = N \backslash N^e_1$, and let $n^e_1 = |N^e_1|, n^e_2 = |N^e_2|$. Trivially, for $i \in N^e_1$, the contribution of the x^e_i sets to $p_{i,e}$ is at most r. For $i \in N^e_2$, the x^e_i sets of the type $S_{v_l,j}$ do not contain even one pair of labels which are consistent for some pair of nodes in e. For each set $S_{v_l,j}$ that player i has received, let $A^e_{t,l}$ be the set from the partition system P^e contained in $S_{v_l,j}$. It follows that the sets $A^e_{t,l}$ corresponding to the labels of player i come from different partitions of U^e. Therefore, by Lemma 25, we get that the sets $S_{v_l,j}$ cover exactly $1 - (1 - \frac{1}{k})^{x^e_i}$ fraction of the subuniverse U^e. Hence the total utility derived by the players from the subuniverse U^e is

$$\sum_i p_{i,e} \leq \sum_{i \in N^e_1} r + \sum_{i \in N^e_2} \left(1 - (1 - \tfrac{1}{k})^{x^e_i}\right) r$$

The loss in the total utility compared to the YES case is:

$$\Delta_e \geq nr - \sum_{i \in N^e_1} r - \sum_{i \in N^e_2} (1 - (1 - \tfrac{1}{k})^{x^e_i}) r = r \sum_{i \in N^e_2} (1 - \frac{1}{k})^{x^e_i}$$

By the convexity of the function $(1-\frac{1}{k})^x$, we have that

$$\Delta_e \geq rn_2^e(1-\frac{1}{k})^{\frac{\sum_{i\in N_2^e} x_i^e}{n_2^e}} \tag{5}$$

Let $N_1^{e,\leq k^2}$ be the set of players from N_1^e who have at most k^2 goods of the type $S_{v_l,j}$. Let $N_1^{e,>k^2} = N_1^e \backslash N_1^{e,\leq k^2}$.

$$\sum_e n_1^e = \sum_e |N_1^{e,>k^2}| + |N_1^{e,\leq k^2}| \leq \frac{kn|E|}{k^2} + \sum_e |N_1^{e,\leq k^2}|$$

where the inequality follows from the fact that for the edge e we cannot have more than n/k players receiving more than k^2 goods from the nodes $v_1, v_2, ..., v_k$.

Claim 28 $\sum_e |N_1^{e,\leq k^2}| < \delta n|E|$, *for* $\delta \leq ck^3 2^{-\gamma s}$, *for some constant* c.

Proof : The proof is similar to that of Claim 6. If we assume the contrary to the statement then we can find a labeling which weakly satisfies more than $k^2 2^{-\gamma s}$ fraction of the edges, a contradiction. □

Hence $\sum_e n_1^e \leq \frac{n|E|}{k} + \delta n|E|$, which implies $\sum_e n_2^e \geq (1-\beta)n|E|$, for some small constant $\beta > 0$. In Section 3.3, this sufficed to obtain the hardness result of 3/4, because $\sum_{i\in N_2^e} x_i^e \leq 2n_2^e$. Here a similar argument would need that $\sum_{i\in N_2^e} x_i^e \leq kn_2^e$, which may not be true for every edge because players in N_1^e are only weakly satisfying e. However, we will see that for most edges, $\sum_{i\in N_2^e} x_i^e$ is still small.

Since $\sum_e n_1^e \leq \beta n|E|$, it follows that for at least a $1-\sqrt{\beta}$ fraction of the edges, $n_2^e \geq (1-\sqrt{\beta})n$. Call these edges good. For each good edge e:

$$\frac{\sum_{i\in N_2^e} x_i}{n_2^e} \leq \frac{kr}{(1-\sqrt{\beta})n} \leq k(1+\beta')$$

for some small constant $\beta' > 0$. From (5), we get that for every good edge the loss $\Delta_e \geq rn_2^e(1-\frac{1}{k})^{k(1+\beta')} \geq rn_2^e(1-\beta'')\frac{1}{e}$, for some small constant $\beta'' > 0$. Summing the loss over all the good edges, we get that the total loss in utility is at least

$$r\sum_{e:e\text{is good}} (1-\sqrt{\beta})n(1-\beta'')\tfrac{1}{e} \geq \frac{n}{e}r|E|(1-\sqrt{\beta})^2(1-\beta'') \geq \frac{1}{e}nr|E|(1-\epsilon)$$

where $\epsilon > 0$ is some small constant. Hence the total utility is at most $(1-\frac{1}{e}+\epsilon)nr|E|$ □

Given any $\epsilon > 0$, we can choose large enough constants s, k so that Lemma 27 holds. Hence we get:

Theorem 29 *For any $\epsilon > 0$, there is no polynomial time $(1-\frac{1}{e}+\epsilon)$-approximation algorithm for the allocation problem with monotone submodular utilities, unless* **P**=**NP**.

3.5 Conclusion and Open Problems

In this chapter we proved a $(1 - 1/e \simeq 0.632)$-hardness of approximation in the value query model. Thus the known results for the allocation problem with submodular utilities can be summarized as in Table 1.

Table 1: Approximability results for submodular utilities

	Algorithms	Hardness
Value Queries	$1/2$ [43]	$1 - 1/e$
Demand Queries	$1 - 1/e$ [15]	$1 - O(1/m)$ [58]

There is a gap between the upper and lower bounds in both the models. It would be interesting to narrow these gaps. It will also be interesting to obtain truthful mechanisms with a good approximation guarantee for the allocation problem.

CHAPTER IV

THE USE OF RANDOMIZATION IN AUCTION DESIGN

4.1 Introduction: Truth-revealing Auctions

In this chapter we will introduce the notion of a truth-revealing auction. A truth-revealing (or strategy-proof) auction is one in which bidders, upon knowing the rules of the auction, have no incentive to misreport their utility for the item or items being sold. One (trivial) way to design a truth-revealing auction is to declare up front that all the goods will be sold to one special player for free. Clearly no bidder can change the outcome (and hence his profit) by misreporting his utility. Of course, one wants to design auctions so that not only is the auction truth-revealing, but also that it results in a desirable outcome, e.g., one which maximizes total utility of the bidders, or one which maximizes the profit of the auctioneer.

Classical auction design techniques restrict the auctioneer to designing deterministic auctions. However, there is not much freedom of choice when it comes to designing deterministic truth-revealing auctions. It is a classical result (folk theorem) that the only deterministic truth-revealing auctions are those which are bid-independent, i.e., the price offered to a bidder is independent of his own bid (but may depend on the other bidders' bids), and the goods are sold to that bidder whose bid is at least the price offered to him.

This freedom of design is further cut down when one wants to design deterministic truth-revealing auctions which also maximize the total utility of the outcome. Here, it is known that the famous Vickrey-Clarke-Groves mechanism (which, in the single-item case, is simply the second-price auction) is the only auction with both these properties.

Very recently, researchers have started looking at the role randomization can play in designing better auctions. Randomization has yielded extremely efficient algorithms for many optimization problems in computer science, in several cases where no deterministic algorithms were known. One may expect such a powerful role even in auction design. In a line of very interesting results, [22, 17, 21] prove that randomization indeed provides truth-revealing auctions which maximize auctioneer profit much better than deterministic auctions can. They demonstrate this in one very important case - that of unlimited supply, or what is called digital goods.

Of course the notion of truth-revelation in randomized auctions is different from that in deterministic auctions. The simplest notion in the randomized case is truth-revelation in expectation - i.e. truth-revealing against *risk-neutral* bidders (bidders who wish to simply maximize their expected profit, where the expectation is over the coin tosses of the auctioneer).

With this new notion of truthfulness and armed with the power of randomization, we get much more freedom to design auctions. Indeed now auctions need not be bid-independent, but the price offered to a bidder may depend on his own bid in some random manner. However, we will show in this chapter that the freedom of design is still quite limited in the case of unlimited supply.

In the digital goods setting we prove that for any randomized auction which is truthful in expectation, there exists an equivalent randomized auction which randomizes over truthful deterministic auctions. By equivalent auctions we mean auctions in which the probability of winning and the expected price offered are the same for all bidders for all bid values. We also prove an approximate equivalence proof in the case where the bids come from a discrete space. Finally, we consider the computational issue of finding an efficient equivalent auction.

4.2 Previous work and a statement of the main result

Pioneering work in the study of auctions includes that of Vickrey [73], its generalization to the Vickrey-Clarke-Groves mechanism [73, 12, 23], and that of Myerson [64]. Recent interest in the Computer Science community has led to the study of various computational issues in auction design and mechanism design in general - computational infeasibility of VCG mechanisms [57], the complexity of combinatorial auctions [45, 55, 4, 41], approximating optimal revenue of auctions [65, 66], and competitive analysis[1] of auctions [22, 17, 21], among others. One important paradigm that has emerged [22, 17, 21, 8, 4] is that of randomized auctions - auctions in which the auctioneer is allowed to randomize over different outcomes. One advantage of randomized auctions over deterministic auctions is that they can be more efficient for the auctioneer in terms of generating revenue. For example, [21] prove that in the digital goods setting no deterministic auction can be competitive w.r.t. a utility-knowing auction (in terms of revenue maximization) but there exist simple competitive randomized auctions. [8] prove a similar result in an online digital goods auction model.

In auction design one important property that is often desired is truth-revelation. A deterministic auction is said to be truthful if for every bidder, bidding true utility maximizes his profit, no matter what the other bidders bid (truth-telling as a weakly dominant strategy). Truthful auctions have the advantage that the auctioneer knows that the bidders will bid their true utility, and the bidders themselves can decide what to bid without worrying about modeling other bidders. It is well known that deterministic truthful auctions are precisely those which are bid-independent - the price offered to a bidder does not depend on his own bid.

In the case of randomized auctions truth revelation does not have a standard definition. In [21] truthful randomized auctions are defined to be distributions over deterministic truthful auctions. The competitive randomized auctions of [21] are in fact of this type. We call these auctions *strongly truthful* or *randomizations over truthful auctions.*

Another definition that has been considered in the literature is that for every bidder, bidding true utility maximizes his expected profit, no matter what the other bidders bid. We call such auctions *truthful in expectation.* [4] provide a combinatorial auction for single-parameter agents which is truthful in expectation. [2]

The natural question which arises is: what is the relationship between these two definitions of truthful randomized auctions? It is clear that every randomized auction which is

[1] Bounding the worst case ratio of the revenue generated by the given auction to that generated by an optimal utility-knowing auction.

[2] Other notions of truthful randomized auctions have also been considered, e.g. truth-telling with high probability, or small (bounded) incentives to lie. The example of [4] is also truthful with high probability.

strongly truthful is also truthful in expectation. For the other direction, strongly truthful is considered to be a strictly stronger definition than truthful in expectation ([21, 4]). Indeed, there exist simple examples of auctions which are truthful in expectation but are not strongly truthful (see Section 4.4 for an example).

In this chapter we consider the digital goods setting in which the auctioneer has an unlimited number of copies of the item to be sold, each bidder bids for one copy of the item, and any number of bidders can be sold a copy, possibly at different prices. In this setting we prove that for every randomized auction which is truthful in expectation there exists an *equivalent* auction which is strongly truthful. By equivalence we mean that for any bid values, the probability that a bidder wins is the same in the two auctions and so is the expected price offered. Equivalent auctions look identical to bidders who bid to maximize their expected profit.

Thus our result proves that when the bidders and the auctioneer are just concerned about their expected profit, then the auctioneer may as well (in principle) restrict himself to strongly truthful auctions. Our simulation considers each bidder independently, hence it cannot meet additional constraints that auctions in other settings may have, e.g. that only a single item is sold (in single-item auctions) or that no item is over-sold (in combinatorial auctions).

We provide the main definitions in Section 4.3. In Section 4.4.1 we prove the main result. Section 4.4.2 deals with the case that the bidders bid from a discrete set of values, for example, the integers. We consider the question of efficiency of our simulation in Section 4.4.3.

4.3 Preliminaries

In the digital goods auction setting [22] there is an auctioneer (mechanism designer) who is selling some item. There are n bidders who wish to purchase that item. The auctioneer has unlimited copies of the item and can sell copies to any number of bidders, possibly offering different prices to different bidders. Each bidder requires only one copy, and has a private valuation, or utility for the item. We assume that the utility of bidder i is a single number

u_i. The bidders bid for the item in a sealed bid manner. Let the bids be denoted by the n-vector $\mathbf{b}$. Let b_i denote bidder i's bid. Let $\mathbf{b^{-i}}$ be the vector of bids without b_i. We may write $\mathbf{b}$ as $(\mathbf{b^{-i}}; b_i)$. The auctioneer looks at $\mathbf{b}$ and decides which bidders get a copy of the item and at what price. We say that a bidder wins if he gets a copy. The price offered to a winning bidder must be less than or equal to his bid. If a bidder wins, he has to buy the item at the offered price, even if the price is more than his true utility (which may happen if he bid more than his true utility). Bidders bid with the aim of maximizing their profit, which for a winning bidder is the difference between the true utility and the price offered, and for a losing bidder is 0.

A randomized auction is an auction in which the auctioneer is allowed to flip coins to determine who wins and how much they pay. The auction should be such that every sequence of coin tosses results, for each bidder, either in a rejection or a price less than the bid (it is not sufficient, e.g., that the expected price is less than the bid). Thus, a randomized auction can be considered to be a probability distribution over deterministic auctions. The expected profit of bidder i in a randomized auction is the expected value of the random variable which is 0 if the bid is rejected, and is $u_i - p_i$ if the bid is accepted with a price of p_i. Bidders bid to maximize their expected profit.

Truthfulness is an extremely useful property that is often desired in auction design - knowing the rules of the auction, the bidders should want to bid their true utility. For deterministic auctions, a very strong notion of truthfulness that we may impose on the bidders is that of truth-telling as (weakly) dominant strategy. This is also the easiest way to analyze auctions. A bid value b_i is said to be a *weakly dominant strategy* for bidder i if it is the case that no matter what the other bidders bids $\mathbf{b^{-i}}$ are, b_i is a best bid.

Definition 3 *A deterministic auction is said to be* truthful *if for every bidder, bidding the true utility maximizes profit, no matter what the other bidders bid, i.e. reporting the true utility is a weakly dominant strategy for bidders who wish to maximize their profit.*

Truthful deterministic auction have a neat characterization.

Definition 4 *A deterministic auction is said to be* bid-independent *if it is of the following* *form: for each bidder i, compute a threshold t_i which is a function only of the other bidders' bids; bidder i wins if $b_i \geq t_i$ and is offered a price of t_i; bidder i loses otherwise.*

The following is well-known and is easy to see:

Theorem 30 *A deterministic auction is truthful iff it is bid-independent.*

The simplest example of a bid-independent auction is Vickrey's single item auction [73], in which the threshold is the maximum of the other bidders' bids.

For randomized auctions there are two different definitions of truthfulness that have been considered in the literature [5, 4, 21].

Definition 5 *A randomized auction is said to be* truthful in expectation *if for every bidder, bidding the true utility maximizes expected profit, no matter what the other bidders bid, i.e. reporting utility is a weakly dominant strategy for bidders who wish to maximize their expected profit.*

Definition 6 *A randomized auction is said to be* strongly truthful *if it is an oblivious randomization over deterministic truthful auctions.*

By an oblivious randomization we mean that the different deterministic truthful auctions are chosen (for bidder i) with probabilities that do not depend on bidder i's bid. However, these probabilities may depend on $\mathbf{b^{-i}}$.

4.4 The Equivalence Theorem

Truthful in expectation is considered to be a (strictly) weaker definition of truthfulness than strongly truthful ([4, 21]). Bidders in a strongly truthful auction will bid truthfully even if they are told the outcome of the coin tosses, while this may not be the case in an auction which is truthful in expectation. Indeed, the following is an example of an auction which is truthful in expectation, but is not strongly truthful.

Example: Given a bid vector $\mathbf{b}$, let $M_i = \max\{b_j, j \neq i\}$. D_1 is the deterministic auction that always rejects. D_2 is the deterministic auction that for each bidder i, rejects if $b_i < M_i$, offers M_i if $M_i \leq b_i < M_i + 2$ and offers $M_i + 1$ if $b_i \geq M_i + 2$. The randomized auction R considers each bidder i independently. For bidder i, R calls D_2 with a probability of 0 for $b_i < M_i$, with a probability of 1/2 if $M_i \leq b_i < M_i + 2$, and with a probability of 1 if $b_i \geq M_i + 2$. With the remaining probability R calls D_1. It can be verified that R is truthful in expectation. However, D_2 is not a truthful deterministic auction. Furthermore, the probability with which R calls D_2 depends on b_i. Hence R is not strongly truthful.

Our main result (Theorem 32) says that for every randomized auction which is truthful in expectation there is an equivalent strongly truthful randomized auction, where the equivalence is as defined below.

Let R be a randomized auction. Fix a bidder, say i. In the rest of the analysis we will only consider R's actions for bidder i. For bids $\mathbf{b}$, we say that bidder i wins in R if bidder i is offered the item. Let $q_i(\mathbf{b})$ be the probability with which bidder i wins. Let $p_i(\mathbf{b})$ be the expected value of the price offered, conditioned on bidder i winning. Define the *expected price tuple* offered to bidder i to be the tuple $(q_i(\mathbf{b}), p_i(\mathbf{b}))$. In the case that bidder i loses (i.e. $q_i(\mathbf{b}) = 0$), we define the expected price offered to be $(0, \infty)$.

We say that two expected price tuples (q, p) and (q', p') are equal if $q = q'$ and $p = p'$.

Definition 7 *Two randomized auctions R and R' are said to be* equivalent *if for every bidder i, for every bid vector $\mathbf{b}$, the expected price tuple offered in R is equal to the expected price tuple offered in R'.*

Note that two equivalent auctions would appear identical to bidders and auctioneers who aim to maximize their expected profit.

Example contd. (see figure)**:** The following strongly truthful randomized auction R' is equivalent to the auction R in the example above. Again, R' considers each bidder i independently. R' reads $\mathbf{b^{-i}}$ and randomizes over two deterministic truthful auctions. With $M_i = \max\{b_j, j \neq i\}$, as in R, let D_{M_i} be the truthful deterministic auction with threshold M_i, and let D_{M_i+2} be the deterministic truthful auction with threshold $M_i + 2$. R' calls D_{M_i} and D_{M_i+2} with probability 1/2 each. It can be verified that R' is equivalent to R.

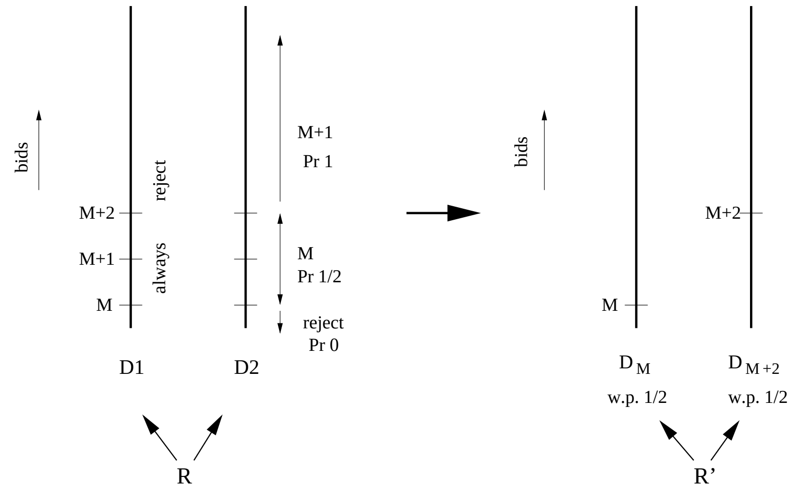

Figure 4: An example of the simulation proving the equivalence between two notions of truthfulness in randomized auctions

We need the following definition.

Definition 8 *We call a randomized auction* monotonic *if for every bidder i, for fixed bids of the other players $\mathbf{b^{-i}}$ if $b_i < b'_i$ then $q_i(\mathbf{b^{-i}}; b_i) \le q_i(\mathbf{b^{-i}}; b'_i)$, i.e. the probability of winning does not decrease if the bid increases, all other bids remaining the same.*

It is well known that every randomized auction which is truthful in expectation has to be monotonic. We provide a proof below for completeness.

Lemma 31 *Every randomized auction which is truthful in expectation is monotonic.*

Proof : Suppose a randomized auction which is truthful in expectation is not monotonic. This means that for some bids $\mathbf{b^{-i}}$ there exist two bids $b_i = x$ and $b_i = y$, $x < y$ s.t. the probability of acceptance for x is greater than that for y. Let the expected price tuple for $(\mathbf{b^{-i}}; x)$ be (q_x, p_x) and for $(\mathbf{b^{-i}}; y)$ be (q_y, p_y). We have $x < y$ and $q_x > q_y$. Now since the auction is truthful in expectation we have:

When the utility is x, bidding y does not increase expected profit:

$$q_x(x - p_x) \ge q_y(x - p_y)$$

When the utility is y, bidding x does not increase expected profit:

$$q_y(y - p_y) \ge q_x(y - p_x)$$

Let $\alpha = q_x/q_y > 1$. We get

$$y \leq \frac{\alpha p_x - p_y}{\alpha - 1} \leq x$$

a contradiction. $\square$

Thus for any randomized auction which is truthful in expectation, $q_i(\mathbf{b^{-i}}; b_i)$ is a non-decreasing function of b_i. We assume that there is a number a_0 s.t. $q_i(\mathbf{b^{-i}}; x) = 0$ for all $x < a_0$, i.e. if the bid is low enough then it will be rejected with probability 1. We also assume that there is a number a_1 s.t. $q_i(\mathbf{b^{-i}}; x) = 1$ for all $x \geq a_1$, i.e. if the bid is high enough then it will be accepted with probability 1. These assumptions are made so that we can think of $q_i(\mathbf{b})$ as a probability distribution with bounded support. The analysis will go through even without these assumptions.

4.4.1 The Main Theorem

The main idea behind the theorem is the following: The given auction R looks at all the bids $\mathbf{b}$ and decides $q_i(\mathbf{b})$ and $p_i(\mathbf{b})$. The simulating auction R' will only look at $\mathbf{b^{-i}}$ and then average over all possible bids b_i that bidder i could have made (without reading the actual bid). In particular, R' will consider $q_i(\mathbf{b^{-i}}; b_i)$ as a probability distribution, choose at random a bid value b_i according to the distribution $q_i(\mathbf{b^{-i}}; b_i)$ and play the deterministic auction with threshold b_i.

Theorem 32 *For any randomized auction R which is truthful in expectation, there is a strongly truthful randomized auction R' s.t. R' is equivalent to R.*

Proof : Given R, a randomized auction truthful in expectation, we shall define a strongly truthful randomized auction R' equivalent to R. Fix bids $\mathbf{b^{-i}}$. For R we know that $q_i(\mathbf{b^{-i}}; b_i)$ is a non-decreasing function of b_i, which attains a minimum of 0 and a maximum of 1. Thus we can think of $q_i(\mathbf{b^{-i}}; b_i)$ as a (cumulative) probability distribution function.

R' is a randomized auction that considers each bidder independently. Let D_x be the deterministic truthful auction with threshold x. For each bidder, R' randomizes over (possibly) all D_x, $x \in \mathbf{R}$:

On bids $\mathbf{b}$, for bidder i, R' picks $x \in \mathbf{R}$ according to the probability distribution $q_i(\mathbf{b^{-i}}; x)$, and runs the auction D_x.

R' is strongly truthful since it randomizes over the truthful deterministic auctions D_x. Note that the coin tosses of R' also do not depend on the bid b_i.

To prove that R' is equivalent to R we need to show that for any bid values $\mathbf{b}$, (i) the probability that bidder i wins is the same in R and R', and (ii) the expected price offered is also the same.

For the sake of presentation we shall assume that $q_i(\mathbf{b^{-i}}; b_i)$ and $p_i(\mathbf{b^{-i}}; b_i)$ are both differentiable functions of b_i. (The result holds in general). Fix $\mathbf{b^{-i}}$. For ease of notation we shall denote $q_i(\mathbf{b^{-i}}; b_i)$ by $q(b_i)$, when i and $\mathbf{b^{-i}}$ is known from context. Similarly $p_i(\mathbf{b^{-i}}; b_i)$ will be shortened to $p(b_i)$.

(i) For a bid $b_i = x$, R accepts with probability $q(x)$. R' accepts iff it picks D_y with $y \leq x$. But this happens with probability precisely $q(x)$.

(ii) The expected price offered by R is $p(x)$. From the definition of R' we can see that the expected price offered by R' is

$$\frac{1}{q(x)} \int_{a_0}^{x} bq'(b)db$$

Hence we need to prove that for any randomized auction which is truthful in expectation

$$p(x) = \frac{1}{q(x)} \int_{a_0}^{x} bq'(b)db$$

However, this is known (originally in [54], and also used recently in [5]). We provide the proof for the sake of completeness:

Since R is truthful in expectation the function

$$f_b(y) = q(y)(b - p(y))$$

is maximized at $y = b$. ($f_b(y)$ denotes the expected profit when the utility is b and the bid is y). For $y > a_0$, $q(y)$ and $p(y)$ are both differentiable, hence we have:

$$f_b'(y) = q'(y)(b - p(y)) - q(y)p'(y)$$

Hence we have that for every $b > a_0$,

$$q'(b)(b - p(b)) = q(b)p'(b)$$

$$i.e. \quad bq'(b) = [p(b)q(b)]'$$

Hence

$$\int_{a_0}^{x} bq'(b)db = \Big[p(b)q(b)\Big]_{a_0}^{x} = p(x)q(x)$$

as required. (It can be shown that since R is monotonic, the second derivative is negative at $y = b$).

□

In the general case when $q(b_i)$ is not a differentiable function of b_i, the proof is the same, by treating $q(b_i)$ as a cumulative probability distribution function. We note here that this proof technique has been used before - the lemma on probability and price is originally from [54], and has been applied in [64, 5, 4].

Note: Since the equivalent auction considers each bidder independently, it makes sense only in the digital goods setting. While the statement of the theorem holds for all one-parameter mechanisms, general one-parameter mechanisms may have further constraints which the simulation may not meet. For example, a k-item auction cannot sell more than k items. Combinatorial auctions for known single minded bidders (see e.g., [4]) require that no item is over-sold. Our simulation works independently for each bidder and may not meet such constraints. (The simulation will, however, work in auctions in which the set of winning bidders is the same over all outcomes - the example given above works even if the auction is a single-item auction, since only the highest bidder ever wins).

4.4.2 Discrete Bid Values

In the case when the bidders bid from some discrete set of values, say for example the integers, then the equivalence is not exact as in Theorem 32. However we can still prove an almost exact equivalence.

First note that for discrete bid values deterministic truthful auctions can be more general than for real bid values. Asusme that the bid values are integers. Then a deterministic truthful auction can find a threshold $t(\mathbf{b^{-i}}) \in \mathbf{Z}$, reject if $b_i < t(\mathbf{b^{-i}})$ and accept if $b_i \geq t(\mathbf{b^{-i}})$ and offer a price of t or $t-1$.

Definition 9 *Two randomized auctions R and R' are said to be (λ, s)-equivalent (for $0 \leq \lambda \leq 1$, $s > 0$) if for every bidder i, for every bid vector $\mathbf{b}$, if (q,p) is the expected price tuple in R and (q', p') is the expected price tuple in R' then $|q - q'| < \lambda$ and $|p - p'| < s$. If $\lambda = 0$ then we simply say that R and R' are s-equivalent.*

We get the following version of Theorem 32. The proof is a discrete version of the proof of Theorem 32.

Theorem 33 *Suppose the bidders can only bid integer values. Then for any randomized auction R which is truthful in expectation, there is a strongly truthful randomized auction R' s.t. R' is 1-equivalent to R.*

4.4.3 Efficient Simulations

Definition 10 *A randomized auction is said to run in polynomial time, if for every bid vector $\mathbf{b}$ and for every bidder i, the auction can determine in polynomial time (possibly after reading a polynomial number of random bits) whether to offer the item to bidder i or not and at what price to offer.*

Given an auction which is truthful in expectation and runs in polynomial time it is not clear that the equivalent strongly-truthful auction of Theorem 32 also runs in polynomial time. For a polynomial time simulation one needs to be able to sample from the probability of winning, $q(b_i)$, considered as a probability distribution over the reals.

Definition 11 *For a randomized algorithm R and a bidder i, we say that the probability of winning $q(b_i)$ is polynomial time samplable if there exists a polynomial time randomized algorithm which can generate a random number distributed according to $q(b_i)$ viewed as a cumulative probability distribution.*

Corollary 34 *For every randomized auction R which is truthful in expectation and is such that the probability of winning $q(b_i)$ is polynomial time samplable for every bidder i, there exists a polynomial time strongly-truthful randomized auction R' which is equivalent to R.*

Proof : This follows from the proof of Theorem 32: the equivalent strongly truthful auction R' of Theorem 32 samples from the set of all deterministic truthful auctions, sampling D_t according to the cumulative probability distribution $q(b_i)$. □

The following lemma shows that we can approximately sample from $q(x)$ in polynomial time when the bids are from a discrete space.

Lemma 35 *Suppose we are given a polynomial time auction which is truthful in expectation and in which the bidders bid integer values. Suppose that for any b_i we can compute $q(b_i)$ in polynomial time and also that for each bidder i we know the bid values a_0 and a_1 such that $q(a_0) = 0$ and $q(a_1) = 1$. Let $A = a_1 - a_0$. Then we can sample $x \in \mathbf{Z}$ approximately according to the probability distribution $q(b_i)$ in time poly$(n, \log A)$ with error exponentially small in n.*

Proof : We only sketch the proof here. Since the given auction runs in polynomial time (and uses, say, $m = poly(n)$ random bits) $q(b_i)$ is a discrete distribution with minimum weight at least 2^{-m}. Use m^2 random bits to pick a number r between 0 and 1 uniformly at random with granularity 2^{-m^2}. Now perform a binary search between a_0 and a_1 and find an x such that $q(x-1) < r \leq q(x)$. □

From Corollary 34 and Lemma 35 we get:

Theorem 36 *Suppose we are given an auction R which runs in polynomial time, is truthful in expectation and in which the bidders bid integer values. Suppose that for any b_i we can compute $q(b_i)$ in polynomial time and also that for each bidder i we know the bid values a_0 and a_1 such that $q(a_0) = 0$ and $q(a_1) = 1$. Let $A = a_1 - a_0$. Then there is a strongly truthful auction which runs in time poly$(n, \log A)$ and is $(O(2^{-n}), 2)$-equivalent to R.*

4.5 Conclusion and Open Questions

While it is clear that every strongly truthful randomized auction is also truthful in expectation, there are simple examples of auctions which are truthful in expectation but not strongly truthful. Our result provides the following equivalence in the digital goods setting:

For any auction which is truthful in expectation we can find one which is strongly truthful such that it looks identical to bidders who are bidding to maximize their expected profit. In the equivalent auction even if the bidders know the coin tosses of the auction they will have no incentive to lie.

For other auction settings it is important to also preserve the correlations between the probabilities of winning of different bidders, e.g., a single item auction can only have one winning bidder in any outcome. It is interesting to see if such correlations can also be preserved in an equivalent auction.

It is also interesting to see if this kind of equivalence also holds for randomized group-strategyproof mechanisms, for example, for randomized cost-sharing mechanisms.

CHAPTER V

PLAYING LARGE GAMES USING SIMPLE STRATEGIES - COMPUTING NASH EQUILIBRIA IN GAMES

In this chapter we prove the existence of ϵ-Nash equilibrium strategies with support logarithmic in the number of pure strategies. We also show that the payoffs to all players in any (exact) Nash equilibrium can be ϵ-approximated by the payoffs to the players in some such logarithmic support ϵ-Nash equilibrium. These strategies are also uniform on a multiset of logarithmic size and therefore this leads to a quasi-polynomial algorithm for computing an ϵ-Nash equilibrium. This is the first subexponential algorithm for finding an ϵ-Nash equilibrium. Our results hold for any multiple-player game as long as the number of players is a constant (i.e., it is independent of the number of pure strategies). A similar argument also proves that for a fixed number of players m, the payoffs to all players in any m-tuple of mixed strategies can be ϵ-approximated by the payoffs in some m-tuple of constant support strategies.

We also prove that if the payoff matrices of a two person game have low rank then the game has an exact Nash equilibrium with small support. This implies that if the payoff matrices can be well approximated by low rank matrices, the game has an ϵ-equilibrium with small support. It also implies that if the payoff matrices have constant rank we can compute an exact Nash equilibrium in polynomial time.

5.1 Introduction: Nash equilibrium in finite games

Non-cooperative game theory has been extensively used to analyze situations of strategic interactions. Recently, it has been pointed out [61, 39, 68] that many Internet related problems can be studied within the framework of this theory. The most important solution concept in non-cooperative games is the notion of Nash equilibrium.

Consider a two person game G, where for simplicity the number of available (pure) strategies for each player is n. We will refer to the two players as the row and the column player and we will denote their payoff matrices by R, C respectively. The results of Section 5.4.1 are also generalized for multiple person games in which the players do not have the same number of pure strategies.

A *mixed strategy* (or a randomized strategy) for a player is a probability distribution over the set of his pure strategies and will be represented by a vector $x = (x_1, x_2, ..., x_n)$, where $x_i \geq 0$ and $\sum x_i = 1$. Here x_i is the probability that the player will choose his ith pure strategy. If $x_i > 0$ we say that the mixed strategy x *uses* the ith pure strategy. The *support* of x ($Supp(x)$) is the set of pure strategies that it uses. A mixed strategy is called *k-uniform* if it is the uniform distribution on a multiset S of pure strategies, with $|S| = k$. For a mixed strategy pair x, y, the payoff to the row player is the expected value of a random variable which is equal to R_{ij} with probability $x_i y_j$. Therefore the payoff to the row player is (x, Ry), where $(. \,, .)$ denotes the inner product of two n-dimensional vectors. Similarly the payoff to the column player is (x, Cy).

The notion of a Nash equilibrium [56] is formulated as follows:

Definition 12 *A pair of strategies x^*, y^* is a Nash equilibrium point if:*

(i) For every (mixed) strategy $\bar{x}$ of the row player,
$(\bar{x}, Ry^*) \leq (x^*, Ry^*)$, and

(ii) For every (mixed) strategy $\bar{y}$ of the column player, $(x^*, C\bar{y}) \leq (x^*, Cy^*)$

Similarly we can define ϵ-equilibria (this definition is well known in the literature):

Definition 13 *For any $\epsilon > 0$ a pair of mixed strategies x', y' is called an ϵ-Nash equilibrium point if:*

(i) For every (mixed) strategy $\bar{x}$ of the row player,
$(\bar{x}, Ry') \leq (x', Ry') + \epsilon$ and

(ii) For every (mixed) strategy $\bar{y}$ of the column player, $(x', C\bar{y}) \leq (x', Cy') + \epsilon$

5.2 Two criticisms of Nash equilibrium as a solution concept

In this chapter we consider the following two issues concerning Nash equilibria:

First, it is currently not known if Nash equilibria can be computed efficiently. For two player games the known algorithms [36, 37, 38, 40, 46, 47, 51] either have exponential worst-case running time (in the number of available pure strategies) or it is unknown whether they run in polynomial time. For three player games, the problem seems to be even more difficult. While for two player games it can be formalized as a Linear Complementarity Problem (and hence some of the algorithms above) the problem for three player games is a Non-linear Complementarity Problem. Furthermore there exist examples of small three player games with rational payoff matrices in which all Nash equilibria are irrational. Algorithms for approximating equilibria in multiple player games (among others, [67, 75]) are also believed to be exponential. The problem of computing Nash equilibria has been of considerable interest in the computer science community and has been called one of the central open problems in computational complexity (Papadimitriou [61]). In fact it is known that the problem for two-person games lies in some class between P and NP [60]. It is also known that determining the existence of a Nash equilibrium with some additional natural properties (e.g. maximizing payoff sum, maximizing support) is NP-hard [19, 70]. For surveys on computational issues of Nash equilibria see [74, 53].

A second and related issue is the need to play simple strategies. Even if Nash strategies can be computed efficiently, they may be too complicated to implement. This has been pointed out, among others, by Simon [72] and later by Rubinstein [69] in the context of bounded rationality. Players tend to prefer strategies as simple as possible. They might prefer to play a sub-optimal strategy (with respect to rationality) instead of following a complex plan of action which might be difficult to learn or to implement. In this chapter we consider normal form games and our notion of simple strategies is strategies which are uniform on a small support set. The importance of small support strategies becomes clear if we consider the pure strategies to be resources. In this case an equilibrium is almost impractical if a player has to use a mixed strategy which randomizes over a large set of pure strategies. The problem with the requirement of small strategies, of course, is that there exist games whose Nash equilibria are completely mixed (i.e., a player has to randomize over all his available pure strategies).

5.3 Statement of the results and previous work

We address both these problems (namely, the need for efficient algorithms and the need for simple strategies), by using the weaker concept of ϵ-equilibrium (strategies from which each player has only an ϵ incentive to defect). More precisely:

Our main result (Section 5.4.1) is that for any two-person game there exists an ϵ-equilibrium with only logarithmic support (in the number of available pure strategies). Moreover the strategy of each player in such an equilibrium is uniform on a small multiset and can be expressed in polylogarithmically many bits. In our opinion, this is an interesting observation on the structure of competitive behavior in various scenarios - namely, extremely simple approximate solutions exist. This result directly yields a quasi-polynomial ($n^{O(\ln n)}$, where n is the number of available pure strategies) algorithm for computing such an approximate equilibrium. To our knowledge this is the first subexponential algorithm for ϵ-equilibria. In addition to being small, our approximate equilibria provide both players with a good payoff too: the payoff that each player gets using these strategies is almost the same as that in some exact Nash equilibrium. Finally, our result holds not only for two person games but also for games in which the number of players is independent of the number of pure strategies. It is interesting to note that although the problem of finding exact equilibria seems to become more difficult in the "transition" from two player games to three and more, this is not the case for approximate equilibria.

A second result (Section 5.4.2) is that if the players are allowed to communicate and "sign treaties" then there are *constant* support strategies which approximate the payoffs that each player gets in an equilibrium (in fact there are constant support strategies that approximate the payoffs of *any* pair of strategies). In real life, such treaties are not unknown (though often tacit) - this result can be considered as an explanation of why certain small strategies behave well and are used in real games, as opposed to a large and complicated

Nash equilibrium.

A third question we investigate is: “when does a game have small support exact Nash equilibria?” In Section 5.4.3 we give a sufficient condition for two person games: if the payoff matrices of the players have low rank then there exists a Nash equilibrium with small support. Our original proof of this Theorem was a generalization of a result due to Raghavan ([62]) which deals with completely mixed equilibria. The generalization was based on a careful Gaussian elimination type step. However, we suspect that this Theorem should not be unknown to the Game Theory community as we have recently found simple proofs. We would still like to bring the Theorem to the attention of the broader CS and Economics community as it has some interesting corollaries regarding the computation of Nash equilibria. We prove that if the matrices can be well approximated by low rank matrices, then there exists an approximate equilibrium with small support. It also follows that if the payoff matrices have constant rank, we can compute an exact Nash equilibrium in polynomial time.

The problem of looking for small support equilibria has been studied earlier. Koller and Megiddo [36] prove that for two person games in *extensive form* there exist equilibrium strategies whose support is at most the number of leaves of the game tree. However, not all games can be represented in the extensive form with a small number of leaves (where by small we mean logarithmic in the number of pure strategies). Our result guarantees the existence of equilibria with logarithmic support for any two person normal form game (and also for multiple players as stated above) but the equilibria are only approximate.

It should be noted that since Nash equilibria are fixed points of a certain map [56], ϵ-equilibria can be found using Scarf’s algorithm [71], a general algorithm for finding approximate fixed points of continuous mappings. However, no sub-exponential upper bounds are known for approximating equilibria using this algorithm. In fact, Scarf’s algorithm is known to take exponential time in the worst case for a general fixed point approximation ([26]). Polynomial time algorithms for exact or approximate equilibria but only for special classes of games have also been obtained in [35, 48, 34].

For the class of two-person zero-sum games, results for approximate minmax strategies have been proved independently by Lipton and Young [49] and Althöfer [1]. In fact the proofs of Section 5.4.1 use the same technique (sampling). While [1] gives no details, the author claims that a similar result holds for non-zero sum two person games. The implication from approximate minmax strategies to ϵ-Nash equilibria which also approximate the payoffs in some exact Nash equilibrium does not seem to be direct. Furthermore our result holds for multiple player games too and not only for bimatrix games, which is interesting because multiple player games seem to be more difficult.

The rest of the chapter is structured as follows: In Section 5.4.1 we prove our main result on logarithmic support ϵ-Nash equilibria, and the resulting algorithm. In Section 5.4.2 we prove our result on constant support strategies that approximate the payoffs of arbitrary strategy pair. In Section 5.4.3 we prove that low rank payoff matrices imply the existence of equilibria with small support.

5.4 The Main Results

5.4.1 Logarithmic support ϵ-Nash equilibria

For the present we assume that all entries of R and C are between 0 and 1. Our main result is:

Theorem 37 *For any Nash equilibrium x^*, y^* and for any $\epsilon > 0$, there exists, for every $k \geq \frac{12 \ln n}{\epsilon^2}$, a pair of k-uniform strategies x', y', such that:*

1. *x', y' is an ϵ-equilibrium,*

2. *$|(x', Ry') - (x^*, Ry^*)| < \epsilon$, (row player gets almost the same payoff as in the Nash equilibrium)*

3. *$|(x', Cy') - (x^*, Cy^*)| < \epsilon$, (column player gets almost the same payoff as in the Nash equilibrium)*

Proof :

The proof is based on the probabilistic method. For the given $\epsilon > 0$, fix $k \geq 12 \ln n/\epsilon^2$. Form a multiset A by sampling k times from the set of pure strategies of the row player, independently at random according to the distribution x^*. Similarly form a multiset B by sampling k times from the pure strategies of the column player, independently at random according to the distribution y^*.

Let x' be the mixed strategy for the row player which assigns probability $1/k$ to each member of A and 0 to other pure strategies. Let y' be the mixed strategy for the column player which assigns probability $1/k$ to each member of B and 0 to other pure strategies. Clearly, if a pure strategy occurs α times in the multiset, then it is assigned probability α/k.

Denote by x^i the ith pure strategy of the row player, and by y^j the jth pure strategy of the column player. In order to analyze the probability that x', y' is an ϵ-Nash equilibrium it suffices to consider only deviations to pure strategies.

We define the following events:

$$\begin{aligned}
\phi_1 &= \{\mid (x', Ry') - (x^*, Ry^*) \mid < \epsilon/2\} \\
\pi_{1,i} &= \{(x^i, Ry') < (x', Ry') + \epsilon\}, \qquad (i = 1, ..., n) \\
\phi_2 &= \{\mid (x', Cy') - (x^*, Cy^*) \mid < \epsilon/2\} \\
\pi_{2,j} &= \{(x', Cy^j) < (x', Cy') + \epsilon\}, \qquad (j = 1, ..., n) \\
GOOD &= \phi_1 \cap \phi_2 \bigcap_{i=1}^{n} \pi_{1,i} \bigcap_{j=1}^{n} \pi_{2,j}
\end{aligned}$$

We wish to show that $Pr[GOOD] > 0$. This would mean that there exists a choice of A and B such that the corresponding strategies x' and y' satisfy all three conditions in the statement of the theorem.

In order to bound the probabilities of the events ϕ_1^c and ϕ_2^c we introduce the following events:

$$\begin{aligned}
\phi_{1a} &= \{\mid (x', Ry^*) - (x^*, Ry^*) \mid < \epsilon/4\} \\
\phi_{1b} &= \{\mid (x', Ry') - (x', Ry^*) \mid < \epsilon/4\} \\
\phi_{2a} &= \{\mid (x^*, Cy') - (x^*, Cy^*) \mid < \epsilon/4\} \\
\phi_{2b} &= \{\mid (x', Cy') - (x^*, Cy') \mid < \epsilon/4\}
\end{aligned}$$

Note that $\phi_{1a} \cap \phi_{1b} \subseteq \phi_1$. The expression (x', Ry^*) is essentially a sum of k independent random variables each of expected value (x^*, Ry^*). Each such random variable takes value between 0 and 1. Therefore we can apply a standard tail inequality [27] and get:

$$Pr[\phi_{1a}^c] \leq 2e^{-k\epsilon^2/8}$$

Using a similar argument we have:

$$Pr[\phi_{1b}^c] \leq 2e^{-k\epsilon^2/8}$$

Therefore $Pr[\phi_1^c] \leq 4e^{-k\epsilon^2/8}$ and the same holds for the event ϕ_2^c.

In order to bound the probabilities of the events $\pi_{1,i}$'s and $\pi_{2,j}$'s we define the following auxiliary events:

$$\psi_{1,i} = \{(x^i, Ry') < (x^i, Ry^*) + \epsilon/2\}, \qquad (i = 1, ..., n)$$

$$\psi_{2,j} = \{(x', Ry^j) < (x^*, Ry^j) + \epsilon/2\}, \qquad (j = 1, ..., n)$$

We can easily see that

$$\psi_{1,i} \cap \phi_1 \subseteq \pi_{1,i}, \qquad (i = 1, ..., n)$$

$$\psi_{2,j} \cap \phi_2 \subseteq \pi_{2,j}, \qquad (j = 1, ..., n)$$

Using the Hoeffding bound again we get:

$$Pr[\psi_{1,i}^c] \leq e^{-k\epsilon^2/2}$$

$$Pr[\psi_{2,j}^c] \leq e^{-k\epsilon^2/2}$$

Now by combining the above equations we see that:

$$Pr[GOOD^c] \leq Pr[\phi_1^c] + Pr[\phi_2^c] + \sum_{i=1}^{n} Pr[\pi_{1,i}^c] + \sum_{j=1}^{n} Pr[\pi_{2,j}^c]$$

$$\leq 8e^{-k\epsilon^2/8} + 2n[e^{-k\epsilon^2/2} + 4e^{-k\epsilon^2/8}] < 1$$

Thus $Pr[GOOD] > 0$. □

Note that not only do the strategies x', y' form an ϵ-equilibrium, but they also provide both players a payoff ϵ-close to the payoffs they would get in some Nash equilibrium. In fact, the payoffs of every Nash equilibrium can be thus approximated by a small strategy ϵ-equilibrium. This provides another incentive for the players to remain in the ϵ-Nash equilibrium. Furthermore x', y' are k-uniform, which implies the following corollary:

Corollary 38 *For a 2-person game, there exists a quasi-polynomial algorithm for computing all k-uniform ϵ-equilibria (by Theorem 37 at least one such equilibrium exists).*

Proof : Given an $\epsilon > 0$, fix $k = \frac{12 \ln n}{\epsilon^2}$. By an exhaustive search, we can compute all k-uniform ϵ-equilibria (by Theorem 37 at least one such equilibrium exists; verifying ϵ-equilibrium condition is easy as we need to check only for deviations to pure strategies). The running time of the algorithm is quasi-polynomial since there are $\binom{n+k-1}{k}^2$ possible pairs of multisets to look at. □

To our knowledge this is the first subexponential algorithm for finding an approximate equilibrium. Furthermore, given the payoffs of any Nash equilibrium the algorithm can find an ϵ-Nash equilibrium in which both players receive payoffs ϵ-close to the given values.

When the entries of R and C are not between 0 and 1 the ϵ incentive to defect and the ϵ change in payoff both get magnified by $R_{max} - R_{min}$ for the row player and by $C_{max} - C_{min}$ for the column player. Here R_{max} and R_{min} denote the maximum and minimum entry of R, and similarly for C. Additionally if the players do not have the same number of pure strategies (say n_1, n_2) then the same result holds with $k \geq \frac{12 \ln max\{n_1, n_2\}}{\epsilon^2}$.

Our results can also be generalized to games with more than two players. In particular for an m-person game:

Theorem 39 *Let $s_1^*, ..., s_m^*$ be a Nash equilibrium in an m-person game. Let $p_1^*, ..., p_m^*$ be the payoffs to the players in the Nash equilibrium. Then for any $\epsilon > 0$, there exists, for every $k \geq \frac{3m^2 \ln m^2 n}{\epsilon^2}$, a set of k-uniform strategies $s_1', s_2', ..., s_m'$, such that:*

1. *$s_1', s_2', ..., s_m'$ is an ϵ-equilibrium,*

2. *$|p_i' - p_i^*| < \epsilon$ for $i = 1, ..., m$*

where $p'_1, ..., p'_m$ are the payoffs to the players if they play strategies s'_i.

As we see from Theorem 39 we can guarantee an
ϵ-equilibrium with logarithmic support only when m is independent of n. It seems to us that the technique of sampling cannot help us prove a more general theorem than that. It is an interesting question to see whether this can be done using a different technique. However, it is still interesting that we can prove the existence of simple approximate equilibria even for three player games. This is so because the problem of finding exact equilibria for three player games seems to be much more difficult than for two player games due to irrational equilibria and non-linearity of the Complementarity Problem.

Corollary 38 also generalizes to games with a constant number of players since in this case the number of combinations of multisets that the algorithm has to look at is still quasi-polynomial . Again it would be interesting if a more general result could be proved.

5.4.2 Approximating Payoffs of Nash equilibria with Constant Support

In terms of the size of the support we can do much better, if we have weaker requirements. There may be applications in which we would not even insist on an approximate equilibrium. All we would care for is to approximate the payoffs in an actual Nash equilibrium. The next result is in that direction:

Theorem 40 *For any Nash equilibrium x^*, y^* and any $\epsilon > 0$, there exists, for every $k \geq 5/\epsilon^2$, a pair of k-uniform strategies (x, y), such that*

1. *$|(x, Ry) - (x^*, Ry^*)| < \epsilon$ (row player gets almost the same payoff), and*

2. *$|(x, Cy) - (x^*, Cy^*)| < \epsilon$ (column player gets almost the same payoff),*

Again this result can be generalized to multiple player games. For an m-person game the support of the k-uniform strategies will be $O(m^2 \ln m)$.

Theorem 40 establishes the existence of *constant* support strategies which approximate the payoffs that both players get in a Nash equilibrium. The techniques used to prove this are the same as those used to prove Theorem 37, and the proof is omitted. Again, we assume that the entries of R and C are between 0 and 1 (in the general case we get a magnification by $R_{max} - R_{min}$ and $C_{max} - C_{min}$ as before). Note that Theorem 40 is true for any pair of strategies x^*, y^*, not necessarily for Nash equilibria.

A situation in which this result could be applicable is the following: Consider a game between two players both having a very large number of pure strategies at their disposal. Let v_1, v_2 be the payoffs in a Nash equilibrium to the row and column player respectively. If the support of the equilibrium strategies is very big, then it would be preferable for both players to sign a "bilateral treaty" and use only a small number of strategies, as provided by the result. In that case, both players would still receive a payoff close to v_1 and v_2 respectively, while using a small number of strategies. Furthermore, each player will be able to check, during the game, if the other player has violated the treaty, in which case he can switch to any other strategy.

5.4.3 Low Rank Implies Small Support Exact Equilibria

In this section we investigate the question: when does a two person game have small support exact Nash equilibria? We show that if the payoff matrices have low rank then the game has a small support Nash equilibrium. Furthermore we show that if the payoff matrices can be approximated by low rank matrices then the game has a small support approximate equilibrium (where the approximation factor depends on how well the matrices can be approximated).

Denote again by R, C the payoff matrices for the row and column player respectively. Suppose that R and C are $m \times n$ matrices.

Theorem 41 *Let x^*, y^* be a Nash equilibrium.*
If $rank(C) \leq k$, then there exists a mixed strategy x for the row player with $|Supp(x)| \leq k+1$ such that x, y^* is an equilibrium point. Similarly, if $rank(R) \leq k$, then there exists a mixed strategy y for the column player with $|Supp(y)| \leq k+1$ such that x^*, y is an equilibrium point. Furthermore the payoff that both players receive in the equilibria x, y^* and x^*, y is equal to the payoff in the initial equilibrium x^*, y^*.

Our original proof of this Theorem was a generalization of a result due to Raghavan ([62]) which deals with "completely mixed equilibria", i.e. equilibria which use all the pure strategies. The generalization was based on a careful Gaussian elimination type step. However, we suspect that this Theorem should not be unknown to the Game Theory community as we recently realized that a simple proof follows from the polyhedral structure of the problem and the polyhedral structure of the set of Nash equilibria (see [74, 30]). We would still like to bring the theorem to the attention of the broader CS and Economics community as it has some interesting corollaries regarding the computation of Nash equilibria. We present below another simple proof suggested to us by N. Vishnoi and N. Devanur ([14]):

Let S be the k-dimensional space spanned by the columns of R. Since Ry^* is a convex combination of the columns of R, it can be written as a convex combination of at most $k+1$ columns of R (by Caratheodory's Theorem). Let this new convex combination be Ry. Note that $Supp(y) \subseteq Supp(y^*)$. This implies that y is a best response to x^*. Since $Ry^* = Ry$, x^* is also a best reponse to y. Hence x^*, y is a Nash equilibrium. Since $Ry^* = Ry$ the first player receives the same value in x^*, y as in x^*, y^*. The second player will also receive the same value as in the initial equilibrium because $Supp(y) \subseteq Supp(y^*)$.

Definition 14 *For $n \times n$ matrices C, D, D is an*
ϵ-approximation of C if $C = D + E$, where $|E_{ij}| \leq \epsilon$ for $i, j = 1, ..., n$.

Lemma 42 *Let D be an ϵ-approximation of C. Let x^*, y^* be a Nash equilibrium for the game with payoff matrices R, D. Then x^*, y^* is a 2ϵ-Nash equilibrium for the game with payoff matrices R, C.*

Proof : Clearly $(x^*, Ry^*) \geq (\bar{x}, Ry^*)$, $\forall\ \bar{x}$. For any strategy $\bar{y}$:

$$(x^*, Cy^*) = (x^*, Dy^*) + (x^*, Ey^*) \geq (x^*, D\bar{y}) + (x^*, Ey^*)$$

Since $|E_{ij}| \leq \epsilon$, $\forall\ i, j$,

$$(x^*, E\bar{y}) - (x^*, Ey^*) \leq 2\epsilon$$

Hence,

$$(x^*, Cy^*) \geq (x^*, D\bar{y}) + (x^*, E\bar{y}) - 2\epsilon = (x^*, C\bar{y}) - 2\epsilon$$

□

Corollary 43 *For any game R, C, and for any $k < n$, if C can be ϵ-approximated by a rank k matrix then there exists a 2ϵ-equilibrium x, y with $|Supp(x)| \leq k+1$. Similarly for R.*

In particular, we can use the Singular Value Decomposition to approximate the payoff matrices R, C by rank k matrices for any k. The approximation factor ϵ of Corollary 43 is then a function of the singular values of the matrices.

A useful corollary arises from the observation that for 2-person games, if we know the support of a Nash equilibrium, then we can compute the exact equilibrium strategies in polynomial time. This is because an equilibrium strategy y for the column player equalizes the payoff that the row player gets for every pure strategy in his support and vice versa. Hence we can write a linear program and compute the Nash equilibrium with the given support. The following is a direct consequence of this observation and Theorem 41.

Corollary 44 *If the payoff matrices R, C have constant rank, then we can compute an exact Nash equilibrium in polynomial time. In particular if one of the players has a constant number of pure strategies, we can compute a Nash equilibrium in polynomial time.*

5.5 Discussion

Another attempt to prove the results of Section 5.4.1 would be to approximate the vectors of a Nash equilibrium by vectors of small support. It is not difficult to see that we can approximate any probability distribution vector by a vector of logarithmic support in the l_∞ norm with error at most $1/\log n$. However, approximating an equilibrium x^*, y^* in this manner does not imply that the approximating vectors will form an ϵ-equilibrium, for any given fixed ϵ. On the other hand it can be shown that an ϵ-approximation in the l_1 norm does yield an ϵ-equilibrium, but such an approximation is not always possible (e.g. if the Nash strategies are the uniform distributions).

An interesting open question is whether we can generalize the results of Section 5.4.1 to games where the number of players is an increasing function of n. Another question would be to generalize the result so that the incentive to defect won't depend on the range of the payoff matrices (which can be much higher than the expected payoff in any equilibrium).

REFERENCES

[1] Althöfer, I., "On sparse approximations to randomized strategies and convex combinations," *Linear Algebra and Applications*, vol. 199, pp. 339–355, 1994.

[2] Andelman, N. and Mansour, Y., "Auctions with budget constraints," in *9th Scandinavian Workshop on Algorithm Theory (SWAT)*, pp. 26–38, 2004.

[3] Andelman, N. and Mansour, Y., "Auctions with budget constraints," in *SWAT*, 2004.

[4] Archer, A., Papadimitriou, C., Talwar, K., and Tardos, E., "An approximate truthful mechanism for combinatorial auctions with single parameter agents," in *SODA*, pp. 205–214, 2003.

[5] Archer, A. and Tardos, E., "Truthful mechanisms for one-parameter agents," *FOCS*, 2001.

[6] Arora, S., Lund, C., Motwani, R., Sudan, M., and Szegedy, M., "Proof verification and hardness of approximation problems," in *FOCS*, pp. 14–23, 1992.

[7] Bansal, N., Fleischer, L., Kimbrel, T., Mahdian, M., Schieber, B., and Sviridenko, M., "Further improvements in competitive guarantees for QoS buffering," in *ICALP*, vol. 3142 of *LNCS*, pp. 196–207, Springer, 2004.

[8] Bar-Yossef, Z., Hildrum, K., and Wu, F., "Incentive-compatible online auctions for digital goods," *SODA*, 2002.

[9] Bartal, Y., Gonen, R., and Nisan, N., "Incentive compatible multi unit combinatorial auctions," in *TARK*, pp. 72–87, 2003.

[10] Blumrosen, L. and Nisan, N., "On the computational power of ascending auctions." Manuscript, 2004.

[11] Borgs, C., Chayes, J., Immorlica, N., Mahdian, M., and Saberi, A., "Multi-unit auctions with budget-constrained bidders," 2004. Manuscript.

[12] Clarke, E. H., "Multipart pricing of public goods," *Public Choice*, 1971.

[13] Cramton, P., Shoham, Y., and Steinberg(editors), R., *Combinatorial Auctions*. MIT Press, Forthcoming (2005).

[14] Devanur, N. and Vishnoi, N., "Private communication," 2003.

[15] Dobzinski, S., Nisan, N., and Schapira, M., "Approximation algorithms for combinatorial auctions with complement-free bidders." Manuscript available at http://www.cs.huji.ac.il/ noam/mkts.html, 2004.

[16] Feige, U., "A threshold of lnn for approximating set cover," *Journal of the ACM*, vol. 45, no. 4, pp. 634–652, 1998.

[17] Fiat, A., Goldberg, A., Hartline, J., and Karlin, A., "Competitive generalized auctions," *STOC*, 2002.

[18] Freund, Y. and Schapire, R., "Adaptive game playing using multiplicative weights," *Games and Economic Behavior*, vol. 29, pp. 79–103, 1999.

[19] Gilboa, I. and Zemel, E., "Nash and correlated equilibria: Some complexity considerations," *Games and Economic Behavior*, vol. 15, no. 5, pp. 745–770, 1989.

[20] Goemans, M. and Kleinberg, J., "An improved approximation algorithm for the minimum latency problem," *Mathematical Programming*, vol. 82, pp. 111–124, 1998.

[21] Goldberg, A., Hartline, J., Karlin, A., Wright, A., and Saks, M., "Competitive auctions," 2003.

[22] Goldberg, A., Hartline, J., and Wright, A., "Competitive auctions and digital goods," *SODA*, 2001.

[23] Groves, T., "Incentives in teams," *Econometrica*, 1973.

[24] Gul, F. and Stacchetti, E., "Walrasian equilibrium with gross substitutes," *Journal of Economic Theory*, vol. 87, pp. 66–95, 2000.

[25] Henzinger, M., "Personal communication," 2004.

[26] Hirsch, M. D., Papadimitriou, C. H., and Vavasis, S. A., "Exponential lower bounds for finding brouwer fixed points," *Journal of Complexity*, vol. 5, pp. 379–416, 1989.

[27] Hoeffding, W., "Probability inequalities for sums of bounded random variables," *American Statistical Journal*, pp. 13–30, March 1963.

[28] Jain, K., Mahdian, M., Markakis, E., Saberi, A., and Vazirani, V., "Greedy facility location algorithms analyzed using dual fitting with factor-revealing lp," *J. ACM*, 2003.

[29] Jain, K., Mahdian, M., and Saberi, A., "A new greedy approach for facility location problems," in *STOC*, pp. 731–740, 2002.

[30] Jurg, A. P., "Some topics in the theory of bimatrix games." www.ub.rug.nl/eldoc/dis/non-rug/a.p.jurg/.

[31] Kalyanasundaram, B. and Pruhs, K., "On-line network optimization problems," in *Developments from a June 1996 seminar on Online algorithms*, pp. 268–280, Springer-Verlag, 1998.

[32] Kalyanasundaram, B. and Pruhs, K. R., "An optimal deterministic algorithm for online b -matching," *Theoretical Computer Science*, vol. 233, no. 1–2, pp. 319–325, 2000.

[33] Karp, R., Vazirani, U., and Vazirani, V., "An optimal algorithm for online bipartite matching," in *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, 1990.

[34] Kearns, M. J., Littman, M. L., and Singh, S. P., "Graphical models for game theory," in *UAI*, pp. 253–260, 2001.

[35] Kearns, M. J. and Mansour, Y., "Efficient nash computation in large population games with bounded influence," in *UAI*, 2002.

[36] Koller, D. and Megiddo, N., "Finding mixed strategies with small support in extensive form games," *International Journal of Game Theory*, vol. 25, pp. 73–92, 1996.

[37] Koller, D., Megiddo, N., and von Stengel, B., "Fast algorithms for finding randomized strategies in game trees," in *Annual ACM Symposium on the Theory of Computing*, pp. 750–759, 1994.

[38] Koller, D., Megiddo, N., and von Stengel, B., "Efficient computation of equilibria for extensive two-person games," *Games and Economic Behavior*, vol. 14, no. 2, pp. 247–259, 1996.

[39] Koutsoupias, E. and Papadimitriou, C. H., "Worst case equilibria," in *Annual IEEE Symposium on Theoretical Aspects of Computer Science*, pp. 404–413, 1999.

[40] Kuhn, H. W., "An algorithm for equilibrium points in bimatrix games.," in *Proceedings of the National Academy of Sciences*, pp. 1657–1662, 1961.

[41] Lavi, R., Mu'alem, A., and Nisan, N., "Towards a characterization of truthful combinatorial auctions," 2003.

[42] Lehman, B., Lehman, D., and Nisan, N., "Combinatorial auctions with decreasing marginal utilities," in *Proceedings of the 3rd ACM conference on Electronic Commerce*, pp. 18 –28, 2001.

[43] Lehmann, B., Lehmann, D., and Nisan, N., "Combinatorial auctions with decreasing marginal utilities," in *ACM Conference on Electronic Commerce*, 2001.

[44] Lehmann, D., O'Callaghan, L., and Shoham, Y., "Truth revelation in approximately efficient combinatorial auctions," in *ACM Conference on Electronic Commerce*, 1999.

[45] Lehmann, D., O'Callaghan, L., and Shoham, Y., "Truth revelation in approximately efficient combinatorial auctions," *JACM*, no. 49(5), 2002.

[46] Lemke, C. E., "Bimatrix equilibrium points and mathematical programming," *Management Science*, vol. 11, pp. 681–689, 1965.

[47] Lemke, C. E. and Howson, J. T., "Equilibrium points of bimatrix games," *Journal of the Society for Industrial and Applied Mathematics*, vol. 12, pp. 413–423, 1964.

[48] Littman, M. L., Kearns, M. J., and Singh, S. P., "An efficient, exact algorithm for solving tree-structured graphical games," in *NIPS*, pp. 817–823, 2001.

[49] Lund, C. and Yannakakis, M., "On the hardness of approximating minimization problems," *Journal of the ACM*, vol. 41, no. 5, pp. 960–981, 1994.

[50] Mahdian, M., Markakis, E., Saberi, A., and Vazirani, V., "A greedy facility location algorithm analyzed using dual fitting," *RANDOM-APPROX*, pp. 127–137, 2001.

[51] Mangasarian, O. L., "Equilibrium points of bimatrix games," *Journal of the Society for Industrial and Applied Mathematics*, vol. 12, no. 4, pp. 778–780, 1964.

[52] McEliese, R., Rodemich, E., Jr., H. R., and Welch, L., "New upper bounds on the rate of a code via the delsarte-macwilliams inequalities," *IEEE Trans. Inform. Theory*, pp. 157–166, 1977.

[53] McKelvey, R. and McLennan, A., "Computation of equilibria in finite games," *Amman, H., Kendrick, D., Rust, J. eds, Handbook of Computational Economics*, vol. 1, 1996.

[54] Mirrlees, J., "An Exploration into the Theory of Optimal Income Taxation," *Review of Economics Studies*, vol. 38, pp. 175–208, April 1971.

[55] Mu'alem, A. and Nisan, N., "Truthful approximation mechanisms for restricted combinatorial auctions," *AAAI-02*, 2002.

[56] Nash, J. F., "Non-cooperative games," *Annals of Mathematics*, vol. 54, pp. 286–295, 1951.

[57] Nisan, N. and Ronen, A., "Computationally feasible vcg mechanisms," *Electronic Commerce*, 2001.

[58] Nisan, N. and Segal, I., "The comminication requirements of efficient allocations and supporting lindahl prices." To appear in Journal of Economic Theory, preliminary version available at http://www.cs.huji.ac.il/ noam/mkts.html, 2004.

[59] Papadimitriou, C. and Yannakakis, "Optimization, approximation and complexity classes," *Journal of Computer and System Sciences*, vol. 43, pp. 425–440, 1991.

[60] Papadimitriou, C. H., "On the complexity of the parity argument and other inefficient proofs of existence," *Journal of Computer and System Sciences*, vol. 48, no. 3, 1994.

[61] Papadimitriou, C. H., "Algorithms, games, and the internet," in *Annual ACM Symposium on the Theory of Computing*, pp. 749–753, 2001.

[62] Raghavan, T. E. S., "Completely mixed strategies in bimatrix games," *Journal of London Math Society*, vol. 2, no. 2, pp. 709–712, 1970.

[63] Raz, R., "A parallel repetition theorem," *SIAM Journal of Computing*, vol. 27, no. 3, pp. 763–803, 1998.

[64] R.B.Myerson, "Optimal Auction Design," *Mathematics of Operations Research*, vol. 6, pp. 58–73, Feb. 1981.

[65] Ronen, A., "On approximating optimal auctions," *Electronic Commerce*, 2001.

[66] Ronen, A. and Saberi, A., "Optimal auctions are hard," *FOCS*, 2002.

[67] Rosenmuller, J., "On a generalization of the lemke-howson algorithm to noncooperative games," *SIAM Journal of Applied Mathematics*, vol. 21, pp. 73–79, 1971.

[68] Roughgarden, T. and Tardos, E., "How bad is selfish routing," in *Annual IEEE Symposium on Foundations of Computer Science*, pp. 93–102, 2000.

[69] Rubinstein, A., *Modeling Bounded Rationality.* Cambridge, Massachusetts: MIT Press, 1998.

[70] Sandholm, T., "An algorithm for optimal winner determination in combinatorial auctions," in *IJCAI*, 1999.

[71] Scarf, H., "The approximation of fixed points of a continuous mapping," *SIAM Journal of Applied Mathematics*, vol. 15, pp. 1328–1343, 1967.

[72] Simon, H., *Models of Bounded Rationality, Volume 2.* Cambridge, Massachusetts: MIT Press, 1982.

[73] Vickrey, W., "Counterspeculation, auctions and competetive sealed tenders," *J. Finance*, 1961.

[74] von Stengel, B., "Computing equilibria for two-person games," *Aumann, R. and Hart, S. eds, Handbook of Game Theory*, vol. 3, 2002.

[75] Wilson, I., "Computing equilibria of n-person games," *SIAM Journal of Applied Mathematics*, vol. 21, pp. 80–87, 1971.

[76] Yao, A. C., "Probabilistic computations: towards a unified measure of complexity," *FOCS*, pp. 222–227, 1977.

VITA

Education

Georgia Institute of Technology, Atlanta, GA

Algorithms, Combinatorics and Optimization.

Indian Institute of Technology, Mumbai, India

Bachelor of Technology, Computer Science and Engineering, August 2000.

Publications

1. Aranyak Mehta, Amin Saberi, Umesh Vazirani, Vijay Vazirani. Adwords and Generalized Online Matching. *To appear in IEEE Foundations of Computer Science (FOCS)*, 2005.

2. Richard Lipton, Evangelos Markakis, Aranyak Mehta, Nisheeth Vishnoi. On the Fourier Spectrum of Symmetric Boolean Functions with Applications to Learning Symmetric Juntas. *IEEE Conference on Computational Complexity (CCC)*, 2005.

3. Deeparnab Chakrabarty, Viswanath Nagarajan, Aranyak Mehta, Vijay Vazirani. Fairness in Congestion Games. *ACM Electronic Commerce*, 2005 (Full Version).

4. Aranyak Mehta, Vijay Vazirani. Randomized Truthful Auctions of Digital Goods are Randomizations over Truthful Auctions. *ACM Electronic Commerce*, 2004 (Full Version).

5. Parikshit Gopalan, Richard Lipton, Aranyak Mehta. Randomized Time Space Tradeoffs for Directed Graph Connectivity. *Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, 2003.

6. Richard Lipton, Evangelos Markakis, Aranyak Mehta. Playing Large Games using Simple Strategies. *ACM Electronic Commerce*, 2003 (Full Version).

7. Aranyak Mehta, Scott Shenker, Vijay Vazirani. Profit-Maximizing Multicast Pricing by Approximating Fixed Points. *ACM Electronic Commerce*, 2003. *Also to appear in Journal of Algorithms.*

8. Parikshit Gopalan, Howard Karloff, Aranyak Mehta, Milena Mihail, Nisheeth Vishnoi. Caching with Expiration Times. *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2002. *Also to appear in Internet Mathematics.*

9. Bharat Adsul, Aranyak Mehta, Milind Sohoni. Keeping Track of the Latest Gossip in Shared Memory Systems. *Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, 2000.

10. Howard Karloff, Subhash Khot, Aranyak Mehta, Yuval Rabani. On Earthmover Distance, Metric Labeling and 0-Extension, *(Manuscript)* 2005.

11. Mihalis Kolountzakis, Evangelos Markakis, Aranyak Mehta. On the Fourier Spectrum of Symmetric Boolean Functions and Learning Symmetric Juntas in time $n^{o(k)}$, *(Manuscript)* 2005.

12. Subhash Khot, Richard Lipton, Evangelos Markakis, Aranyak Mehta. Inapproximability Results for Combinatorial Auctions with Submodular Utility Functions, *(Manuscript)* 2005.

13. Kamal Jain, Aranyak Mehta, Vijay Vazirani. A Simple Characterization for Truth-revealing Single Item Auctions, *(Manuscript)* 2005.

14. Sanjiv Kapoor, Aranyak Mehta, Vijay Vazirani. An Auction-Based Market Equilibrium Algorithm for a Production Model, *(Manuscript)* 2005.