



ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Online truthful mechanisms (Άμεσοι φιλαλήθεις μηχανισμοί)

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ειρήνη Πογιατζή

Επιβλέπων: Ευστάθιος Ζάχος
Καθηγητής

Αθήνα, Οκτώβριος 2010



ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Online truthful mechanisms (Άμεσοι φιλαλήθεις μηχανισμοί)

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ειρήνη Πογιατζή

Επιβλέπων: Ευστάθιος Ζάχος
Καθηγητής

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 20^η Οκτωβρίου 2010

.....
Ευστάθιος Ζάχος
Καθηγητής

.....
Δημήτρης Φωτιάκης
Λέκτορας

.....
Άρης Παγουρτζής
Επίκουρος Καθηγητής

Αθήνα, Οκτώβριος 2010

.....
Ειρήνη Πογιατζή

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π

Copyright ©Ειρήνη Πογιατζή. 2010 Εθνικό Μετσόβιο Πολυτεχνείο.

Με επιφύλαξη παντός δικαιώματος. All rights reserved.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα.

Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

Abstract

In this thesis, we deal with limited supply and unlimited supply online auctions, as a subfield of online mechanism design. For the former case, we define a general model, on which we examine problems such as single-item and multi-item auction and also knapsack problem. We talk about the famous secretary problem and its applications-ideas to the previous auctions. Finally, we make a friendly introduction to matroids and their relationship with the online auctions where we have some restrictions (such as the size of the supply etc). For the latter case, we also define a general model and deal with randomized algorithms that manage to have good approximation ratio with respect to optimal fixed price revenue for both single-price and multi-price auctions. We also consider two cases for the model that depend on the information the auctioneer takes from the bidders after their acceptance or rejection of the product. Finally, we mention results for online pricing in other settings and give some implementations that evaluate the algorithms described in this thesis.

Acknowledgements

The research in this thesis would have taken far longer without the encouragement from many others. It is a delight to acknowledge those who have supported me over the last five years, during this long and intense endeavour, and I do not refer to the process of writing a dissertation, but the 5 years that took to get here. First of all, I would like to thank my supervisor, Prof. S. Zachos, for his guidance and relaxed, thoughtful insight. I am also heartily thankful to Prof. D. Fotakis for his persistence in helping me and for his remarkable demonstrations of the relevant existing results. This thesis was supported by the whole 3 member committee (S.Zachos,A.Pagourtzis,D.Fotakis) as well as by all members of the Computation and Reasoning Laboratory (A.Galanis,A.Gobel, G.Kaouri, P.Koutris, T.Liameas, E.Mpakalh, I.Panageas, G.Pierrakos, V.Syrgkanis) without either of which this work would not have been possible. I am particularly thankful for the help and advice of my friends I.Panageas, Th.Douvropoulos during my first few months, without whom the learning curve would have been very much steeper. The days would have passed far more slowly without the support of my undergraduate friends, Giannis, Thanasis, Theodosis, Nikos and many others, both academic and not, whom I thank for putting up with my idiosyncrasies and for providing such a rich source of conversation, education and entertainment. I cherish our many holidays together and hope that we will be able to stay in touch despite the large distances between us. Several people have played a decisive role in saving me from leaving this tough field. I am indebted to those at the CoreLab who welcomed me at such short notice, and to the many members of staff at E.M.P. who encouraged me on this journey. To all the above individuals, and to several colleagues whose names I cannot continue listing and who have assisted me one way or another, especially in challenging me with alternative views, I feel very much indebted. Finally, i wish to thank my parents for their love and encouragement, without whom I would never have enjoyed so many opportunities. While I believe that all of those mentioned have contributed to an improved final dissertation, none is, of course, responsible for remaining weakness. Last but not least, never enough thanks to one who doesn't want to be specially named but he knows who is and so do I.

Thank you all.

15/10/2010, Irene Poyiatzi

Contents

1 Introduction	9
1.1 Introduction to Game Theory	9
1.1.1 Basic Definitions	9
1.1.2 Examples	12
1.2 Introduction to Mechanism Design	16
1.2.1 Introduction	16
1.2.2 Definitions	16
1.3 Auctions - VCG mechanism	20
1.3.1 Model	20
1.3.2 Vickrey's second price auction	21
1.4 Online Algorithms	22
1.4.1 Introduction	22
1.4.2 Competitive ratio	22
1.4.3 Problems	23
1.5 Online Mechanism Design	24
2 Limited supply online auctions	27
2.1. Introduction	27
2.1.1 Basic Definitions	28
2.1.2 Secretary Problem	29
2.2 Online auctions with limited supply	30
2.2.1 Single item online auction	31
2.2.2 Multi-item auction	33
2.2.3 Knapsack Secretary Problem	35
2.3 Matroids	37
2.3.1 Definitions	37
2.3.2 Algorithmic Model	38
2.3.3 Importance of matroid structure	38
2.3.4 Logarithmically competitive algorithm for general matroids	40
3 Digital Goods	45
3.1 Introduction	45
3.2 Limited Supply Auctions	45
3.2.1 Basic Definitions	45
3.2.2 Competitive ratio benchmark	47
3.3 Truthfulness	47
3.3.1 Bid-independent	47
3.3.2 The deterministic approach	48
3.4 Randomized sampling	49
3.4.1 Introduction	49
3.4.2 Random Sampling Optimal Threshold Auction	49
3.5 Online case	50
3.5.1 Online Weighted Buckets Auction	51
3.5.2 WM Algorithm	53
3.5.3 Posted price auctioning - partial information case	54
3.6 Conclusion	56

4 Other settings for pricing in online auctions	57
4.1 Supply Curves	57
4.1.1 Incentive Compatibility for Supply Curves	58
4.1.2 Competitive Analysis	59
4.2 Expiring Items	60
4.2.1 Definition	60
4.2.2 Approximation ratio for deterministic truthful mechanism	61
4.2.3 Semi-Myopic Allocation Rules	62
4.3 Graph Vertex Pricing	63
4.3.1 Definition	63
4.3.2 The Online k-Hypergraph Vertex Pricing	64
Appendix A	69

Chapter 1

Introduction

1.1 Introduction to Game Theory

1.1.1 Basic Definitions

Game theory, known as a branch of applied mathematics that is used in the social sciences, most notably in economics, as well as in biology, engineering, political science, international relations, computer science, and philosophy, attempts to mathematically capture behavior in strategic situations, in which an individual's success in making choices depends on the choices of others, in other words it aims to model situations in which multiple participants interact or affect each other's outcomes.

While initially developed to analyze competitions in which one individual does better at another's expense (zero sum games), it has been expanded to treat a wide class of interactions, which are classified according to several criteria. Today, "game theory is a sort of umbrella or 'unified field' theory for the rational side of social science, where 'social' is interpreted broadly, to include human as well as non-human players (computers, animals, plants)".

Traditional applications of game theory attempt to find equilibria in these games. In an equilibrium, each player of the game has adopted a strategy that they are unlikely to change. Many equilibrium concepts have been developed in an attempt to capture this idea. These equilibrium concepts are motivated differently depending

on the field of application, although they often overlap or coincide. This methodology is not without criticism, and debates continue over the appropriateness of particular equilibrium concepts, the appropriateness of equilibria altogether, and the usefulness of mathematical models more generally.

As a matter of fact, in game theory the model regards rational, in the sense of self-interest, and intelligent, namely aware of all the existing knowledge of the game and capable of making all the logical inferences, agents.

Here, we will consider the well-known class of one-shot simultaneous move games, games in which all players simultaneously chose an action from their set of possible strategies. Specifically, such a game consists of a finite set of players, each one having his own set of possible strategies and the obligation to select one of them (deterministically or not), which will determine the -generally different- outcome for him. It should be noted that each player has an explicit preference ordering on these outcomes, playing the crucial role for his strategy decision.

Formally, elements of the game:

- a set n of players, $\{1, 2, \dots, n\}$.
- a set of possible strategies for each player i , S_i .
- individual's selected strategy, $s_i \in S_i$.
- set of all possible ways in which players can pick strategies. $S = \times_i S_i$.
- vector of strategies selected by the players $s \in S = (s_1, \dots, s_n)$
- generally different outcomes for different players depending on s .
- a preference ordering for each player on these outcomes by a complete, transitive, reflexive binary relation on the set of all strategy vectors S .

Let's name and slightly analyze some basic solution concepts - specifying the notion of stability which is the final disirata- that can help us with studying various kinds of games, some of them we will describe afterwords.

A desirable game-theoretic solution is one in which individual players act in accordance with their incentives, maximizing their own payoff. This idea is best captured by the notion of a *Nash equilibrium*, which, despite its shortcomings, has emerged as the central solution concept in game theory, with extremely diverse applications. The *Nash equilibrium* captures the notion of a stable solution, a solution from which no single player can individually improve his or her welfare by deviating.

It is a solution concept of a game involving two or more players, in which each player is assumed to know the equilibrium strategies of the other players, and no player has anything to gain by changing only his or her own strategy unilaterally. If each player has chosen a strategy and no player can benefit by alternating his or her strategy while the other players keep theirs unchanged, then the current set of strategy choices and the corresponding payoffs constitute a Nash equilibrium. Observe that such a solution is self-enforcing in the sense that once the players are playing such a solution, it is in every player's best interest to stick to his or her strategy.

However, Nash equilibrium may not be unique and also does not necessarily mean the best cumulative payoff for all the players involved; For games with multiple Nash equilibria, different equilibria can have (widely) different payoffs for the players. In many cases all the players might improve their payoffs if they could somehow agree on strategies different from the Nash equilibrium (e.g., competing businesses forming a cartel in order to increase their profits).

Definition 1: A strategy vector $s \in S$ is said to be a Nash equilibrium if for all players i and each alternate strategy $s_{-i} \in S_i$, we have that:

$$u_i(s_i, s_{-i}) \geq u_i(s_{-i}, s_{-i})$$

(for a strategy vector $s \in S$, s_i denotes the strategy played by player i and s_{-i} denotes the $(n - 1)$ -dimensional vector of the strategies played by all other players. Moreover, for each player i , u_i is simply a function of s_i , the strategy chosen by player i , rather than s , the strategies chosen by all n players. Now we obtain n independent optimization problems, which means that the payoff of each player depends only on his own strategy and not also on the strategies chosen by all other players. $u_i(s)$ denotes the utility incurred by player i . The notation $u_i(s_i, s_{-i})$ when is more convenient.)

Nevertheless, games sometimes possess dominant strategy solutions, a stringent and less widely applicable solution concept. In game theory, strategic dominance occurs when one strategy is better than another strategy for one player, no matter how that player's opponents may play. Many simple games can be solved using dominance. The opposite, intransitivity, occurs in games where one strategy may be better or worse than another strategy for one player, depending on how the player's opponents may play.

If a strictly dominant strategy exists for one player in a game, that player will play that strategy in each of the game's Nash equilibria. If both players have a strictly dominant strategy, the game has only one unique Nash equilibrium. However, that

Nash equilibrium is not necessarily Pareto optimal, meaning that there may be non-equilibrium outcomes of the game that would be better for both players. The classic game used to illustrate this is the Prisoner's Dilemma. Strictly dominated strategies cannot be a part of a Nash equilibrium, and as such, it is irrational for any player to play them. On the other hand, weakly dominated strategies may be part of Nash equilibria.

It is important to notice that a dominant strategy solution may not give an optimal payoff to any of the players. Having a single dominant strategy for each player is an extremely stringent requirement for a game and very few games satisfy it. On the other hand, mechanism design, aims to design games that have dominant strategy solutions, and where this solution leads to a desirable outcome (either socially desirable, or desirable for the mechanism designer).

Definition 2: A strategy vector $s \in S$ is a dominant strategy solution, if for each player i , and each alternate strategy vector $s_- \in S$, we have that:

$$u_i(s_i, (s_-)_{-i}) \geq u_i((s_-)_i, (s_-)_{-i})$$

It is not the objective of this analysis to demonstrate all the possible scenarios. Whilst, for completeness, we should mention that there are cases where the players are allowed to randomize and each player pick each of his strategies with some probability (non-deterministically), there are cases where no-nash equilibrium can be found, there are cases where we are able to reach a strong nash equilibrium in the sense that even no subset of players has a way to simultaneously change their strategies, improving each of the participant's welfare, there are cases where players have limited information and so we need to consider strategies that are only based on the available information, and find the best strategy for the player, given all his or her available information etc.

1.1.2 Examples

Now let's examine some representative examples of games where the players' optimal selfish strategies depend on what the other players play or there exist dominant strategies, or even there is only a weak notion of nash equilibrium.

Tragedy of the commons (Nash Equilibrium Example)

Borrowing some terms from the "economical crisis" that has been such a hot topic

in game theory circles nowadays we can consider a general product and manufacturing companies in relation to the consuming abilities of the working class. We make the assumption that the companies' profit depends on the amount of units they produce but also sinks as the demand of the product falls. That is as the ability of the general public to buy the product diminishes. An obvious candidate for a strategy would be for each of the selling companies to maximize their profit independently of the other players actions. Let's model the situation:

x_i would be the production (choose your imaginary unit) of the i th company and in order to take into account the finite spending capabilities of the masses we calculate $x_i \cdot (1 - s - x_i)$ the real gain of the company (s being the sum of the other players productions); we call this the utility function of the i th player. This is a quadratic function and analysis dictates $x_i = (1 - s)/2$ maximizes it. If all companies abided by the selfish optimal strategy we would get an end state where $x_i = (1 - s_i)/2$ where $s_i = \sum_{j, j \neq i} x_j$. The only solution of this matrix is given by $x_i = \frac{1}{n+1}$. In that case the sum of the utility functions (let's say the total well being of the corporate giants) is $\sum_j \frac{1}{n+1} \cdot (1 - \frac{n-1}{n+1} - \frac{1}{n+1}) = \sum_j \frac{1}{(n+1)^2} = \frac{n}{(n+1)^2}$ very close to $1/n$ a very slim prospect indeed.

If on the other hand the players agreed to negotiate beforehand and take under consideration the actions of the other players on the terrain things could turn out fashionably different. Let's say for instance they agree to balance their production, that is x_i is the same for all i 's. Then the communal utility value is

$$nx(1 - (n - 1)x - x) = nx(1 - nx) = nx - (nx)^2 = \theta - \theta^2$$

and since $\theta < 1$ this maximizes for $\theta = 1/2$ and raises the social welfare to a respectable $1/4$.

The reasoning of unlimited production with no respect of market demand, the pyramid looking "make more, sell all" schemes played an important role in the events leading to the great depression. Of course no genius is needed to reach the conclusion that consuming capabilities are limited but on the other hand no game-theorists were in charge those days.

Unscrupulous diner's dilemma (Dominant-strategy example)

The Unscrupulous diner's dilemma (or just Diner's dilemma) is an n -player Prisoner's dilemma. The situation imagined is several individuals go out to eat, prior to ordering they agree to split the check equally between all of them (They leave the receipts for the lucky one). Each individual must now choose whether to order the expensive or inexpensive dish. It is presupposed that the expensive dish is better than the cheaper, but not by enough to justify paying the difference compared to eating alone. Each individual reasons that the expense they add to their bill by ordering the more expensive item is very small (since they burdened their friends

with it), and thus the improved dining experience is worth the money. However, due to lack of universal information every individual reasons this way and they all end up paying for the cost of the more expensive meal, which, by hypothesis, is worse for everyone than ordering and paying for the cheaper meal.

Let g represent the joy of eating the expensive meal, b the joy of eating the cheap meal, h is the cost of the expensive meal, l the cost of the cheap meal, and n the number of players. From the description above we have the following ordering $h > g > b > l$. So as to facilitate for the player's reasoning that the shared expense is no longer a "no go" we give the numbers the extra property that:

$$g - b > \frac{h - l}{n}$$

Imagine an arbitrary set of strategies by a player's opponent. Let the total cost of the other player's meals be x . The cost of ordering the cheap meal is $\frac{1}{n}(x + l)$ and the cost of ordering the expensive meal is $\frac{1}{n}(x + h)$. So the utilities for each meal are $g - \frac{1}{n}(x + h)$ for the expensive meal and $b - \frac{1}{n}(x + l)$ for the cheaper meal. By hypothesis the utility of ordering the expensive meal is higher. Remember that the choice of opponents' strategies was arbitrary and that the situation is symmetric. This proves that the expensive meal is strictly dominant and thus the unique Nash equilibrium. If everyone orders the expensive meal all of the diners pay h and their total utility is $g - h < 0$. On the other hand suppose that all the individuals had ordered the cheap meal, their utility would have been $b - l > 0$. Once again, everyone is worse off by playing the unique equilibrium than they would have been if they collectively pursued another strategy.

The pirate game (a weak notion of Nash Equilibrium)

The *pirate game* is a multi-player version of the ultimatum game. There are five rational pirates, A, B, C, D and E . They find 100 gold coins. They must decide how to distribute them. The pirates obey an authority ordering: A is superior to B , who is superior to C , who is superior to D , who is superior to E . The pirate world's rules of distribution are thus: that the most senior pirate should propose a distribution of coins. The pirates, including the proposer, then vote on whether to accept this distribution. If the proposal is approved by a majority or a tie vote, it happens. If not, the proposer is thrown overboard from the pirate ship and (hopefully) dies, and the next most senior pirate makes a new proposal to begin the system again. Pirates (being rational beings) base their decisions on three factors. First of all, each pirate wants to survive. Secondly, each pirate wants to maximize the number of gold coins he receives. Thirdly, each pirate would prefer to throw another overboard, even if all other results would otherwise be equal -they are simply mean.

The game-theorist's chest: It might be expected intuitively that Pirate A will have

to allocate little if any to himself for fear of being voted off so that there are fewer pirates to share between. However, this is as far from the theoretical result as is possible. This is apparent if we work backwards: if all except D and E have been thrown overboard, D proposes 100 for himself and 0 for E . He has the casting vote, and so this is the allocation. If there are three left (C , D and E) C knows that D will offer E 0 in the next round; therefore, C has to offer E 1 coin in this round to make E vote with him, and get his allocation through. Therefore, when only three are left the allocation is $C : 99, D : 0, E : 1$. If B, C, D and E remain, B knows this when he makes his decision. To avoid being thrown overboard, he can simply offer 1 to D . Because he has the casting vote, the support only by D is sufficient. Thus he proposes $B : 99, C : 0, D : 1, E : 0$. One might consider proposing $B : 99, C : 0, D : 0, E : 1$, as E knows he won't get more, if any, if he throws B overboard. But, as each pirate is eager to throw each other overboard, E would prefer to kill B , to get the same amount of gold from C . Assuming A knows all these things, he can count on C and E 's support for the following allocation, which is the final solution:

- A : 98 coins
- B : 0 coins
- C : 1 coin
- D : 0 coins
- E : 1 coin

Also, $A : 98, B : 0, C : 0, D : 1, E : 1$ or other variants are not good enough, as D would rather throw A overboard to get the same amount of gold from B .

The pirates game illustrates the idea of *nash equilibrium* in a very weak form. Here we clearly have no rivaling strategies but the sole dominance of one of the players. Similar types of games that are turn-based (that is, players differentiate their course of action after each move) need a tree-form strategy, a set of responses for every possible state of the game. It is there that we have to search for equilibrium. Of course these strategy-trees are often enormous even in computational terms (think of chess for example).

1.2 Introduction to Mechanism Design

1.2.1 Introduction

Mechanism design (sometimes called reverse game theory) is the intersection of the field of micro-economics (utility maximization and mechanism design) and the field of game theory (rationality and Nash equilibrium). If we wish to add to it concepts as complexity and algorithm design from discrete mathematics and worst case analysis and approximation ratios from theoretical computer science, then we are talking about algorithmic mechanism design.

In essence, *Mechanism design* attempts implementing desired social choices (aggregations of the preferences of the different participants toward a single joint decision, namely generalization of scenarios such as elections, markets, auctions, government policy) in a strategic setting - assuming that the different members of society each act rationally in a game theoretic sense. Such strategic design is necessary since usually the preferences of the participants are private. That means that it is studying solution concepts for the class of private information games (Bayesian games), whose distinguishing feature is firstly that a game "designer" chooses the game structure rather than inheriting one and secondly that the designer is interested in the game's outcome, and it usually solves them by motivating agents to disclose their private information.

1.2.2 Definitions

In this section we will give some basic definitions that concern mechanism design. Let n be the set of agents and i an arbitrary agent. We denote by $\theta_i \in \Theta_i$ the type of agent i , which shows i 's preference profile over the possible outcomes, assume O . For example if we have a set A of m candidates, we consider of Θ_i as the set of linear orders on A (let L) and θ_i as a specific linear order on A , let $\prec_i \in L$ (i 's preference over the candidates, where $a \prec_i b$ means voter i prefers b to a). Obviously $A = O$. For the case of voting, when we introduce the theorem of Arrow and Gibbard-Satterwaite, we will use the notation A, L, \prec .

In order to measure the happiness of agent i for an outcome $o_j \in O$, we de-

fine the utility function u_i .

Definition 1: Let $u_i : \theta_i \times O \rightarrow R$ be the utility function of i . We consider $u_i(\theta_i, o_j) > u_i(\theta_i, o_k)$ if i prefers outcome o_j to o_k .

Considering the previous example, if $a \prec_i b$ then $u_i(\theta_i, b) > u_i(\theta_i, a)$.

Definition 2: A function $f : \times_i \Theta_i \rightarrow O$ is called a social choice function.

Below, we define what a mechanism, which shows formally mechanism design relation with the game theory model.

Definition 3: A mechanism $M = (S_1, \dots, S_n, g(s))$ defines the set of strategies S_i for agent i and g is a function that denotes the outcome of M , that is $g : S_1 \times \dots \times S_n \rightarrow O$ and $g(s)$ is the outcome, where $s = (s_1, \dots, s_n)$ the vector of strategies of the agents.

Thus, the strategy of agent i s_i , is a function that depends on his type θ_i . The implementation of a social function f , given a mechanism M , we have that $f(\theta_1, \dots, \theta_n) = g(s_1(\theta_1), \dots, s_n(\theta_n))$ for every vector of types. In this thesis, we are interested in problems where the agents' strategies consider the reporting of their type. Hence, we search solutions that truthtelling is a dominant strategy, that is $(s_1(\theta_1), \dots, s_n(\theta_n))$ is a dominant strategy where $s_i(\theta_i) = \theta_i$.

There are two well-known theorems that concern the impossibility of achieving "reasonable" voting, Arrow's and Gibbard-Satterthwaite, but first, we need to mention some properties social choice function f must satisfy (in order to make the voting "reasonable").

A social welfare function must satisfy the following to have "reasonable" voting:

- Function f satisfies *unanimity* if for every $\prec \in L$, $f(\prec, \dots, \prec) =_{arg\,first} \prec$. intuitively, if all voters have the same preference, the result of the voting must be the first candidate of the preference.
- Function f is not a *dictatorship*, which means there is no voter i such that $f(\prec_1, \dots, \prec_n) =_{arg\,first} \prec_i$, for all $\prec_1, \dots, \prec_n \in L$.
- Function f satisfies *independence of irrelevant alternatives*, which means for every $a, b \in A$ and every $\prec_1, \dots, \prec_n, \prec'_1, \dots, \prec'_n$, if we denote $a = f(\prec_1, \dots, \prec_n)$ and the relevant position of a, b remains unchanged in the two profile preferences then $b \neq f(\prec'_1, \dots, \prec'_n)$ that is b can't be a winner.

It's been proven by Arrow the following theorem (Nisan et al. [21] to see the proof):

Theorem: Every social choice function over A , with $|A| \geq 3$ that satisfies unanimity and independence of irrelevant alternatives, is a dictatorship.

Actually the proof in Nisan et al. [21] concerns a social welfare function, but the idea is the same and the proof is similar.

From another view, suppose a social choice function must satisfy the following in order to have "reasonable" voting:

- Function f is *incentive-compatible*, that is there is no voter i such that for some $\prec_1, \dots, \prec_n \in L$ and some $\prec'_i \in L$ we have that $a \prec_i a'$ where $a = f(\prec_1, \dots, \prec_i, \dots, \prec_n)$ and $a' = f(\prec_1, \dots, \prec'_i, \dots, \prec_n)$. Intuitively, there is no i that prefers a' to a and can ensure that a' gets socially chosen rather than a by misrepresenting his preferences.
- Function f is not a *dictatorship*, which means there is no i such that for all $\prec_1, \dots, \prec_n \in L$, $\forall b \neq a$, $b \prec_i a \Rightarrow f(\prec_1, \dots, \prec_n) = a$.

It's been proven by Gibbard and Satterthwaite independently the following theorem (Nisan et al. [21] to see the proof):

Theorem: There is no f that is incentive compatible social choice function onto A and is not a dictatorship.

In order to surpass the previous results of Arrow and Gibbard-Satterthwaite, we put some restrictions to our problems. An idea is to use money (see auctions) in order to achieve "reasonable" and incentive compatible mechanisms. Another way to deal with that impossibility result is randomization, where we search for the *expected* utility for each agent (mechanism makes nondeterministic choices) and the *expected* welfare.

Mechanisms with Money:

In this section, we have a set of alternatives A and a set of n players I . Each player's preference is expressed with a valuation function $v_i : A \rightarrow R$, where $v(a)$ shows i 's assignment "value" if a is chosen. If player a is also given an amount m of money, i 's utility function (function that player i intends to maximize) is denoted

$$u_i = v_i + m$$

Utilities of this form are called *quasilinear preferences*.

Definition 4: A (direct revelation) mechanism is a social choice function $f : V_1 \times \dots \times V_n \rightarrow A$ and a vector of payment functions p_1, \dots, p_n where $p_i : V_1 \times \dots \times V_n \rightarrow R$

is the amount of money player i pays.

In words, the mechanism restricts agents strategies to their types-preferences that are expressed with the valuation functions and also decides about the payment of each agent (which depends on their type).

Most of the times, when we are searching for mechanisms to solve an algorithmic game theory problem, we want our mechanism to be truthful and truthfulness be a dominant strategy.

Definition 5: A mechanism with money (f, p_1, \dots, p_n) is called incentive compatible if for every i , every $v_1 \in V_1, \dots, v_n \in V_n$ and every $v'_i \in V_i$, the following inequality holds:

$$v_i(a) - p_i(v_i, v_{-i}) \geq v_i(a') - p_i(v'_i, v_{-i})$$

where $a = f(v_i, v_{-i})$ and $a' = f(v'_i, v_{-i})$.

Although it sounds strange, any general mechanism that implements a function in dominant strategies can be converted into an incentive compatible one. Formally, we have the following theorem.

Revelation Principle

Theorem: If there exists an arbitrary mechanism that implements f in dominant strategies, then there exists an incentive compatible mechanism that implements f . The payments of the players in the incentive compatible mechanism are identical to those obtained at equilibrium, of the original mechanism.

Mechanisms without Money:

As opposed to the previous paragraph, in mechanism design without money, we search for social choice function without the presence of the payment vector. Even though the existence of Gibbard-Satterthwaite theorem described above, we examine dominant strategy implementation on restricted domains of preferences. It is remarkable to mention that in this case a direct revelation mechanism also restricts the strategies of agents to their types and the revelation principle still holds (we omit anything that has to do with payments). In this thesis, we are not interested in analyzing the mechanisms without money, thus we will just mention a problem and a strategy proof mechanism that deals with it. A well-known example is house allocation problem. House allocation involves a set of n agents, each owning a unique house and a strict preference ordering over all n houses. Our goal is to reallocate the houses among the agent in an "appropriate" way. Specifically, we search for an allocation rule that is strategy-proof. The difference in this problem

is that the set of alternatives and agents actually are the same. A strategy-proof mechanism is the following:

Top Trading Cycle Algorithm: We construct a directed graph, where each vertex presents an agent. Suppose house j is agent i 's k -th ranked choice, then insert an edge (i, j) with color k . Notice that the set of cycles, loops with the same color are node disjoint. At step k let N_k be the set of vertices incident to cycles with color k . For each cycle C , take all edges of C (i_l, i_{l+1}) and give house i_{l+1} to agent i_l . Delete all edges with color k and repeat. It's proven that the algorithm (Nisan et al. [21]) above is strategy-proof, namely no agent takes a more preferable house if he deviates.

1.3 Auctions - VCG mechanism

1.3.1 Model

Consider an auction of an item that takes place among n bidders. Each bidder i has a value w_i that shows how much "money" i is willing to pay. Suppose that bidder i wins the auction and he buys the item at some price p . Then his utility is $w_i - p$. If someone else (not i) wins the auction, then his utility is 0. Our goal is to design a mechanism that allocates the item - chooses the winner bidder in a way that the auction cannot be strategically manipulated. Moreover, we want in a way to maximize the social welfare:

$$\sum_i u_i$$

An approach to solve this problem we will discuss in section 1.3.2 below, the VCG mechanism. There are also auctions with k identical items that each bidder is interested in buying only one (k Vickrey auction). Furthermore, there are auctions where each bidder is interested in buying a set of items and the problem is to price the items. Finally, other modifications of auctions can be occurred if each bidder i has arriving and departure time (i must buy an item before he departs), etc. Vickrey (1961) gives a celebrated result that any member of a large class of auctions assures the seller of the same expected revenue and that the expected revenue is the best the seller can do. This is the case if:

1. The buyers have identical valuation functions (which may be a function of type)
2. The buyers' types are independently distributed

3. The buyers types are drawn from a continuous distribution
4. The type distribution bears the monotone hazard rate property
5. The mechanism sells the good to the buyer with the highest valuation

1.3.2 Vickrey's second price auction

Assume we have an auction of a single item, as described above and a set I of n bidders. The set of alternatives is the winners of the auction. Formally $A = \{i - \text{wins} \mid i \in I\}$. Also, the valuation of each bidder is denoted by $v_i(i - \text{wins}) = w_i$ (how much he wills to pay for the item) and $v_i(j - \text{wins}) = 0$, $j \neq i$ (intuitively, every bidder wants to win). The algorithm that solves the latter problem is the following:

Let the winner be the player i with the highest bid (w_i) and i pays p , which is the second highest declared bid, $p = \max_{j \neq i} w_j$. The following theorem holds (incentive compatible mechanism).

Theorem: For every w_1, \dots, w_n and every w'_i , let u_i be i 's utility if he bids w_i and u'_i his utility if he bids w'_i . Then $u_i \geq u'_i$.

PROOF: Assume that by saying w_i he wins and the second highest (reported) value is p^* , then $u_i = w_i - p^* \geq 0$. If i attempted to manipulate, then we consider two cases:

- $w'_i > p^*$, i would still win if he bids w'_i and would still pay p^* . Thus $u'_i = u_i$.
- $w'_i \leq p^*$, i would lose so $u'_i = 0 \leq u_i$.

If i loses by bidding w_i then $u_i = 0$. Let j be the winner, so $w_j \geq w_i$. We again consider two cases:

- For $w'_i < w_j$, i would still lose and so $u'_i = 0 = u_i$.
- $w'_i \geq w_j$, i would win and pay w_j , thus bidder's utility would be $u'_i = w_i - w_j \leq 0 = u_i$.

Notice that the algorithm above also maximizes the social welfare function (because the winner is the only bidder with utility $\neq 0$). The second price auction is a special case of the general *VCG* mechanisms. Generally, in *VCG* mechanisms (f, w_1, w_2, \dots, w_n) the following properties must hold:

- $f(w_1, \dots, w_n)$ maximizes $Rev = \sum_{j=1}^n w_j$ (social welfare).
- $p_i(w_1, \dots, w_n) = h_i(w_{-i}) - Rev(f(w_1, \dots, w_n)) + w_i(f(w_1, \dots, w_n))$, that is p_i doesn't depend on i 's marginal value function.

The idea is rather simple, for bidder i it occurs that $u_i = Rev - h_i(w_{-i})$, so in order bidder i to maximize his utility function, he must maximize the social welfare. The *VCG* mechanism is incentive compatible (similar proof to the proof of truthfulness of second price auction). To choose a "proper" h_i , we have to demand that $p_i \geq 0$ for every i . Thus choosing $h_i(w_{-i}) = \max_{b_i \in A} \sum_{j \neq i} w_j(b_i)$ (Clarke pivot payment), and letting $a \in A$ such that $Rev(a)$ is maximized we have that $p_i \geq \sum_{j=1}^n w_j(a) - w_j(b_i) \geq 0$ since $w_i \geq 0$. It is rather straightforward to observe that for the statement of the second price auction, we have that the winner pays the second largest price.

1.4 Online Algorithms

1.4.1 Introduction

Online Algorithms are a subfield of Algorithms where we are dealing with interactive computing. More specifically, in Online Algorithms, we are facing problems whose input arrives as a sequence of portions and the interactive system must react in response to each portion. Moreover, we consider that the future input is not known and we are interested in finding a solution that is close to the optimal *offline* solution. *Offline Algorithms* are more or less what we consider classic Algorithms. Formally, an online algorithm receives a sequence of requests $\sigma = \sigma(1), \dots, \sigma(m)$. These requests must be served in the order of occurrence. When serving request $\sigma(t)$, an online algorithm does not know requests $\sigma(t')$ with $t' > t$. Serving requests incurs cost and the goal is to minimize the total cost paid on the entire request sequence.

1.4.2 Competitive ratio

In order to analyze online algorithms, we compare the performance of an online algorithm with the optimal offline algorithm. This method is called *competitive analysis*. Assume that we have online algorithm ALG for a problem and the respective optimal offline algorithm OPT and input x . Let $ALG(x)$ and $OPT(x)$ be the cost of online and offline algorithm respectively. If the relation below holds for every input x :

$$ALG(x) \leq c \cdot OPT(x) + b$$

where b is a constant number, then ALG is c -competitive. It is remarkable that *competitive analysis* is a worst-case analysis measure.

1.4.3 Problems

There are online problems which have great interest over the years such as paging problem, selforganizing lists, the k -server problem as well as metrical task systems. In this section, we discuss the statement of k -server problem and results referred to competitive analysis for the latter problem.

k -server Problem

In this problem we have k servers and m requests which are represented as points in a metric space S . Let G be the graph that is formed of the locations of the requests. As each request arrives, the algorithm must determine which server to move to the requested point. The goal is to keep the total distance the servers move as small as possible. For competitive analysis, the respective offline problem is to know from the beginning the requests and move the servers optimally to serve them. Symmetric k -server problem is the k -server problem where $d(i, j) = d(j, i)$ for every points i, j (distance between i, j).

Theorem: For any symmetric k -server problem, there is no c -competitive deterministic algorithm for $c < k$.

PROOF: Let A be an online algorithm for k -server. Let H be a subgraph of G of size $k + 1$ induced by the k initial positions of A 's servers and one other vertex. Let $\sigma(i)$ be the unique vertex in H not covered by A at time i . Then the cost of A is:

$$C_A(\sigma, t) = \sum_{i=1}^t d_{\sigma(i+1), \sigma(i)}$$

Let S be any k -element subset of H containing $\sigma(1)$. We define $A(S)$ as follows: Initially, the servers occupy the vertices in S . At step i , if $\sigma(i) \in S$ do nothing, else move server from $\sigma(i - 1)$ to $\sigma(i)$ and update S . It's obvious when step i begins, $\sigma(i - 1) \in S$.

Assume now S_1, S_2 sets with the property of S that initially are not equal. We will prove that during the procedure, they never become equal (i).

Let j be an index such that S_1, S_2 differ before $\sigma(j)$ procedure. We examine the following cases:

- $\sigma(j) \in S_1, S_2$ then after updating, S_1, S_2 still differ (they don't change).
- $\sigma(j)$ belongs to exactly one of them. Then after updating, the one set contains $\sigma(j - 1)$, but not the other.
- $\sigma(j) \notin S_1, S_2$ then after updating, S_1, S_2 still differ ($\sigma(j)$ replaced $\sigma(j - 1)$ from both sets).

This completes the induction. Let us consider simultaneously running $A(S)$ starting from all k -elements subsets of H containing $\sigma(1)$. There are k such sets. At each step, there are all k -elements subsets (using i) of H . At step $i + 1$, each of these algorithms either does nothing at no cost, or it moves a server from $\sigma(i)$ to $\sigma(i + 1)$ at cost $d_{\sigma(i)\sigma(i+1)}$. From the k algorithms, one of them doesn't contain $\sigma(i + 1)$ (since we have all sets at each step). So at step i , the total cost of the k algorithms is $d_{\sigma(i)\sigma(i+1)}$. Thus the total cost of running all of these algorithms up to $\sigma(t)$ (including) is:

$$\sum_{i=1}^{t-1} d_{\sigma(i)\sigma(i+1)}$$

So the expected cost is $E = \frac{1}{k} \sum_{i=1}^{t-1} d_{\sigma(i)\sigma(i+1)}$ of the offline algorithms. Thus $C_A(\sigma, t) \geq k \cdot E$.

The Work Function:

Let X be a configuration of the servers. Given a request sequence $\sigma = \sigma(1), \dots, \sigma(t)$, the work function $w(X)$ is the minimal cost of serving σ and ending in configuration X . Suppose that the algorithm has served $\sigma = \sigma(1), \dots, \sigma(t - 1)$ and that a new request $r = \sigma(t)$ arrives. Let X be the current configuration of the servers and let x_i be the point where server s_i , $1 \leq i \leq k$, is located. Serve the request by moving the server s_i that minimizes $w(X_i) + d(x_i, r)$, where $X_i = X - \{x_i\} + \{r\}$.

Theorem: The Work Function algorithm is $(2k - 1)$ -competitive in an arbitrary metric space.

1.5 Online Mechanism Design

Sections 1.2, 1.4 are a friendly introduction to Mechanism Design and Online Algorithms. In this section we discuss the combination of these two fields. Suppose that we have a game theory problem and we search for incentive compatible mechanisms (truthful is a dominant strategy). However, in the statement of the problem, we have that the self-interested agents are dynamically arriving or departing, or there is uncertainty about the set of feasible decisions in the future. Online mechanism design, generalization of the theory of computational mechanism design deals with these kind of problems. Decisions must be made dynamically and without knowledge of future decision possibilities, in the sense of online algorithms. A well-known example is the *secretary problem* that we analyse in chapter 2. Moreover, agents can misrepresent except of their valuation, their arrival and departure time (it's usually unavailable to report an earlier than the true arrival time). Finally, for the worst-case analysis, we can also use competitive analysis, as defined in section

1.4.2 because we deal with mechanisms with online setting. Below, we give formally the general model.

Model: Consider discrete time periods $T = \{1, 2, \dots\}$ indexed by t . A mechanism makes decisions in time t that depend on events that took place in time $< t$. The events may be bids that are received from agents, or the loss of an item (if the time it expires passed), or uncertain events that don't have any relation with agents' types. The agents have a type of the form (a_i, d_i, w_i) that concern their arrival, departure time and valuation. The decisions that have to do with a certain agent, let i , must be made during the period $[a_i, d_i]$. Most of the times, we have some restrictions on the announced type of an agent, that is the agent is not allowed to announce every type he wants that is different from his true type (namely we may have no early-arrival misreports or no late-departure misreports). This is rather important, because without these assumptions we can't use the direct revelation principle in the online case (Nisan et al. [21]).

Definition 1: A (direct-revelation) on-line mechanism $M = (\pi(t), x(t))$ restricts each agent to make a single announce about his type and defines decision policy $\pi(t)$ and payment policy $x(t)$ happened in time t , where $t \in T$.

In words, $\pi(t)$ is the function that given the time t , denotes the decisions that must be made in t and $x(t)$ is the payment-rule for some agents that are present in t .

In Online Mechanism Design (as generally in mechanism design), we are interested in finding mechanisms that are dominant-strategy incentive compatible. In this case, the only thing that changes is that truth-telling must be dominant strategy for every set of events that occur in the dynamic environment. For the randomized mechanisms, we care about the *expected* utility to be maximized when the agents are truthful over all uncertain events that take place and over all the flip coins a randomized mechanism uses, namely, truth-telling is independent of the uncertain events.

The main part of this diploma thesis, deals with these kind of problems in mechanism design. In the next two chapters, we examine different modifications of online auctions that the field of online mechanism design deals with. In chapter 2, we deal with problems where we try to find strong-strategy proof mechanisms, that is agents truth-telling about their valuation, arrival and departure is a dominant strategy and have the constraint that the auctioneer has limited supply of the items he sells (the number of items is less than the number of agents). For this case, we consider the random order scenario. However, in chapter 3 we deal with problems with weak-strategy proofness, that is truth-telling about only their

valuation is a dominant strategy (we don't discuss about arrivals and departures of the agents). For this case, we consider the worst-case scenario.

Chapter 2

Limited supply online auctions

In this chapter we examine the auctions at which the auctioneer and the bidders deal with a limited supply of goods. Except of the internet, in real auctions, the auctioneer sells a finite number of goods. So it is very important to focus on this kind of auctions. First of all, we present an online algorithm for the well-known *Secretary problem*, the idea of which has a lot of applications with respect to the online auctions with limited supply. Furthermore, we give a common framework that all the problems of this chapter can be reduced to, such as single-item auction, multi-item auction, knapsack secretary. Finally, we make a friendly introduction to matroid domains and their relation to the previous problems.

2.1 Introduction

Assume that we are interested in selling an item with the following procedure. N bidders come one at a time and announce a bid that shows how willing they are to buy the item, namely how much "money" they are willing to pay. After hearing bidder i , we must immediately decide whether or not to sell the item to him. What is the best approach for the preceding problem? In order to give a right answer, there are a lot of questions that have to be answered first, such as, do we have any *a priori* knowledge of the bid values, or any assumptions with respect to the bids' order? Suppose for the contrary that we don't have any assumptions for the order. Then, in order to analyse our problem, an idea is the *worst-case* scenario, that is, we consider the input to be the worst permutation of bids according to the online algorithm we examine (for a given vector of bids). Unluckily, for this scenario the bound is rather pessimistic, as there is no online algorithm with com-

petitive ratio less than N . Another scenario, is to consider that the bidders come in random order, that is we compare the *expected* profit of the online algorithm with the optimal offline (the expectation is taken over all permutations of bids, for a given vector b_1, \dots, b_N). For the rest of this chapter, we make the assumption that this scenario holds and the analysis for all the different modifications of the online auctions we study is based on this scenario. The Secretary problem we discuss in section 2.1.2, is very similar to the initially given problem. In the first case, we are interested in maximizing the expected value of the chosen bid and in the Secretary problem we are interested in maximizing the probability of choosing the *maximum* element (highest applicant). It is rather straightforward to observe that the solution to the secretary problem yields an algorithm for choosing a bid from a randomly-ordered bid sequence (as described above). For now, we postpone the analysis of the problem because firstly, we have to give the necessary definitions and background knowledge.

2.1.1 Basic Definitions

As we have already mentioned, we consider an environment with N agents that want to allocate a single, indivisible item. Each agent's type is denoted $\theta_i = (a_i, d_i, w_i) \in \Theta$ (Θ is the set of types), where a_i, d_i, w_i denote the agent's i arrival time, departure time and value for the item respectively. Additionally, bidders are strategic and may try to manipulate the auction by misreporting their type. Assuming no early-arrival misreports and allowing all other kinds of misreports - namely, when bidder i announces his type, he can't report a'_i, d'_i, w'_i with $a'_i < a_i$ - we try to find a solution with a good revenue as far as the auctioneer is concerned. Revenue is defined by

$$Rev = \sum_i p_i(\theta_i)$$

where $p_i(\theta_i)$ is the payment of bidder i . In order to compare the online mechanisms with the offline, we have to define what a c -competitive mechanism means, with respect to efficiency and revenue.

Let S be the set of all inputs of the algorithm. As an example, consider that for the problem described in the introduction, S is all the permutations of the bid values. Assume an arbitrary input $s \in S$. If $E_{s \in S}[\frac{V_{online}(\theta_s)}{V_{opt-offline}(\theta_s)}] \geq \frac{1}{c}$, where V is an objective function, $V : \Theta \rightarrow R$ defined by the problem (value of the allocation, usually $\sum_i v_i$), we say that our algorithm is c -competitive with respect to efficiency.

It is remarkable to say that we take the expectation with respect to the random order of the elements of S .

In a similar way, if $E_{s \in S}[\frac{Rev_{online}(\theta_s)}{Rev_{Vick}(\theta_s)}] \geq \frac{1}{c}$ then our algorithm is c -competitive with respect to revenue, where Rev_{Vick} is the returned revenue from Vickrey's second price auction. It is remarkable to mention that for the revenue case, there are different definitions as far as the optimal offline revenue is concerned, that depend on the nature of the problem. Namely, if we deal with auctions where we have more than one item to sell, we define different revenues from Rev_{Vick} .

2.1.2 Secretary Problem

It is rather straightforward to see that the example described in introduction is familiar with the secretary problem, the statement of which is the following. An interviewer wants to hire one of N job applicants. The total number of them is known. Each applicant meets the interviewer in turn and informs him about his *quality*. If the interviewer denies to hire an applicant, he can't change his mind. The goal is to hire the best applicant. Actually, the goal is to maximize the probability of hiring the highest rank applicant, for all adversarially selected inputs. Ferguson in [12] describes the following mechanism to solve the secretary problem. For a given parameter r the policy is to interview the first $r - 1$ applicants and hire the i applicant ($i \geq r$) with the best quality so far, if there exists one (it is possible that for all $i \geq r$, there exists $j < i$ such that $quality_i < quality_j$). In his paper [1], he proves the theorem below.

Theorem: The probability of selecting the best applicant converges to $\frac{1}{e}$ as $N \rightarrow \infty$ for the optimal choice of r , which is $\lfloor \frac{N}{e} \rfloor$.

PROOF: Define $\phi_N(r)$ as the probability that the algorithm selects the best applicant after declining the $r - 1$ applicants and $\psi_N(j)$ as the probability that the j -th applicant is the best and the algorithm has declined the first $j - 1$ applicants. It is obvious that $\phi_N(r) = \sum_{j=r}^N \psi_N(j)$ because if the algorithm rejects the $j - 1$ applicants and the j -th is the best, then the algorithm will choose him ($j \geq r$). Now, let $p_M(j) = \frac{(M-1)!}{M!} = \frac{1}{M}$ be the probability that the j -th applicant is the best among M applicants and finally $q_M(j)$ be the probability the best applicant among the first M has index less than or equal to j . Then, $\psi_N(r) = p_N(r) = \frac{1}{N}$ (the probability that r -th applicant is the best) and $\psi_N(j) = p_N(j) \times q_{j-1}(r - 1)$ for $j > r$. Furthermore, apparently $q_M(j) = \sum_{t=1}^j p_M(t) = \frac{j}{M}$. Thus

$$\phi_N(r) = \sum_{j=r}^N p_N(j) \times q_{j-1}(r - 1) = \sum_{j=r}^N \frac{1}{N} \left(\frac{r-1}{j-1} \right) = \frac{r-1}{N} \sum_{j=r}^N \left(\frac{1}{j-1} \right)$$

As $N \rightarrow \infty$ then the sum can be approximated by the following integral (using x, t

and dt as limit of r/N , j/N and $1/N$ respectively).

$$x \int_x^1 \left(\frac{1}{t}\right) dt = -x \log x$$

The previous integral is maximized for $x = \frac{1}{e}$ (using derivatives). The integral is equal to $\frac{1}{e}$ and so is $\phi_N(r)$. It is remarkable to mention that r that maximizes the propability $\phi_N(r)$ is $\left\lfloor \frac{N}{e} \right\rfloor$ (substitutue x for r/N).

Formally, the algorithm for the secretary problem is the following:

<i>Algorithm for Secretary problem</i>
<ol style="list-style-type: none"> 1. Observe $r = \lfloor \frac{N}{e} \rfloor - 1$ without picking any element and let p be the highest quality among the first r elements. 2. for $i = r + 1, r + 2, \dots, n$ do 3. if $w_i \geq p$ then output w_i, break 4. end for.

However, except of the secretary problem, there are lots of modifications of the standard model (discussed in introduction and also in next section) and we examine them seperately. For the simple case, we assume the following constraints. Each bidder i has a type θ_i and is strategic, namely he may misreport his type to increase his utility function. The goal is to find a truthful, efficient and competitive mechanism in order to maximize the revenue of the auctioneer.

2.2 Online auctions with limited supply

All the problems that we discuss below, can be described in the following common framework, that is an extention of the problem we discussed in the introduction. Suppose we have a set U of bidders and let $\mathcal{I} \subseteq 2^U$ be a collection of sets of the simultaneously winning bidders (the ones that buy an item), namely, if $S \in \mathcal{I}$ then S is a feasible set of accepted bidders. Our goal is to design online algorithms in which the structure of U is known (see for example section 2.3 matroids), as opposed to the elements and their values that belong to U , which are revealed one at a time. As each bidder arrives, our algorithm must decide whether to accept or reject him, satisfying the following constraints: decision of acceptance or rejection can't be changed, decision must be made before the arrival of the next element and finally the set of decisions must belong to \mathcal{I} . Last but not least, in any case,

we consider the assumption we mentioned in the previous section about the order of the bidders. Hence, the expectation is calculated over all permutations of the bidders and according to the random choices that are made from the algorithm.

2.2.1 Single item online auction

In this section, we consider an online auction for one single item, in which each bidder i has a type $\theta_i = (w_i, a_i, d_i)$ which is his private data. We consider that bidder i learns about the auction, or its value (w_i) for the good, at his arrival time a_i and needs a decision by departure time d_i . As we have already mentioned, we consider the scenario that the bidders come in random order. For this case, we mean that for a specific set of valuations $V = \{w_1, \dots, w_n\}$ and a specific set of intervals $T = [a_1, d_1], \dots, [a_n, d_n]$, we take the expectation over all the matchings from V to T ($n!$ matchings). As the general framework indicates for this special case, $U = \{1, 2, \dots, n\}$ the set of bidders and $\mathcal{I} = \{\{\}, \{1\}, \{2\}, \dots, \{n\}\}$ the collection of the sets with at most one bidder.

To deal with this problem we consider the following algorithm, ascribed to Dynkin, that is based on two simple ideas (it can be seen in Nisan et al. [21], Hajiaghayi et al. [15]). The first idea is the one of examining the first $\lfloor N/e \rfloor$ bidders and then decide to whom to sell the product, idea that we came up with from the analysis of Secretary problem. This is the best case for achieving a solution close to the optimal with respect to revenue and efficiency. The second idea is the one of Vickrey's second price auction, where we sell the product to the best bidder seen so far at the price of the second maximum bid. The latter idea is the reason for the algorithm-mechanism below to be truthful. Combined together the two ideas, we manage to have good competitive ratio and strong truthfulness.

<i>ALG1</i>
1. (transition): In period τ in which the $\lfloor N/e \rfloor$ -th bid is received, let $p \geq q$ be the top two bid values received so far. If an agent bidding p is still present in time τ then sell it to that agent at price q (breaking ties at random).
2. (Second phase): Otherwise, sell the item to the next agent to bid a value at least p at the price of p (breaking ties at random).

We prove now that ALG1 is truthful and find the competitive ratio as far as the revenue we defined in basic definitions and efficiency.

Theorem: ALG1 is strongly truthful for single-unit and limited supply environment, supposing that we have no early-arrival misreports.

PROOF: Fix θ_{-i} and assume no breaking ties (breaking ties are independent of the types). We consider the following cases according to θ_i :

- Case 1: If d_i is at the left of the transition, agent i is not allocated and thus he will not win the auction even if he deviates his type θ_i .
- Case 2: If $[a_i, d_i]$ spans the transition, agent i wins the bid if $w_i > q$. If he bids later departure or higher w_i he still wins. If $w_i \leq q$, he loses because the price of the item will be $\geq p$ (or $= q$ if there is in time τ an agent j that has $w_j = p$). For later arrival (misreporting), agent i still can't win the auction with $w_i < p$ and with $w_i \geq p$ there is a possibility of losing the auction (if reported arrival $a'_i > \tau$ and at the transition the bidder with the second highest bid is still present). Hence, in case i misreports, he can't win unless he would win being truthful.
- Case 3: If a_i is after the transition, agent i wins the bid if $w_i \geq p$ and there is no bidder j with $w_j \geq p$ that appears in time interval $[\tau, a_i]$. For later arrival, agent i still can't win the auction with $w_i < p$ and with $w_i \geq p$ there is a possibility of losing the auctions (another bidder j with w_j with $a_j > \tau$ and $a_j < a'_i$ would be the winner). For later departure, i can't win the auction with less bid value than p and still wins if he was the winner before. Earlier arrival misreporting is not allowed (in that case agent i would manipulate the auction).
- Case 4: If agent i triggers the transition, then he wins if and only if $w_i > q$, namely $w_i = p$. The case for arrival and departure misreports is similar to case 2.

The theorem below shows that ALG1 is e -competitive with respect to revenue and efficiency. A similar proof can be found at Nisan et al. [21], Hajiaghayi et al. [15].

Theorem: The ALG1 is $e+o(1)$ -competitive for efficiency and $e^2+o(1)$ -competitive for single-unit and limited supply environment as $N \rightarrow \infty$.

PROOF: For efficiency which is denoted w_i where i is the winning bidder, we consider the definition of section 2.1.1 and assuming that C is the fact of choosing the agent with the highest value (let w^*), we have to prove that $\Pr[C] \geq \frac{1}{e} - o(1)$, because $E\left[\frac{w_{winner}}{w^*}\right] = \frac{E[w_{winner}]}{w^*} \geq \frac{\Pr[C]w^*}{w^*} \geq \Pr[C]$. We examine two cases:

- Case A: Selling at step 1 (transition) of ALG1.
The probability of choosing the agent with the highest value is at least $\frac{\lfloor N/e \rfloor}{N} =$

$\frac{1}{e} - o(1)$, that is the probability the agent with highest value to belong to the interval $[1, \dots, \lfloor \frac{N}{e} \rfloor]$.

- Case B: Selling at step 2 (after transition) of ALG1.
The probability of choosing the highest value event is at least $\frac{1}{e} - o(1)$. (See secretary problem analysis made above).

So the total probability $\Pr[C] = \Pr[A] \cdot \Pr[C|A] + \Pr[B] \cdot \Pr[C|B] \geq (\frac{1}{e} - o(1))(\Pr[A] + \Pr[B]) = \frac{1}{e} - o(1)$ and the competitive ratio is at most $\frac{1}{\Pr[C]} \leq \frac{e}{1 - o(1)} = e + o(1)$.

For revenue which is denoted by p_i where i is the winning bidder (p_i is the amount i pays) we consider the definition of section 2.1.1 and assuming that C is the fact of selling the item to the agent with the highest value at a price equal to the second-highest bid (let w^{**} be the second highest bid), we have to prove that $\Pr[C] \geq \frac{1}{e^2} - o(1)$, because $E \left[\frac{p_{winner}}{w^{**}} \right] = \frac{E[p_{winner}]}{w^{**}} \geq \frac{\Pr[C]w^{**}}{w^{**}} \geq \Pr[C]$. We examine two cases too:

- Case A: Selling the item at step 1 (transition) of ALG1. The probability of selling the item to the agent with the highest value at a price equal to the second-highest bid can be occurred as follows. The probability of the first event is $\frac{1}{e} - o(1)$ and of the second event is also $\frac{1}{e} - o(1)$. So the total probability $\Pr[C|A]$ is $(\frac{1}{e} - o(1))^2 = \frac{1}{e^2} - o(1)$.
- Case B: Selling the item at step 2 (after transition) of ALG1. The probability of selling the item to the agent with the highest value at a price equal to the second-highest bid can be occurred as follows. The probability of the first event is $\frac{1}{e} - o(1)$ and of the second event is $1 - (\frac{1}{e} - o(1))$. Hence, the total probability $\Pr[C|B]$ is $(\frac{1}{e} - o(1))(1 - \frac{1}{e} + o(1)) = (\frac{1}{e})(1 - \frac{1}{e}) - o(1) > \frac{1}{e^2} - o(1)$.

So the total probability $\Pr[C] = \Pr[A] \cdot \Pr[C|A] + \Pr[B] \cdot \Pr[C|B] \geq (\frac{1}{e^2} - o(1))(\Pr[A] + \Pr[B]) = \frac{1}{e^2} - o(1)$ and the competitive ratio is at most $\frac{1}{\Pr[C]} \leq \frac{e^2}{1 - o(1)} = e^2 + o(1)$.

The case of the single item is rather an easy application of the algorithm of the secretary problem. However, in online auctions we may face problems with more than one item that has to be sold (that is we have more than one winning bidder). The section below, deals with the more generalised version.

2.2.2 Multi-item auction - k -choice secretary problem ($k > 1$)

In this section, we consider the model of the previous section (2.2.1). However, in this case the auction has k ($n \geq k > 1$) identical items instead of one. If we consider that U is the set of bidders and I is the family of all the subsets of U

with at most k elements, we can integrate this case to the general framework. A first approach is to generalize ALG1. In ALG2 described below, we observe the first $t = \lfloor n/e \rfloor$ bidders (suppose at time τ comes the t -th bidder) and we create a set T with the k highest (if we have $k > t$ then we fill T with zeros). Set T is used for comparison and the elements that belong to T are not all of them actually selected. Whenever a bidder j , with $a_j > a_\tau$ arrives, if his bid value w_j is larger than the minimum of T and the minimum of T belongs to a non-selected bidder, an item is sold to j and T is updated (w_j replaces the minimum). This continues till we run out of bidders or items.

ALG2

1. Observe the first $t = \lfloor n/e \rfloor$ without picking any of them.
2. Let T be the set of the best k elements (if $k > \lfloor n/e \rfloor$ fill with dummy zero variables)
3. Whenever an element arrives whose value is greater than the minimum-value element in T (which must not be selected), select this element, put it in T and delete the minimum-value element from T .

Babaioff et al [5] proved that ALG2 is e -competitive with respect to $\sum_{j=1}^k w_{j^*}$ where w_{j^*} is the j -th highest bidder as far as the bid values are concerned.

Theorem: The ALG2 is e -competitive with respect to revenue.

PROOF: Let S be the set of the selected bidders, $w(S) = \sum_{j=1}^k w_j$, i^* be the i -th larger bid value ($i \leq k$) and $S^* = \{1^*, \dots, k^*\}$. We prove that $\Pr[i^* \in S] \geq \frac{t}{n} \ln(n/t)$. Suppose the i -th bidder is observed at time j ($j > t$, $i \geq k$). Then the probability of selecting j is equal to the probability of the minimum-value element in T not to be selected, namely to be arrived at time t or earlier. Thus the probability is equal to $\frac{1}{n} \cdot \frac{t}{i-1}$. Hence the total probability $\Pr[i^* \in S]$ by union bound is equal to $\sum_{j=t+1}^n \frac{1}{n} \cdot \frac{t}{i-1} \geq \frac{t}{n} \int_t^n \frac{1}{x} dx = \frac{t}{n} \ln(\frac{n}{t})$. Thus $E[w(S)] = \sum_{j=1}^n w_j \Pr[j \in S] \geq \sum_{j=1}^k w_{j^*} \Pr[j^* \in S] > \frac{t}{n} \ln(n/t) w(S^*)$ (we assumed $t = \lfloor n/e \rfloor$). Substituting t for $\lfloor n/e \rfloor$ it follows that $E[w(S)] \geq ew(S^*)$.

However, Kleinberg in [18] proved that e -competitive ratio is far from optimal as $k \rightarrow \infty$. He defined a recursive algorithm that is $1 + O(\frac{1}{\sqrt{k}})$ -competitive. It is remarkable that competitive ratio reaches 1 as $k \rightarrow \infty$. The algorithm works as follows. It separates the bidders (in order of arriving) into two almost equal in size segments (with the use of binomial distribution). It chooses recursively l (up to $k/2$) bidders from the first segment and $k-l$ from the second segment that exceed a threshold bid value. The threshold bid value is the bid of the l -th bidder that was chosen from the first segment. For the special case of $k = 1$, it uses ALG1. Formally the algorithm is the following:

$ALG3(1, n, k)$ 1. if $k = 1$ then return $ALG1(1, n)$ else { 2. Choose m according to binomial distribution $B(n, 1/2)$ 3. Let $S = ALG3(1, m, k/2)$ and $l = S $ 4. Let $p_t =$ minimum element of S 5. $O = S$. 5. for $i = m + 1$ to n do 6. if $ O = k$ then break else 7. if $w_i > p_t$ then $O = O \cup \{i\}$ 8. return O } }

Theorem: The ALG3 is $(1 + O(\frac{1}{\sqrt{k}}))$ -competitive with respect to revenue.

Proof-sketch: Kleinberg in [18] uses induction on k to prove that for any set S with n , the expected value of the elements ALG3 selects is at least $(1 - 5/\sqrt{k})v$, where v is the sum of the k largest elements in S . First of all, he defines as T the set of the k largest elements and modified value of $x \in S$ equal to x if $x \in T$, zero otherwise. Then he shows that $Y = \{w_1, \dots, w_m\}$ is a sample of the uniform distribution over all 2^m subsets of S . This is straightforward as the probability $|Y| = m$ is $(1/2)^n \frac{n!}{m!(n-m)!}$ and thus the probability $Y = \{w_1, \dots, w_m\}$ is $(1/2)^n \frac{n!}{m!(n-m)!} \times \frac{m!(n-m)!}{n!} = (1/2)^n$. Using that argument, it can be easily seen that $Y \cap T$ is a uniform random subset of T and thus $|Y \cap T|$ follows $B(k, 1/2)$ (binomial distribution). Hence, using the inductive hypothesis for $k/2$, Kleinberg proves that the elements selected from Y have expected modified value at least $(1 - 5/\sqrt{k/2})(1 - 1/(2\sqrt{k}))v/2$. Additionally, defining $Z = \{w_{m+1}, \dots, w_n\}$ and q as a random variable of the number of the elements that belong to Z and are greater than w_l (l is defined in ALG3), he proves that the expected modified value of the elements of Z selected by the ALG3 is at least $(1/2 - \sqrt{1/k})v$. Finally, by adding the expected modified values of the two sets (Y, Z) we have that $(1 - 5/\sqrt{k/2})(1 - 1/(2\sqrt{k}))v/2 + (1/2 - \sqrt{1/k})v > (1 - 5/\sqrt{k})v$ which completes the induction step.

2.2.3 Knapsack Secretary Problem

In this section, we are dealing with the online knapsack problem. Online knapsack problem is a modified version of knapsack problem and the online auction framework we discussed in section 2.2. Knapsack is NP -complete problem, ad-

mits FPTAS using dynamic programming as well as a simple 2-approximation. However, the online version is inapproximable to within any non-trivial factor. Below we describe the model formally and mention an algorithm with constant $(10e)$ competitive ratio, as it can be seen in Babaioff et al [5].

Model:

Let $U = 1, \dots, n$ set of n secretaries each have non-negative weight $w(i)$ and value $v(i)$. Given a weight bound W , we must select in an "online fashion" (assuming random-ordering hypothesis) a set $S \subset U$ such that:

$$\text{Maximize } \sum_{i \in S} v(i) \quad \text{subject to } \sum_{i \in S} w(i) \leq W$$

This case can be ascribed to the general framework if we consider U as the set of bidders and I as the family of sets whose total weight doesn't exceed W . For the unweighted case, every secretary i has weight $w(i) = 1$. This problem has already been analysed in the previous section (k -choice secretary problem).

For the weighted case, Babaioff et al. [5] propose the following algorithm, namely ALG4. In order to proceed to ALG4, we need to define *density*, $\rho(i) = \frac{v(i)}{w(i)}$ first.

ALG4:

Assume in this section that $W = 1$. (To reduce from the general case to the $W = 1$ case, simply rescale the weight of each element by a factor of $1/W$). The algorithm begins by sampling a random number $a \in \{0, 1, 2, 3, 4\}$ from the uniform distribution. The case $a = 4$ is a special case which will be treated in the following paragraph.

- If $0 \leq a \leq 3$, then the algorithm sets $k = 3^a$ and runs the k -secretary algorithm from Section 2.2 (ALG2, with $t = \lfloor n/e \rfloor$) to select at most k elements. If the k -choice secretary algorithm selects an element i whose weight $w(i)$ is greater than $1/k$, we override this decision and do not select the element.
- If $a = 4$, the algorithm operates as follows. It samples a random $t \in \{1, 2, \dots, n\}$ from the binomial distribution $B(n, 1/2)$. Let $X = \{1, 2, \dots, t\}$ and $Y = \{t + 1, t + 2, \dots, n\}$. For every element $i \in X$, the algorithm observes $v(i)$ and $w(i)$ but does not select i . It then sets $\rho' = \rho_X^{1/2}$ and selects every element $i \in Y$ which satisfies $w(i) \leq 3^{-4}$, $\rho(i) \geq \rho'$, and $w(S_{<i} \cup \{i\}) \leq 1$, where $S_{<i}$ denotes the set of elements which were selected by the algorithm before observing i .

The algorithm ALG4 is a combination of ALG2 and another algorithm (let KLP), the idea of which is the following. Instead of a threshold price, in this case we have threshold density. Initially, the algorithm separates the bidders in two segments (with expected equal size because we use binomial distribution to make the separation). It observes the first segment of elements and define a threshold density. Then it chooses an element of the second segment, if it's density is larger than the threshold. This idea comes from the fact that discrete knapsack problem can be reduced to an ILP problem. Thus ALG4 makes use of ALG2 (k -choice) with probability 0.8 (0.2 probability for each $k \in \{1, 3, 9, 27\}$) and the KLP with probability 0.2. Babaioff et al [5] proved the following theorem for the competitive ratio of ALG4:

Theorem: ALG4 is $(10e)$ -competitive.

Proof-sketch: Assume that $OPT = \{i_1, \dots, i_m\}$ in decreasing order of weight and S the set of elements selected by ALG4. The idea of the proof is to define $B_j = \{i_l | 3^j \leq l < 3^{j+1}\}$ for $0 \leq j \leq 3$ and $B_4 = \{i_{81}, \dots, i_m\}$, thus $OPT = B_0 \cup \dots \cup B_4$. Also, let $g_j = E[v(S) | a = j]$ ($v(S) = \sum_{i=1}^{|S|} v(i)$). By proving $b_j = v(B_j) \geq 2eg_j$ (for $0 \leq j \leq 3$ is easy, more difficult for $j = 4$) we have that $v(OPT) = b_0 + b_1 + b_2 + b_3 + b_4 \leq 2e(q_0 + q_1 + q_2 + q_3 + q_4)$, thus $v(OPT) \leq 10e(\sum_{j=0}^4 \Pr[a = j] \cdot E[v(S) | a = j]) = 10eE[v(S)]$.

2.3 Matroids

In this part, we deal with the same framework as predefined in the introduction of section 2.2. In this case, we consider that the allowed sets which belong to \mathcal{I} , form a combinatorial structure called a *matroid*. The restrictions that a matroid structure has to satisfy, help us to design algorithms solving problems that belong to the common framework and are $\log k$ -competitive, where k is the rank of the matroid, or even c -competitive with c constant. This work can be seen in Babaioff et al. [3]. Hence, the matroid structure is very important and this can be justified from the example we give in section 2.3.2. Below, we proceed with the formal definitions.

2.3.1 Definitions

Definition. A *matroid* (U, I) is constructed from a ground set $U \neq \emptyset$ and a nonempty family of subsets of U denoted by I , called the independent subsets of

U , such that if $B \in I$ and $A \subseteq B$ then $A \in I$. Additionally, if $A, B \in I$ and $|A| < |B|$, then $\exists x \in B \setminus A$ such that $A \cup \{x\} \in I$ (exchange property).

A matroid is a common extension of the ideas of a base in linear algebra (U is the set of vectors, and an independent set is a set with linearly independent vectors), of a graph (U is the set of edges, and an independent set is a set of edges that doesn't contain a circle) and other mathematical notions. Exchange property is the one that gives us the ability to form competitive algorithms since an early mistake can only lead to one suboptimal element being chosen and not for instance to having minimal/overall weight because the independent sets with our unlucky choice are comprised of unworthy items. It is also the main reason that the matroid (as it is a generalization of base) has maximal independent subsets all of which have the same number of elements called the *rank* of the matroid. This is true, because if we suppose for the contrary that there are two maximal independent set A, B with $|A| < |B|$, then from exchange property, there is x such that $A \cup \{x\}$ is also independent, which is a contradiction as A is maximal.

2.3.2 Algorithmic Model

The algorithmic model that we adapt considers an algorithm that is given the matroid structure at the beginning and receives the elements online. It maintains a set of selected elements and chooses to add or not the next one in line according to its weight and the possible independent sets.

Our strategic model considers matroid preference domains. Generally we have a set of U of n agents and a set Ω of possible outcomes (in auctions a possible outcome is a matching of successful buyers and products). In such domains each agent has a satisfying set of outcomes $A_i \subseteq \Omega$ in which he always obtains value V_i while 0 in outcomes in $\Omega \setminus A_i$. A set of agents is called independent if there is an outcome that satisfies exactly those agents. If for any profile of types the family of independent sets is a matroid the domain is called a matroid domain. Thus, we maintain the common framework we have already discussed with the constraint that the family feasible-allowed sets consists a matroid.

2.3.3 Importance of matroid structure

Matroid domains represent interesting economic problems. For instance, the *unit-demand domain* is built as a *transversal-matroid* domain. Given a set of elements, U , and a class I of subsets of U , a transversal of I is a subset S of U such that there is a one-to-one function f from S to I by which x belongs to $f(x)$ for each $x \in S$. (I may have repeated members, which are treated as separate subsets of U .) The set of transversals forms the class of independent sets of a matroid,

called the transversal matroid of (U, I) . Additionally, in the unit-demand preference domain there are n agents and a set M of m non-identical items. Each agent i desires a set of items T_i and obtains value v_i if he is successful. Each outcome is represented as a one-to-one matching of agents to items.

We give a proof-sketch which shows that the unit-demand domain has the structure of a transversal-matroid.

Theorem: The unit-demand domain has the structure of a transversal-matroid.

Proof-sketch: We consider a random profile of types. Let A be an independent set of agents and Ω its winning outcome. $\forall A' \subseteq A$ there is the outcome where some of the items are not obtained leaving the members of A' the sole agents with a matching to an element. Thus A' is independent. Let $|A| < |B|$ independent set. An agent b_i must have an item X that no agent in A has. If $b_i \notin A$ and is a_j we consider $A \cup \{b_i\}$ and we are done. If $b_i \in A$, we consider the outcomes where he obtains element X and then the sets $A_1 = A - \{a_j\}$ and $B_1 = B - \{b_i\}$. Inductively (consider A becoming empty ...) $\exists b \in B_1$ such that $A_1 \cup \{b\}$ is independent. Then since no agent in B_1 or A_1 obtains element X , the set $A_1 \cup \{b\} \cup \{a_j\}$ is independent. Thus $A \cup \{b\}$ is independent. Since by these the exchange property also holds the unit-demand domain is a matroid domain.

It is remarkable to mention that single-item (2.2.1 section) and multi-item auction (2.2.2 section) is a subcase of a unit-demand domain. For the first case, $\forall i \in U$ we have that $T_i = \{\text{item}\}$ and for the second case, assuming the items are $\{g_1, g_2, \dots, g_k\}$ then $\forall i \in U$ we have that $T_i = \{g_1, g_2, \dots, g_k\}$.

As the previous example demonstrated, however general, the assumption of a matroid domain demands some not-so-trivial properties in the structure of the economic problem we study. The truth is we cannot generalize further into set-systems with no restrictions whatsoever. The following problem shows that if we did that, it would be impossible to achieve constant-competitive algorithms:

Example problem without matroid structure

We consider an integer n and $k = \lfloor \log n \rfloor$ and manufacture the following set system (U, I) : U has n elements partitioned into $m = \lceil (n/k) \rceil$ subsets S_1, \dots, S_m . A set $A \subseteq U$ is independent if and only if it is contained in some S_i . The weight function assigns weight 1 with probability $1/k$ and 0 otherwise. It is independent of the structure of U .

Here we will demonstrate how the lack of the *exchange property* along with the fact that the algorithm can have no hint on which independent set to prefer since

every element appears online lead to a very low expected weight (less than 2) for the answer of any randomized online algorithm. On the other hand the density of elements with weight 1 cannot be the same in every set S_i (there are infinite...) and the probability that all of them have less than some constant weight tends to zero.

1. Suppose at some time t the algorithm makes its first choice and picks an element with weight 1 from set S_i . Afterwards it can choose up to k other elements that belong to S_i and since each one has probability $1/k$ to have weight 1 (0 otherwise) and the weight of each element is independent of the others the expected value of the answer will be less than 2. ($1 + 1/k + 1/k + \dots + 1/k$)
2. On the other hand if we consider E_i to be the event that set S_i has weight more than j (where j is some random number) then $\Pr[E_i]$ is much more than $1/k^j$. Thus the probability that none of the events E_i occurs (there is no "favoured set") is less than $[1 - 1/k^j]^m$ (the events are independent). However $[1 - 1/k^j]^m = [1 - 1/k^j]^{n/k} = [1 - 1/k^j]^{(k^j) \cdot (n/k^{j+1})}$ which is close to $(1/e)^{n/k^{j+1}}$ and since $k = \log n$, by de l'Hospital (n/k^{j+1}) has the same limit as $n/(j!)$ that is, infinite. Thus $(1/e)^{n/k^{j+1}}$ tends to 0 as well as probability that none of the events E_i occurs.

Since j could be any constant value there cannot exist any constant-competitive algorithm, for then a big enough n would provide (with almost certain probability) a set E_i with weight more than j . If we furthermore consider j to be dependent on n (and equal to $k/(2 \ln k)$) it would be easy to show with similar calculations that the expected value of the maximum-weight independent set is $\Omega(\log n / \log \log n)$ (see *balls and bins* in Rajeev Motwani and Prabhakar Raghavan - Randomized Algorithms).

2.3.4 Logarithmically competitive algorithm for general matroids

One of the properties that matroids inherit from linear algebra's bases is the dimension. It can be shown (through use of the exchange property) that every largest independent set in a matroid has the same number of elements, namely the rank of the matroid. The previous result showed we cannot achieve constant-competitive algorithms in the general case; here we present an algorithm as can be seen in Babai et al. [3] that is $O(\log k)$ -competitive for general matroids, k being the rank of the matroid.

Threshold Price Algorithm

1. Observe half of the elements without picking any, and call the observed elements the sample set S .
2. Consider the element t^* in the sample set that has maximum weight. Pick a random number j between 0 and $\lceil \log k \rceil$ (k is the rank of the matroid). The threshold price will be the weight of t^* divided by 2^j .
3. Let l_i be the element in $U \setminus S$ observed at time $i = s + 1, \dots, n$ and B the set of selected elements (which is an empty set at the beginning). If the weight of l_i is bigger than the threshold value ($w(l_i) \geq w(t^*)/2^j$) and $l_i \cup B$ is an independent set then select l_i . (i.e. add it to B).

Theorem: The threshold price algorithm is $32 \lceil \log k \rceil$ -competitive for any matroid domain where k is the rank of the matroid.

Before the proof we provide some comments on the structure of the algorithm. First of all it is clear that the algorithm will be $O(\log k)$ competitive only in respect to the expected weight of the answer; as we said at the beginning, great economic interest exists for matroids where the weight function is random and evenly distributed. We also like to comment on the choice of the threshold value: The divisor is less than the rank (the maximum size of any independent set) and thus provides reasonable boundaries when the algorithm decides on expanding an independent set. It also can have several values which we will be able (with a certain sacrifice to the competitiveness) to use for the best answer in different cases. Actually it is exactly this part that adds the $\log k$ ration in the competitiveness of the algorithm.

PROOF: Let B^* be the maximum weight independent set of the matroid with elements x_1, \dots, x_k with values in descending order $v_1 \geq v_2 \geq \dots \geq v_k$. We only consider those that are bigger than v_1/k let them be v_1, \dots, v_q and mention that the rest can only add up to less than v_1 (since there are k elements in the base). Thus $v_1 + \dots + v_q$ contributes more than half the weight of B^* . Next, for any set $A \subseteq U$, let $n_i(A)$ denote the number of elements in A with weight at least v_i and $m_i(A)$ be the number of elements in A with weight at least $v_i/2$. A known summation technique gives us

$$v_1 + v_2 + \dots + v_q = \sum_1^{q-1} (v_i - v_{i+1}) \cdot n_i(B^*) + v_q \cdot n_q(B^*)$$

Now if B is the answer of the algorithm, the value of B is bigger than the sum of the values of the elements in B that have weight at least $v_q/2$ which is

$$\sum_1^{q-1} ((v_i)/2 - v_{i+1}/2) \cdot m_i(B) + v_q/2 \cdot m_q(B) = 1/2 \cdot \sum_1^{q-1} (v_i - v_{i+1}) \cdot m_i(B) + 1/2 \cdot v_q \cdot m_q(B)$$

Using the above representations for the value of the max-weight base and the answer set it is easy to see that to provide a factor for the competitiveness of the algorithm we must place the expected value of $m_i(B)$ within a factor of $n_i(B^*)$. We have lost a factor of 2 by considering only v_1, \dots, v_q and another factor of 2 to the coefficient of the last sum. We need to prove that the expected value of $m_i(B)$ is within a $8(\log k)$ factor of $n_i(B^*)$. We will do this by showing that $m_i(B) \geq n_i(B^*)/4$ with probability $\frac{1}{2 \log k}$. Thus even if in every other case $m_i(B)$ is very small the expected value of $m_i(B)$ is bigger than $(1/8 \log k) \cdot n_i(B)$.

1. For $i = 1$ we have a special consideration. With probability $1/4$ the sample doesn't contain the maximum-weight element but does contain the element with the second-highest weight. Conditional on this event with probability $1/\log k$ the threshold price will be $x_2/2^0 = x_2$ and the answer of the algorithm is the set $\{x_1\}$. Thus with probability $\frac{1}{4 \log k}$ the value of $m_1(B)$ is 1, and so the expected value of $m_1(B)$ is bigger than $\frac{1}{4 \log k}$ while $n_1(B^*) = 1$.
2. Now assume that $i > 1$. By definition $n_i(B^*) = i$. We condition on the event that the sample contains x_1 (the maximum weight element) and we wish the value of the threshold to be small enough so that the i biggest elements can pass the test, but not smaller. That is we also condition on the event that j is such that $w(t^*)/2^j$ is the maximum number in the set of possible threshold values that is less than or equal to v_i . Since we have assumed that $v_i \geq v_q \geq v_1/k$ which is the smallest possible threshold value (since $t^* = v_1$) such a j exists and the algorithm selects this particular j with probability $1/\log k$. Thus with probability $\frac{1}{2 \log k}$ every element of the independent set $A = \{x_1, \dots, x_i\}$ exceeds the threshold value. The expected amount of elements in A that appear outside the sample is $(i-1)/2 > i/4$. Now this means that the expected size of the answer set B will also be more than $i/4$ elements: Well the algorithm can decline elements of A only if they don't unite with B in an independent set (because all of them do exceed the threshold value) but we also know from the exchange principle that if $|A^*| > |B|$ then there is an element in A^* that can be moved to B , and therefore it would have been chosen by the algorithm. So of the at least $i/4$ elements of A that are expected to appear in the second half the algorithm has to choose enough for B to have size $i/4$ (it can choose none if B has size $\geq i/4$ but the opposite would mean contradiction).

Finally since the threshold value is at least $v_i/2$ (conditionally on the previous events), and every element of B must exceed the threshold value $m_i(B) = |B| \geq i/4 = n_i(B^*)/4$. And that happens with probability $\frac{1}{2 \log k}$. This completes the argument.

However, for specific matroid domains as Unit-Demand Domain (which has the

structure of a transversal-matroid) we can achieve better competitive ratio. The example below justifies our claim.

Example of transversal matroid with bounded-left degree

Assume that there is a constant d such that $|T_i| \leq d$, for all i , that is, each agent desires one of at most d items. An outcome is mapping of agents to items, such that each agent is matched to at most one item. Let A_i be all the outcomes in which i is matched to an item in T_i . We also assume that the values v_i and set T_i are private information. The matroid elements of a transversal matroid correspond to vertices on the left side (L) of a bipartite graph $G = (L, R, E)$ (thus the size of the ground set is $|L| = n$). A set $S \subset L$ is independent if there is a perfect matching of S to nodes in R . The unit-demand domain is a transversal matroid domain in which L is the set of agents, R is the set of items, and there is an edge from $l \in L$ to $r \in R$ if $r \in T_l$. The bound on the number of items an agent desires translates to a bound of d on the maximal degree of any node in L . The value of an agent l corresponds to the weight of the node $l \in L$ and is denoted by $w(l)$. Below, we present a $4d$ -approximation algorithm to the matroid secretary problem for transversal matroids with left-degree at most d .

Price Sampling Algorithm (PSA)

- Observe $s = \lfloor n/2 \rfloor$ elements without picking any element, and let $S \subset L$ be the set of observed elements (S is called the sample). For a right node $r \in R$, let $l_s^*(r) \in S$ be the sampled left node with maximal weight that is a neighbor of r . Let the price of $r \in R$ be $w(l_s^*(r))$.
- At time $t = s + 1, \dots, n$ we observe element $l \in L$ with weight $w(l)$. Let $R^*(l)$ be the set of unmatched neighbors of l with price lower than $w(l)$. If $R^*(l)$ is not empty, match l to the node with the lowest price in $R^*(l)$.

The theorem below, as it can be seen in Babai et al. [3] shows that Price Sampling Algorithm is truthful and $4d$ -approximation with respect to the weight of the maximum weight matching.

Theorem: For any transversal matroid with bounded left degree d , the above algorithm is a $4d$ -approximation and truthful.

Proof-sketch:

For approximation: Let OPT be a maximum weighting matching in graph, $w(S) = \sum_{l \in S} w(l)$ for a set S and let $w(m(r))$ be the weight of the element that is matched to $r \in R \cap OPT$ (0 if r is unmatched). Assume H be the set of neighbors with

maximal weight for every $r \in R$. Firstly, in [7] it is proven that $w(OPT) = \sum_{r \in R} w(m(r)) \leq \sum_{r \in R} w(h(r)) \leq d \cdot \sum_{l \in H} w(l) = d \cdot w(H)$ (as each vertex $r \in R$ has $d_r \leq d$). Moreover, it is proven that with probability at least $1/4$, each $l \in H$ is matched by PSA. Thus $E[w(PSA)] \geq \sum_{l \in H} 1/4 w(l) = 1/4 w(H)$. Hence $4dE[w(PSA)] \geq w(OPT)$.

For truthfulness: We consider two cases:

- Agent i in the sample: Obviously agent's utility is 0 for any declaration. Thus he doesn't gain anything from lying.
- Agent i not in the sample: For every item that is not taken and belongs to T_i , i takes it and pays its given price. Truthtelling maximizes his utility (about T_i and v_i).

Chapter 3

Digital Goods

3.1 Introduction

Nowadays, a lot of auctions take place in internet. The items that are sold at these kind of auctions, in most of the cases are digital goods - products (songs, videos, pages). It is very important to deal with auctions, where the auctioneer can sell as many units of his products as he is asked to do. Thus, in this section, we are examining online auctions that are in an unlimited supply setting. Specifically, the auctioneer is able to sell any number of units of each item (we may have a single item) and they each have zero marginal cost to seller. We also consider *single-minded* bidder, which means that each bidder is interested in only a single bundle of items. In the first sections (3.3,3.4), we consider the offline case for the unlimited supply auctions. After that we make an introduction to mechanisms for the online case.

3.2 Unlimited Supply Auctions

3.2.1 Basic Definitions

Model:

We consider auctions that have goods which are available in unlimited supply. Particularly, we examine the case of one item which the auctioneer can sell to each one of the n bidders (at least n copies of the item). Each bidder i has a value $v_i \in [1, h]$ (w.l.o.g, if we rescale by dividing with the smallest bid value) that shows how willing he is to buy the item. If a bidder i wins the auction (the auctioneer

sells him a good), then his non-negative utility function is denoted $u_i = v_i - p_i$ (else $u_i = 0$), where p_i is how much money i spent for the good. In order to give a game theoretic notion, bidders are strategic and they might try to manipulate (in order to increase their utilities) by misreporting their valuation. Furthermore, the online notion for this family of auctions as it can be seen at Bar-Yossef et al [7], can be given if we consider that each bidder comes at a time and the auctioneer decides a price p_i for the i -th bidder and bidder i buys that item if $b_i \geq p_i$ and all that must happen before the arrival of the next bidder. The goal is to find truthful mechanism that achieve a good competitive ratio with respect to auctioneer's revenue. Later, we will discuss about the choice of the benchmark to compare a truthful offline with the optimal untruthful offline and also an online with the optimal offline algorithm.

Finally it is remarkable to mention that there are two main categories of auctions as far as the item-pricing is concerned. We have the *single-price* auctions, where all the winning bidders pay the same price and the *multi-price* where the prices may differ. In section 3.4 we mention algorithms for both categories. Additionally, for each category, we define the optimal revenue that a mechanism might output. For the single-price auctions we define the optimal fixed price revenue as

$$F = \max_{1 \leq i \leq n} i \cdot v^{(i)}$$

and for the multi-price auctions the optimal revenue is denoted by

$$T = \sum_{i=1}^n v_i$$

where for $v^{(i)}$ we consider the i -th largest valuation of the bidders. Goldberg et al. [13] proved that there is an inequality relation between F, T (it is also obvious that $F \leq T$). They proved that $F \geq T/(2 \log h)$. The idea of the proof is to partition the bids in $\log h$ bins, where in the i -th bin we put all the bids j with the property that $2^{i-1} \leq v_j < 2^i$ (we start the enumeration from 1). Then from Pigeonhole Principle, there is a bin $S = \{b_1, \dots, b_k\}$ (assume the bids in increasing order) with $\sum_{j=1}^k b_j$ at least $T/\log h$. Assuming S is the winning bidders that pay the minimum element of S , namely b_1 , then $F = k \cdot b_1 \geq 1/2 \sum_{j=1}^k b_j \geq (1/2)T/\log h$. This is interesting, because it means that in a single-price auction we may deal with T revenue and in a multi-price auction may deal with F revenue. Finally Rao and Tardos proved that $F \geq T/(4 \log n)$ (inequality that depends on n). The writer proved a better inequality which is the following:

Theorem: $F \geq T/\ln(en)$.

PROOF: Let $F = k \cdot v^{(k)}$. Then obviously $\frac{k}{l} \cdot v^{(k)} \geq v^{(l)}$ for every $1 \leq l \leq n$. Thus $F \cdot \sum_{j=1}^n \frac{1}{j} = k \cdot v^{(k)} \sum_{j=1}^n \frac{1}{j} \geq \sum_{j=1}^n v^{(j)} = T$. Additionally $\sum_{j=1}^n \frac{1}{j} \leq \int_1^n \frac{1}{x} dx + 1 =$

$\ln n + 1 = \ln(en)$. Thus $F \geq T/\ln(en)$.

3.2.2 Competitive ratio benchmark

Benchmark for competitive ratio:

As opposed to the limited supply auctions where we compared the revenue of the online with the k-Vickrey's revenue, here we compare the revenue of the truthful mechanism with the following which is called optimal fixed price revenue (for the single-price case):

$$F^{(2)} = \max_{1 < i} i \cdot v^{(i)}$$

We didn't take as a benchmark the case for $i = 1$, namely F , because informally, suppose the bidder i with the highest valuation has $v_i = h \gg n$ and all the other bidders have $v_j = 1$ then there is no way to create a truthful mechanism that has expected profit (of the auctioneer) $c \cdot h$ where c is a constant less than 1. It is remarkable that for the multi-price case, we take as a benchmark T . As $T \geq F^{(2)} \geq T/2 \log h$ holds, actually most of the times both are used (assuming that $F > h$, thus $F^{(2)} = F$).

3.3 Truthfulness

3.3.1 Bid-independent

Definition 1: Let f be a predetermined function (before the start of the auction). Bid-independent auctions are the auctions that have the following property:

- Bidder i wins an item if and only if $f(b_{-i}) \leq v_i$ and he pays $p_i = f(b_{-i})$, that is the amount i pays is independent of his announced valuation.

It is rather straightforward to see that the bid-independent mechanism may be for single-price or multi-price auction too. Intuitively, in general case the bid-independent mechanisms must be for multi-price auctions. However, consider the following example. Let $f(b_1, \dots, b_{n-1}) = \min(b_1, \dots, b_{n-1})$. Then every bidder will buy the item at the price of the smallest bid value (the same price) except of the bidder with the smallest bid who will lose. Goldberg et al. [13], proved that a mechanism is truthful if and only if it is bid-independent. This is true for both the deterministic and randomized case. This is actually a corollary from Theorem 13.6 in Nisan et al. [21]. Thus, the target is to find a function $f : R^{n-1} \rightarrow R$ to achieve a good competitive ratio with respect to the untruthful optimal fixed price

revenue (truthfulness will be guaranteed). For example, suppose $f(b_1, \dots, b_{n-1}) = \max(b_1, \dots, b_{n-1})$. It is obvious that the auction above is the Vickrey's second price auction, which is a truthful mechanism as shown in section 1.3. From now on, every mechanism we discuss is bid-independent, thus truthful. So notions b_i (announced valuation) and v_i (true valuation) are considered the same.

3.3.2 The deterministic approach

As we have already said, we are interested in truthful mechanisms. The purpose of this section is to show that any symmetric¹, deterministic truthful mechanism can't have a constant competitive ratio with respect to $R/F^{(2)}$ for the single-price unlimited supply auctions (R is the sum of sale prices which returned from the algorithm). Remember that we don't take F as a benchmark (see example in 3.2.2). The following theorem holds:

Theorem: There is no symmetric, deterministic, truthful mechanism M that is constant-competitive against $F^{(2)}$.

PROOF: We consider the case where all bids are 1 or n (we have n bidders). As M is truthful and symmetric, then the auction is bid-independent. Thus let $f : \{1, n\}^{n-1} \rightarrow R$ be the symmetric function that determines the threshold price. W.l.o.g we assume that the image of f is $\{0, 1\}$. Because of symmetry, assume that $f(j)$ is the threshold price for bid vector with j 1's and $n - j - 1$ n 's. We will prove that $R \leq F/n$.

Clearly $f(n-1) = 1$, else if $f(n-1) = n$ then for a bid vector $\{1, 1, \dots, 1\}$ we would have that $R = 0$ and since $F = n$, the theorem would hold. Also $f(0) = n$, else if $f(0) = 1$ then for a bid vector $\{n, n, \dots, n\}$ we would have that $R = n$ and since $F = n^2$, the theorem would hold. Let k be the smallest integer in $\{1, 2, \dots, n-1\}$ such that $f(k) = 1$. Then for a bid vector with k bids 1's and $n - k$ bids n 's we have that, since $f(k-1) = n$, all the bidders with bid 1 will lose and the threshold price for the winners will be 1. Thus $R = n - k$ and $F = k + n(n - k)$, so $R/F \leq 1/n$.

So, we have to use randomization to achieve a better ratio for the expected R . The next section deals with it.

¹symmetric means that f has to be symmetric, namely independent from the sequence of the bids

3.4 Randomized sampling

3.4.1 Introduction

As we have already mentioned in the previous section, symmetric truthful mechanisms are not competitive with respect to revenue (as benchmark we considered $F^{(2)}$). Thus we have to use randomization to do our work. The algorithm we will describe in section 3.4.2, is based on randomized sampling. The general idea is the following. Let B be the set of bids and B' a subset of B and a symmetric function f . We define a threshold price which is equal to $p_t = f(B')$ and use it to the set $B \setminus B'$. The winners are those who belong to $B \setminus B'$ and have valuation $\geq p_t$. Clearly the family of randomized sampling algorithms are truthful (just observe that they are bid-independent). The difficult part is to choose an appropriate f and B' (of size) in order to achieve a good competitive ratio with respect to R .

3.4.2 Random Sampling Optimal Threshold Auction

Let $f(b_1, \dots, b_{n-1}) = \operatorname{argmax}_{b_i} b_i \cdot (n - i + 1)$ ((b_1, \dots, b_{n-1}) are in increasing order). Consider the following algorithm for the single-price case where B is the set of announced bids.

<i>Random Sampling Optimal Threshold Auction</i>
<ol style="list-style-type: none">1. Choose $B' \subset B$ randomly with $B' \approx \lfloor n/2 \rfloor$. (toss a fair coin for each element)2. Let $p_t = f(B')$ (fill with dummy zeros)3. for $b_i \in B \setminus B'$ do4. if $b_i \geq p_t$ then sell to i with price p_t.

This is an application of the general idea described above. We will prove now that the expected revenue $E[R] \geq F/12$ for if we assume that $F \geq 288 \ln(3)h$ (that is, the benchmark is actually $F^{(2)}$ as $F > h$). We will use the following lemma, the proof of which can be found in Goldberg et al [13].

Lemma: Assume $ah \leq F$. Then $R \geq F/6$ with probability at least $1 - e^{-a/36} - 40e^{-a/72}$.

Theorem: Assume $288 \ln(3)h \leq F$. Then $E[R] \geq F/12 = F^{(2)}/12$.

PROOF: Substitute a for $288 \ln(3)h$ from the lemma above. Then $R \geq F/6$ with probability at least $1 - \frac{1}{e^{8 \ln 3}} - \frac{40}{e^{4 \ln 3}} = 1 - \frac{1}{3^8} - \frac{40}{81} > 1/2$. Thus $E[R] \geq (1/2)(F/6) = F/12 = F^{(2)}/12$.

For the multi-price case, we consider the following modification of random sampling optimal threshold auction (called RSOP) for the single-price auctions. Actually, this case is a subcase of multi-price case, called dual-price auction, where every winning bidder pays one of two possible threshold prices (what he pays depends on the partitioning the algorithm makes).

<i>Random Sampling Optimal Threshold Auction (RSOP)</i>

- | |
|---|
| <ol style="list-style-type: none"> 1. Choose $B' \subset B$ randomly with $B' \approx \lfloor n/2 \rfloor$. (toss a fair coin for each element) 2. Let $p_t = f(B')$ and $p'_t = f(B \setminus B')$ (fill with dummy zeros) 3. for $b_i \in B \setminus B'$ do <li style="padding-left: 20px;">4. if $b_i \geq p_t$ then sell to i with price p_t. 5. for $b_i \in B'$ do <li style="padding-left: 20px;">6. if $b_i \geq p'_t$ then sell to i with price p'_t. |
|---|

For the performance analysis, we just use the theorem above, assuming that $F \geq 288 \ln(3)h$. Suppose R_1, R_2 be the revenue for the two partitions. Then $E[R_1] + E[R_2] \geq 2F/12 = F/6$. Because we have a multi-price auction we should take as a benchmark optimal revenue T . Using the fact that $F \geq T/2 \log h$, we have that $E[R] \geq T/12 \log h$. Additionally, it is important to mention that, Feige et al.[11] proved that RSOP is 15-competitive for the benchmark $F^{(2)}$, namely $E[R] \geq F^{(2)}/15$ without making any assumptions on $F^{(2)}$ (as we did in the theorem above). Finally, it is rather easy to see that we can't achieve better competitive ratio than 4 with RSOP. Consider the bid vector $\{1, 1, \dots, 1, h, h + \epsilon\}$ where $h \gg 1$ (actually we need only $h > n$). Then R is not zero if and only if bids $h, h + \epsilon$ fall on different sides of the partition (this fact happens with probability $1/2$). So, if $h, h + \epsilon$ fall on different sides, then only bidder with bid $h + \epsilon$ will win and he will pay h . Thus $E[R] = h/2$ and $F^{(2)} = 2h$. Hence $F^{(2)}/E[R] = 4$.

3.5 Online case

The previous sections were a friendly introduction to the model of digital goods. We saw some general ideas of how we examine the performance of a truthful mechanism, we rejected the deterministic approach of dealing with digital-goods auction and we saw the randomized sampling method that under some assumptions has constant competitive ratio with respect to the optimal fixed revenue. Here, we will examine digital goods in an online manner, the model of which has already been mentioned in the 3.2.1. We consider two cases, the full information case and the partial information case. The kind of auctions described in 3.5.1, 3.5.2 section are considered to be the full information case, where after the arrival of the bidder, the auctioneer learn about bidder's true valuation. In section 3.5.3, we will

consider the partial information case, where the only information the auctioneer has is the acceptance or rejection of the bidder to buy the item. All the algorithms we examine are bid-independent and thus truthful as far as the valuation of the bidders is concerned (not the time of arrival).

3.5.1 Online Weighted Buckets Auction

Bar-Yossef et al [7] introduced the *online bucket auction* that we will need next when we define *Exp3* algorithm. First of all, we give some basic definitions that will be used in the auction we will present next.

Terminology and Notation: For $l = \lfloor \log h \rfloor + 1$, valuation buckets are the intervals I_0, \dots, I_{l-1} where $I_k = [2^k, 2^{k+1})$. Additionally, we denote $B_k = \{i \in \{1, \dots, n\} | v_i \in I_k\}$, that is the set of bidders whose valuations fall in I_k and finally $w_k = \sum_{i \in B_k} v_i$, namely the weight of set B_k . Consider the following algorithm with parameter d .

<i>Online Weighted Bucket Auction (W_d)</i>	
1.	for $i = 1, \dots, n$ do (v_1, \dots, v_n is the order of arriving)
2.	if every bucket is empty choose one at random
	else choose I_k with probability $\frac{w_k(\{v_1, \dots, v_{i-1}\})^d}{\sum_{j=0}^{l-1} (w_j(\{v_1, \dots, v_{i-1}\}))^d}$.
3.	Let $s_i = 2^t$, where t is the chosen bucket
4.	if $v_i \geq s_i$ then sell to i with price s_i .
5.	Put v_i to the right bucket.

The algorithm works as follows. At each time a bidder comes (let i), we set price threshold $s_i = 2^k$ with probability $\frac{w_k(\{v_1, \dots, v_{i-1}\})^d}{\sum_{j=0}^{l-1} (w_j(\{v_1, \dots, v_{i-1}\}))^d}$. If $s_i \leq v_i$ then bidder i buys the product at the price of s_i and then we update the buckets by putting in the valuation v_i (where it belongs). It is remarkable that $[1, h]$ must be known in advance. For the performance analysis, we assume that $F^{(2)} \geq 9h$ and using the two following lemmas, we prove that RSOP is $O(3^d \log h^{1/(d+1)})$ -competitive w.r.t $F^{(2)}$.

Lemma 1: Assume $9h \leq F^{(2)}$. Then $(1/2) \max_k w_k \leq F^{(2)} \leq 3 \max_k w_k$.

PROOF: Let k be a bucket with maximum weight. Then obviously $F^{(2)} \geq 2^k |B_k|$. Additionally, $(1/2)v_i \leq 2^k \leq v_i$ for every $i \in B_k$, thus $F^{(2)} \geq 2^k |B_k| \geq (1/2) \sum_{i \in B_k} v_i = (1/2) \max_k w_k$. For the upper bound, assume p be the optimal fixed price and let g such that $2^g \leq p < 2^{g+1}$. Then $w_r \geq 2^r |B_r| \geq 2^{r-g-1} p |B_r|$. Thus $F^{(2)} \leq w_g + p \sum_{r=g+1}^{l-1} \frac{w_r}{2^{r-g-1}} \leq \max_k w_k \cdot (1 + \sum_{r=g+1}^{l-1} \frac{1}{2^{r-g-1}}) \leq 3 \max_k w_k$.

Lemma 2: Assume $9h \leq F$, K_1 be the set of buckets with at least two elements. Then $\sum_{j=1}^{l-1} w_j^d < 2 \sum_{i \in K_1} w_i^d$.

PROOF: First of all we argue that the bucket with weight (let m) $\max_k w_k$ has at least three elements. This is true because since $\max_k w_k \geq F^{(2)}/3 \geq 3h$ from lemma above. Hence, $w_m \in K_1$. So we have that $\sum_{i \in K_1} w_i^d \geq w_m^d \geq (3h)^d$. Let K_2 be the set of buckets with one element. If $i \in K_2$ then $w_i < 2^{i+1}$, thus since $2^{\lfloor \log h \rfloor} \leq h$ we have that $w_i < 2h2^{i-\lfloor \log h \rfloor}$ (only one element). Thus $\sum_{i \in K_2} w_i^d < (2h)^d \sum_{j=0}^{l-1} \frac{1}{2^{d(j+1)}} \leq 3h^d < \sum_{i \in K_1} w_i^d$. Finally, we have that $\sum_{j=0}^{l-1} w_j^d = \sum_{i \in K_1} w_i^d + \sum_{i \in K_2} w_i^d < 2 \sum_{i \in K_1} w_i^d$.

Lemma 3: Let v_1, \dots, v_m be real numbers in $[0, 1]$, with at least one $v_i = 1$. Then

$$\frac{\sum_{j=1}^m v_j^d}{\sum_{j=1}^m v_j^{d+1}} \leq m^{1/(d+1)}.$$

PROOF: If $\sum_{j=1}^m v_j^d \leq m^{1/(d+1)}$ then since the denominator ≥ 1 the inequality holds. Suppose the contrary. We will use Holder's inequality which says for every $p, q \geq 1$ with $\frac{1}{p} + \frac{1}{q} = 1$ and x_i, y_i positive sequences, $\sum_{j=1}^m x_j y_j \leq \left(\sum_{j=1}^m x_j^p \right)^{1/p} \cdot \left(\sum_{j=1}^m y_j^q \right)^{1/q}$. Substitute y_i for 1, x_i for v_i^d , p for $(d+1)/d$ and q for $d+1$. Then $\sum_{j=1}^m v_j^d \leq \left(\sum_{j=1}^m v_j^{d+1} \right)^{d/(d+1)} m^{1/(d+1)} \Rightarrow \sum_{j=1}^m v_j^{d+1} \geq \frac{1}{m^{(1/d)}} \left(\sum_{j=1}^m v_j^d \right)^{(d+1)/d}$. Thus $\frac{\sum_{j=1}^m v_j^d}{\sum_{j=1}^m v_j^{d+1}} \leq \frac{m^{(1/d)}}{m^{1/d(d+1)}} = m^{1/(d+1)}$.

Theorem: Assuming that $9h \leq F^{(2)}$, W_d is $O(3^d \log h^{1/(d+1)})$ -competitive w.r.t $F^{(2)}$.

PROOF: Let K_1 be the set of buckets with at least two elements and k be an arbitrary integer such that $k \in K_1$. Assume B'_k are the first $\lceil |B_k|/2 \rceil$ inserted in I_k and B''_k the remaining. It is obvious that since two elements $\in B_k$ are away from each other by a factor of two, then $3w(B'_k) \geq w(B_k)$. Thus for bidder i , the probability of the price threshold to be 2^k is bounded by the following:

$$\Pr[s_i = 2^k] = \frac{w_k(\{v_1, \dots, v_{i-1}\})^d}{\sum_{j=0}^{l-1} (w_j(\{v_1, \dots, v_{i-1}\}))^d} \geq \frac{\left(\sum_{i \in B'_k} v_i \right)^d}{\sum_{j=0}^{l-1} w_j^d} \geq \frac{w_k^d}{3^d \sum_{j=0}^{l-1} w_j^d}.$$

Thus summing over the buckets we have that $E[R] \geq \sum_{k \in K_1} \sum_{i \in B''_k} 2^k \frac{w_k^d}{3^d \sum_{j=0}^{l-1} w_j^d}$. If $i \in B_k$ then

$$2^k \geq v_i/2 \text{ and hence } E[R] \geq \sum_{k \in K_1} \sum_{i \in B''_k} \frac{v_i}{2} \frac{w_k^d}{3^d \sum_{j=0}^{l-1} w_j^d}.$$

Since $\sum_{i \in B'_k} v_i \geq \frac{1}{5} w_k$ (rather easy if we consider that two elements of the set are away from each other by a fac-

tor of two and we have one element less than B'_k). Thus using also lemma 2, letting $w^* = \max_k w_k$ we conclude that $E[R]/w^* \geq \sum_{k \in K_1} \frac{1}{10} \cdot \frac{w_k^{d+1}/(w^*)^{d+1}}{2 \cdot 3^d \sum_{t \in K_1} w_t^d/(w^*)^d}$. Thus from lemma 3, we have that $E[R]/w^* \geq \frac{1}{3^d \cdot 20|K_1|^{1/(d+1)}} \geq \frac{1}{3^d \cdot 20(\log h)^{1/(d+1)}}$. Using from lemma 1 that $w^* \geq F^{(2)}/3$, we complete the proof.

An obvious corollary is that substituting d for $\sqrt{\log \log h}$, RSOP with $F \geq 9h$ is $O(\exp(\sqrt{\log \log h}))$ -competitive.

3.5.2 WM Algorithm

Auer et al [1] presented WM Algorithm, an algorithm that is also considered in the category of the full information case. The main idea is that we have a set X of possible fixed prices. Every time a bidder comes (let the i -th bidder), the algorithm picks price $x_k \in X = \{x_1, \dots, x_m\}$ with a probability $p_k(i)$ (that depends on the revenue). After learning the true valuation of the bidder (it is a bid-independent, thus truthful and $b_i = v_i$), the probability distribution is updated and always depends on the current revenue $r_k(i)$ of each price $x_k \in X$, namely $r_k(i) = |\{1 \leq j \leq i : v_j \geq x_k\}| \cdot x_k$. Below is the algorithm.

<p><i>Algorithm WM</i></p> <p>Parameters: Reals $a \in (0, 1]$ and $X \in [1, h]^m$ Initialization: For each expert k, initialize $r_k(0) = 0$, $w_k(0) = 1$. For each bidder $i = 1, \dots, n$: Chose price $s_i = x_k$, with probability $p_k(i) = \frac{w_k(i-1)}{\sum_{j=1}^m w_j(i-1)}$. For each price x_k, update $r_k(i) = r_k(i-1) + d_k(i)x_k$ where $d_k(i) = 1$ if $v_i \geq x_k$ else $d_k(i) = 0$ and $w_k(i) = (1 + a)^{r_k(i)/h}$.</p>

For the performance analysis, we will prove that $E[R] \geq (1 - \frac{a}{2})F_X - \frac{h \ln m}{a}$, where F_X is the optimal fixed price revenue restricted to fixed prices in X .

Theorem: $E[R] \geq (1 - \frac{a}{2})F_X - \frac{h \ln m}{a}$.

PROOF: Let $W(i) = \sum_k w_k(i)$ (the sum of weights after the arrival of the i -th bidder) and $g_k(i) = d_k(i)x_k$. Thus the expected revenue from bidder $i+1$ is $g_{i+1} =$

$\frac{\sum_{k=1}^m g_k(i+1)w_k(i)}{W(i)}$. Additionally we can bound $W(i+1)$ in terms of $W(i)$. $W(i+1) = \sum_{k=1}^m (1+a)^{r_k(i)/h} (1+a)^{g_k(i+1)/h} \leq \sum_{k=1}^m w_k(i) (1+a \cdot g_k(i+1)/h)$ (comes from applying bernoulli's inequality). Thus $W(i+1) \leq W(i) \left(1 + a \frac{\sum_{k=1}^m g_k(i+1)w_k(i)}{hW(i)}\right) = W(i)(1 + ag_{i+1}/h)$. From the above inequality, over all i from 1 to n we have that $W(n) \leq W(0) \prod_{i=1}^m (1 + ag_i/h) = m \prod_{i=1}^m (1 + ag_i/h)$ since $W(0) = m$. Finally $W(n) \geq (1+a)^{F_X/h}$ (from the sum of all weights, we take only the largest term). Hence $\ln(1+a)F_X/h \leq \sum_{i=1}^m \ln(1 + ag_i/h) + \ln m$. Thus, using the fact that $x - x^2/2 \leq \ln(1+x) \leq x$ we have that $(a - a^2/2)F_X/h \leq aE[R]/h + \ln m \Rightarrow (1 - \frac{a}{2})F_X \leq E[R] + (h \ln m)/a$.

If we want to get rid of F_X and find an inequality with respect to F , we do the following. We assume that $X = \{1, 1+b, (1+b)^2, \dots, (1+b)^{\log_{1+b} h}\}$ and that $a = b = \epsilon/3$ and using the fact that $F \leq (1+b)F_X$ we conclude that $(1 - \frac{\epsilon}{6})F \leq E[R] + O((h \ln \log_{1+\epsilon/3} h)/\epsilon) = E[R] + O(h \ln \ln h / (\epsilon \ln(1 + \epsilon/3))) = E[R] + O(h \ln \ln h / \epsilon^2)$.

3.5.3 Posted price auctioning - partial information case

As opposed to 3.5.1,3.5.2 in this section we examine the partial information case. When a bidder comes, the auctioneer selectes a price p and sell the item to the bidder. By the decision of the bidder, we just learn if his bid is less or larger than the price but the bid value actually. As it is seems that the auctioneer is in worse situation than in full information case, we can design algorithms (based to WM) to achieve good competitive ratio. Before we mention Exp3 algorithm that solves the partial information case, we have to give some information about posted price mechanisms (Exp3 that we describe below belongs to this family of mechanisms).

Formally, an online posted price mechanism can be defined as follows (the randomized posted price mechanisms is just a distribution over deterministic online posted price mechanisms):

Definition: (Online Posted Price Mechanism)

Any class of functions g_k from $\{0, 1\}^{k-1}$ to R defines an deterministic online posted price mechanism as follows. For each agent i ,

1. For $j < i$, let $x_j = 1$ if agent j accepted offer $z_j = g_j(x_1, \dots, x_{j-1})$, and 0 otherwise.
2. $z_i \leftarrow g_i(x_1, \dots, x_{i-1})$
3. If $z_i \leq b_i$ sell to bidder i at price z_i .

4. Otherwise, reject bidder i .

Intuitively, we see that z_i is a function that depends on z_1, \dots, z_{i-1} and not bidders' valuations. Exp3 is a randomized mechanism that belongs to the family of posted price mechanisms. Consider Exp3, as it can be seen in Auer et al [1]. and Blum et al.[8]:

Algorithm Exp3

Parameters: Reals $a \in (0, 1]$, $\gamma \in (0, 1]$ and $X \in [1, h]^m$

Initialization: For each k , initialize $r_k(0) = 0$, $w_k(0) = 1$.

For each bidder $i = 1, \dots, n$:

Set the posted price s_i to be x_k with probability $p'_k(i) = (1 - \gamma)p_k(i) + \frac{\gamma}{m}$, where

$$p_k(i) = \frac{w_k(i-1)}{\sum_{j=1}^m w_j(i-1)}.$$

For the chosen price $s_i = x_l$, if bidder i accepts, set $g_l(i) = s_i$ else set $g_l(i) = 0$.

$$\text{Set } g'_l(i) = \frac{\gamma g_l(i)}{m p'_l(i)}$$

For all other k , set $g'_k(i) = 0$.

For all k , update $r_k(i) = r_k(i-1) + g'_k(i)$ and $w_k(i) = (1 + a)^{r_k(i)/h}$.

The algorithm works as follows. Again we have a set $X = \{x_1, \dots, x_m\}$ with possible fixed prices (as in WM) and parameters a, γ . a plays the same role as in WM but γ is used for mixing the probability distribution of choosing an element of X . At step i , we take the probability distribution $p_k(i)$ (as defined in WM algorithm) and mix it with the uniform distribution (that's why we multiply γ with $\frac{1}{m}$) to obtain a modified probability distribution $p'_k(i)$, which is then used to select a threshold price s_i . Following each bidder's accept/reject decision, we use the information obtained (if $v_i \geq s_i$ or not) to formulate a simulated gain vector, which is then used to update the weights. An interesting difference from WM, is that $r_k(i) = |\{1 \leq j \leq i : v_j \geq x_k, k \text{ was chosen when } j \text{ arrived}\}| \cdot x_k$.

For competitive analysis, we use the theorem below, as it can be seen in Blum et al. [8]. It says that we can achieve a constant competitive ratio with respect to optimal fixed price revenue (F) when F is large "enough".

Theorem: There exists a constant $c(\epsilon)$ such that for all valuation sequences with $F \geq ch(\log h) \log \log h$, mechanism Exp3 is $(1 + \epsilon)$ -competitive relative to the optimal fixed price revenue F .

PROOF: We use theorem 3.1 from Auer et al. [1] and thus $F - E[R] \leq (e-1)\gamma F +$

$\frac{h}{\ln(1+a)} \frac{m \ln m}{\gamma}$. Thus $F(1-\gamma(e-1)) \leq E[R] + \frac{h}{\ln(1+a)} \frac{m \ln m}{\gamma}$. Hence for $a = \epsilon$, $\gamma = \epsilon/(e-1)$ and $m = \ln h$ we have that $F(1-\epsilon) \leq E[R] + O(c(\epsilon)h(\ln h)(\ln \ln h))$.

3.6 Conclusion

In this section, we examine the auctions where the auctioneer has an unlimited supply of goods. We saw both the offline and online case and focus on the randomized online pricing algorithms for solving the problems. It is remarkable, as it also said in Hartline et al [16] and Blum et al [8] that the techniques of the algorithms discussed can be used in limited supply setting too (so long as the sequence of bids can be truncated as soon as we run out of items to sell), but with worse competitive ratio. Also, the game theoretic model we discussed in this section, doesn't consider the case the bidders misreport their arrival (that is, come later or earlier in this case) and the expected profit was not taken over the permutation of the bidders' sequence (as opposed to the previous chapter).

Chapter 4

Other settings for pricing in online auctions

In this chapter, we describe other settings-modifications of online auctions that are interesting and of great importance. We haven't done much analysis about the proofs of our theorems-claims, most of them are results that we found in different papers. For the section 1, we examine online auctions, where the pricing may differ for every unit of the good for a specific bidder and depends on a non-decreasing function. In section 2, we consider online auctions where the goods expire one at a time we want to find a price in order to sell each one at a bidder. Finally, in section 3, we deal with online auctions, where each bidder wants to buy a specific set of goods and only that set.

4.1 Supply Curves

In this subsection, we consider an auction with k identical indivisible goods (if k is very large then this case is viewed as auctioning one divisible good). For each player i , we denote his marginal valuation $v_i(j)$, the additional value gained from the j th good. Thus the total valuation of player i that buys q goods is $\sum_{j=1}^q v_i(j)$. We assume $\forall i, j : v_i(j+1) \leq v_i(j)$ and player i pays P_i , so the utility of i is:

$$U_i(q, P_i) = \sum_{j=1}^q v_i(j) - P_i$$

At some time t_i , player i determines his valuation and must declare a bid b_i to the auctioneer. A bid is a function $b : [1..k] \rightarrow R$ (non-increasing) which may differ from v_i and when it differs we say that player i lies (in order to increase his utility). The auctioneer must answer immediately the quantity which he will sell to the bidder i and the total payment i must pay.

Supply curves for on-line auctions:

An online auction that fixes a function $p_i(q)$ based on previous bids before receiving i 's bid is called "based on supply curves". The following must also hold:

- The quantity q_i that is sold to i maximizes $\sum_{j=1}^{q_i} (b_i(j) - p_i(j))$ (bidder's utility)
- $P_i = \sum_{j=1}^{q_i} p_i(j)$

4.1.1 Incentive Compatibility for Supply Curves

Assuming that $p_i(q)$ is non-decreasing $\forall q$ with $1 \leq q \leq k$, we have that q_i is the largest q such that $b_i(q) \geq p_i(q)$. As we can see from the following theorem, such a mechanism is strategyproof (players will not increase their utility by deviating, declaring different bid).

Theorem: A deterministic online auction is incentive compatible (strategyproof) if and only if it is based on supply curves.

PROOF: We prove the following lemmas and the theorem follows:

Lemma 1: An online auction that is based on supply curves is incentive compatible.

PROOF: Let $b_i \neq v_i$. Suppose the quantity sold for this bid is q'_i , and for the truthful bid is q_i . Then it is the case that:

$$U_i(q_i) = \sum_{j=1}^{q_i} (v_i(j) - p_i(j)) \geq \sum_{j=1}^{q'_i} (v_i(j) - p_i(j)) = U_i(q'_i)$$

because of the first condition of supply curves definition. So it follows that the auction is truthful.

Lemma 2: An incentive compatible online auction is based on supply curves.

PROOF: Assume an incentive compatible online auction A . Firstly, we argue that P_i is determined uniquely by the quantity sold to i . For contradiction, suppose two bids v_i, v'_i . Let P, P' be the payment for declaring v_i, v'_i and receiving quantity q . Without loss of generality, assume that $P < P'$. Then a player with bid v'_i will increase his utility function by declaring $v_i(q)$, because $\sum_{j=1}^q v'_i(j) - P > \sum_{j=1}^q v'_i(j) - P'$. This is a contradiction because A is a truthful mechanism.

As payment P_i depends only on quantity, we observe that $p_i(q) = P_i(q) - P_i(q - 1)$. Payment $P_i(q)$ then is denoted $\sum_{j=1}^q p_i(j)$ (telescopic sum) where $P_i(0) = 0$ from our assumptions. Finally for the total payment, let q_i be the quantity that maximizes $U_i(q) = \sum_{j=1}^q v_i(j) - P_i(q) = \sum_{j=1}^q (v_i(j) - p_i(j))$, b_i be the bid for the quantity q_i and assume that A gives quantity q'_i for bid v_i . If $q'_i \neq q_i$ then player i can increase his utility function by declaring b_i (because q_i maximizes the previous sum) which is contradiction, as A is a truthful mechanism.

Even if the previous theorem allow us to use any online auction that is based on supply curves in order to achieve truthfulness, allowing any (non-increasing) marginal valuation functions may increase significantly the complexity of presenting the valuation function to the auctioneer. In order to overcome this difficulty, we use modified auctions, where the supply curves is known to every player and each bid is just a point on the curve. Before we move on to competitive analysis of some algorithms related to online auctions based on supply curves, we give another definition that we will use below. An on-line auction is called "based on a global supply curve $p(q)$ " if it is based on supply curves and if $p_i(q) = p(q + \sum_{j=1}^{i-1} q_j)$, where q_j is the quantity sold to the j 'th bidder.

4.1.2 Competitive Analysis

In this section we mention theorems for the case of k indivisible goods that referred to competitive ratio of online mechanism for the online auctions "based on supply curves". We assume that all marginal valuations are taken from some known interval $[p_s, p_f]$. Firstly, we need to define *Revenue*, *Social Efficiency* and *Competitiveness* for this kind of auction. The revenue of an auction A , is the utility function of the auctioneer, denoted by:

$$R_A = \sum_i P_i + p_s(k - \sum_i q_i)$$

where P_i is the total price paid by player i and q_i the quantity he bought. Moreover, the social efficiency of an auction A is the sum of the resulting utilities of all players, including the auctioneer:

$$E_A = \sum_i \sum_{j=1}^{q_i} v_i(j) + p_s(k - \sum_i q_i)$$

Finally, an on-line auction A is c -competitive with respect to the revenue if $R_A \geq R_{vic}/c$ (R_{vic} is revenue for Vickrey offline auction). Similarly, A is c -competitive with respect to the social efficiency if $E_A \geq E_{vic}/c$ (E_{vic} is social efficiency for

Vickrey offline auction). For $k = 1$, it can be seen at R.El-Yaniv - Competitive

solutions for online financial problems, the auctioneer gives the good to player i such that $v_i(1) > p$ for a fixed price $p = \sqrt{p_s \cdot p_f}$. The competitive ratio of the algorithm above (reservation price algorithm, denoted RPP) is $\sqrt{\phi}$, where $\phi = \frac{p_f}{p_s}$. For the general case $k \geq 2$, we introduce the global supply curve:

$$p(j) = p_s \cdot \phi^{\frac{j}{k+1}}$$

Then the following theorem as seen at Lavi's, Nisan's paper (*Competitive Analysis of Incentive Compatible On-Line Auctions*) holds:

Theorem: The online auction based on previous supply curve is $k \cdot \phi^{\frac{1}{k+1}}$ -competitive with respect to the revenue and to the social efficiency.

In the last paper, there is also another theorem referred to lower bound of competitive ratio of any incentive compatible online auction of k goods, which is also based on supply curves as it has been already proven.

Theorem: Any incentive compatible online auction of k goods has a competitive ratio of at least $\phi^{\frac{1}{k+1}}$ with respect to the revenue and to the social efficiency.

4.2 Expiring Items

4.2.1 Definition

Model: In this section, we consider online auctions, where the items must be sold during a certain period (different for each one) and then they expire. Specifically, we wish to sell M identical items with different expiration times. W.l.o.g. we assume that the first item expires at time 1, the second at time 2, and so on. Each item must be sold (and received by the buyer) at or before its expiration time. For the bidders, we assume that they arrive over time. Player i arrives to the market at time $r(i) \in N$ and stays in the market for some fixed period of time, until his departure time $d(i) \in N$. Each player desires only one item (unit demand), that expires no earlier than his arrival time. He must receive it at or before his departure time. Player i obtains a value of $v(i)$ from receiving such an item, otherwise his value is 0. We assume w.l.o.g. that different players have different values. We assume the private value model with quasi-linear utilities:

player i privately obtains his variables $r(i)$, $d(i)$, and $v(i)$, and acts rationally in order to maximize his own utility: his obtained value minus his price. A player may arrive at or after his true arrival time, and declare or act as if he has any value, and any deadline.

4.2.2 Approximation ratio for deterministic truthful mechanism

Our goal is to maximize the social welfare, the sum of values of players that receive an item. From the theorem below follows that we can't achieve a better approximation than M with respect to social welfare if our deterministic mechanism is truthful (M is the number of items).

Theorem: Any truthful deterministic mechanism for our online allocation problem cannot always obtain more than $1/M$ fraction of the optimal welfare.

PROOF: Let T_i be the domain of all valid player types $(r(i), v(i), d(i))$. We use the following lemma:

Lemma: Fix some truthful deterministic mechanism with some approximation ratio c . Then, for any player i with $r(i) = 1$ there exists a price function $p_i : T_{-i} \rightarrow R$ such that, for any combination of players that arrive at time 1, b_{-i} :

- If $v(i) > p_i(b_{-i})$ then i wins item 1 and pays $p_i(b_{-i})$ (regardless of his deadline).
- If $v(i) < p_i(b_{-i})$ then i does not win any item.

Fix any price functions $p_i : T_{-i} \rightarrow R$. For any $\epsilon > 0$ we will show that there exist player types b_1, \dots, b_M such that, for all i , $r(i) = 1$, $d(i) = M$, $1 \leq v(i) \leq 1 + \epsilon$, and $v(i) \neq p_i(b_{-i})$. To verify that such types exist, fix $L > M$ real values in $[1, 1 + \epsilon]$. Choose M values $v(i)$ uniformly at random from these L values. Then, for any given i , $Pr(v(i) = p_i(v(-i))) \leq 1/L$, as the values were drawn i.i.d. Thus, $Pr(\exists i, v(i) = p_i(v(-i))) \leq M/L < 1$, hence there exist a choice of values with $v(i) \neq p_i(v(-i))$ for all i . By the above lemma, it follows that the mechanism can obtain welfare of at most $1 + \epsilon$, while the optimal allocation is at least M , and the theorem follows.

It is remarkable that there exists truthful, deterministic online mechanism that obtains at least $\frac{1}{M}$ fraction of the optimal welfare. The algorithm is the following: for any player i , set p_i to be the highest bid received in time slots $1, \dots, t$, excluding i 's own bid. Sell item t to player i if and only if $v(i) > p_i$, for a price of p_i . It is

easy to see that the mechanism is truthful. Additionally, since the player with the highest value always wins, a at least $1/M$ fraction of the optimal is obtained.

4.2.3 Semi-Myopic Allocation Rules

we say that a set S of players is independent with respect to items t, \dots, M if there exists an allocation of (part of) the items t, \dots, M such that every player in S receives an item. The *current best schedule* at time t , S_t , is the allocation with maximal value among all allocations of items t, \dots, M to the active players, A_t . Define

$$f_t = \{j \in S_t \mid S_t \text{ } j \text{ is independent w.r.t items } t+1, \dots, M\}$$

The set f_t contains all players that can receive item t , when one plans to allocate items t, \dots, M to the players of S_t (i.e. these are all the potentially first players). Now define the critical value at time t , v_t^* as:

$$0 \text{ if } S_t \text{ is independent w.r.t. items } t+1, \dots, M \\ \min_{j \in f_t} \{v(j)\} \text{ otherwise}$$

All active players with value larger than v_t^* must belong to S_t , because of its optimality (w.l.o.g the first player in S_t has value v_t^* , and if there was a higher valued player outside of S_t , we could switch between them and increase the value of S_t). Thus, it seems reasonable not to allocate item t to a player with value less than v_t^* , as this player cannot belong to any optimal allocation. Surprisingly, this condition is enough to obtain approximately optimal allocations. Below, we mention *the online iterative auction* algorithm that solves the problem above and some definitions we need for the further analysis. Finally, the theorem below shows that *the online iterative auction* algorithm is 3-approximation with respect to the social welfare.

The online iterative auction: The Online Iterative Auction constantly maintains a current price p_t and a current winner win_t for every item t . These are initialized to zero at $t = 0$, and updated according to players' actions at each time t , as follows:

- Each player, in his turn, may place his name as the temporary winner of some item t' , causing the previous winner to be deleted, and the price to increase by some fixed small δ . A player cannot perform this action, and must relinquish his turn, if he is already a temporary winner.

- When none of the players that are not temporary winners wishes to place their names somewhere, the time t phase ends: item t is sold to the player win_t for a price of $p_t - \delta$.
- At time $t + 1$ the prices and temporary winners from time t are kept. If additional players arrive then the auction continues according to the above rules.

Definition 1: Player i has a **myopic strategy** in the iterative auction if, in his turn, he always places his name on the item $t \leq d(i)$ with the minimal price, unless the minimal price $\geq v(i)$, in which case he does not bid at all.

Definition 2: Player i is **semi-myopic** if, in his turn, i bids on some item t with $p(t) \leq v(i)$ and $r(i) \leq t \leq d(i)$ (not necessarily the one with the lowest price). If there is no such item, i stops participating.

Definition 3: (A semi myopic allocation rule) An allocation rule is semi myopic if every item t is sold at time t to some player j with $v(j) \geq v_t^*$

Theorem: If all players are semi-myopic then the online iterative auction achieves at least one third of the optimal welfare:

$$v(OPT) \leq 3 \cdot v(ON) + 2 \cdot M \cdot \delta$$

where OPT, ON are the optimal, online allocations, respectively.

The theorem above can be proved if firstly we show that, under any semi-myopic behavior, the online iterative auction follows a semi myopic allocation rule, hence obtains the desired welfare level (fraction of $\frac{1}{3}$).

4.3 Graph Vertex Pricing

4.3.1 Definition

In this section we are dealing with auctions, where customers have valuations over pairs of items (e.g., a computer and a monitor), and will only purchase if the combined price of the items in their pair is below their value. In that case, we can model the problem as a (multi) graph, where each edge e has some valuation

w_e , and our goal is to set prices p_i on the vertices of the graph to maximize total profit which is denoted by

$$Profit(\mathbf{p}) = \sum_{e=(i,j):w_e \geq p_i + p_j} (p_i + p_j)$$

where \mathbf{p} is the vector of individual prices. In a more general way, if bidders have valuations over larger subsets ($k > 2$), we can model our computational problem as one of pricing vertices in a *hypergraph*. A hyperedge e connects a set of items bidder e wants to buy followed by value w_e . So for this case we want the price vector \mathbf{p} , (called p^*) that maximizes

$$Profit(\mathbf{p}) = \sum_{\{e:w_e \geq \sum_{i \in e} p_i\}} \sum_{i \in e} p_i$$

The problem above is called k -hypergraph vertex pricing.

4.3.2 The Online k -Hypergraph Vertex Pricing

For the offline case of k -Hypergraph Vertex Pricing, we will discuss an algorithm that is $O(k)$ -approximation. Assume $G(V, E)$ an hypergraph. The algorithm is the following:

1. Randomly partition V into V_L and V_{rest} by placing each node into V_L with probability $\frac{1}{k}$.
2. Let E' be the set of edges with exactly one endpoint in V_L . Ignore all edges in $E - E'$.
3. Set prices in V_{rest} to 0 and set prices in V_L optimally with respect to edges in E' .

Theorem: The algorithm above is $O(k)$ -approximation.

PROOF: Let $OPT_{i,e}$ be the profit made by p^* , selling item i to bidder e (obviously, $OPT_{i,e} \in \{0, p_i^*\}$) and $OPT = Profit(p^*) = \sum_{i \in V, e \in E} OPT_{i,e}$. From the third step of the algorithm above, it's easy to see that the total profit is at least $\sum_{i \in V_L, e \in E'} OPT_{i,e}$. So we need to find a lower bound for $\mathbf{E}[\sum_{i \in V_L, e \in E'} OPT_{i,e}]$. Let $X_{i,e}$ be random variable such that $X_{i,e} = 1$ if ($i \in V_L$ and $e \in E'$), and $X_{i,e} = 0$ otherwise. Therefore, $\mathbf{E}[X_{i,e}] = Pr[i \in V_L \wedge e \in E'] \geq \frac{1}{k}(1 - \frac{1}{k})^{|e|-1} \geq \frac{1}{k}(1 - \frac{1}{k})^{k-1}$. Thus, $\mathbf{E}[\sum_{i \in V_L, e \in E'} OPT_{i,e}] = \mathbf{E}[\sum_{i \in V_L, e \in E} OPT_{i,e} X_{i,e}] = \sum_{i \in V_L, e \in E} OPT_{i,e} \mathbf{E}[X_{i,e}] \geq OPT \cdot \frac{1}{k}(1 - \frac{1}{k})^k = \frac{OPT}{O(k)}$.

For the online case, we use the algorithm described above, adapted to the posted-price setting by using *Exp3* algorithm. The only tricky issue is that a customer who chooses not to buy anything must be fed in as a non-buyer to all of the online algorithms, in order to ensure that the sequence of customers fed into algorithm i is a superset of the true customers for that item (notice that interval $[1, h]$ must be known in advance).

References

- [1] Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire
The nonstochastic multiarmed bandit problem *Siam J. Comput.* c 2002 Society
for Industrial and Applied Mathematics Vol. 32, No. 1, pp. 48-77
- [2] Moshe Babaioff, Nicole Immorlica, David Kempe and Robert Kleinberg. Online
Auctions and Generalized Secretary Problems.
- [3] Moshe Babaioff, Nicole Immorlica, and Robert Kleinberg. Matroids, Secretary
Problems, and Online Mechanisms
- [4] Moshe Babaioff, Michael Dinitz, Anupam Gupta, Nicole Immorlica,
and Kunal Talwar. Secretary problems: Weights and discounts.
- [5] Moshe Babaioff, Nicole Immorlica, David Kempe and Robert Kleinberg. 2007.
A knapsack secretary problem with applications.
- [6] Maria-Florina Balcan and Avrim Blum. Algorithms and Approximation
Online Mechanisms for Item Pricing
- [7] Ziv Bar-Yossef, Kirsten Hildrum, Felix Wu. Incentive-Compatible Online
Auctions for Digital Goods 2002, Symposium on Discrete Algorithms
- [8] Avrim Blum, Vijay Kumar, Atri Rudra and Felix Wu. Online Learning in
Online Auctions
- [9] Avrim Blum, Jason D. Hartline. Near-Optimal Online Auctions
- [10] Edward H. Clarke. Multipart pricing of public goods. *Public Choice*, 11(1),
September 1971.
- [11] U Feige, A Flaxman, J Hartline, R Kleinberg. On the Competitive Ratio
of the Random Sampling Auction
- [12] Thomas S. Ferguson. Who Solved The Secretary Problem
Statistical Science 1989, Vol. 4, No. 3, 282-296

- [13] Andrew V. Goldberg , Jason D. Hartline, Andrew Wright Competitive Auctions and Digital Goods 2001, Symposium on Discrete Algorithms
- [14] T. Groves. Incentives in teams. *Econometrica*, 41:617-631, 1973
- [15] Mohammad T. Hajiaghayi, Robert Kleinberg, and David C. Parkes. Adaptive Limited-Supply Online Auctions
- [16] Jason D. Hartline, Robert McGrew. From Optimal Limited to Unlimited Supply Auctions
- [17] Ankur Jain Irfan Sheriff Shashidhar Mysore. Optimal Clearing of Supply /Demand Curves
- [18] Robert Kleinberg. A Multiple-Choice Secretary Algorithm with Applications to Online Auctions
- [19] Ron Lavi and Noam Nisan. Online Ascending Auctions for Gradually Expiring Items (Extended Abstract)
- [20] Ron Lavi and Noam Nisan. Competitive Analysis of Incentive Compatible On-Line Auctions
- [21] Noam Nisan, Tim Roughgarde, Eva Tardos, and Vijay V. Vaziran Algorithmic Game Theory
- [22] William Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of Finance*, 16(1):8-37, 1961.

Appendix A

Experimental Analysis

A.1 2-Hypergraph Vertex Pricing

We implemented in C programming language the algorithm described in section 4.3.2. We used `rand()` function to create random graphs with specific number of nodes and the weight of each edge is 2. It is obvious that the optimal profit $= 2 \cdot E$, where E is the number of edges. Below you can see the implementation and a graph that gives the approximation ratio as it followed from the output of the program (expected $O(k)$ -approximation, where $k = 2$).

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define n 2000
int matrix[n+1][n+1];
int vl[n+1];
int optprofit;
int edges;
int profit;
int main()
{
    int i,j,k;
    srand( time(NULL) );
    for (i=1;i<=n;i++)
        for (j=1;j<i;j++)
            {
                k = rand() %2;
                if (k==0)
                    {matrix[i][j] = -1;
                    matrix[j][i] = -1;}
                else
```

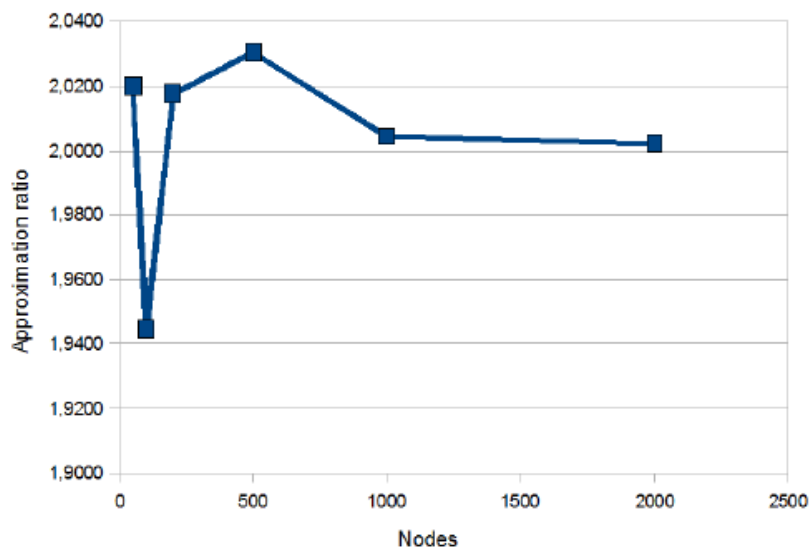
```

    {
    matrix[i][j] = 2;
    matrix[j][i] = 2;
    edges++;
    }
}

optprofit = 2*edges;
for (i=1;i<=n;i++)
vl[i] = rand()%2;

for (i=1;i<=n;i++)
  for (j=1;j<i;j++)
  {
    if (vl[i]+vl[j] == 1 && matrix[i][j]==2) profit +=2;
  }
printf("%d %d %lf",profit,optprofit,((double)optprofit / (double)profit));
return 0;
}

```



A.2 RSOP

We implemented in C programming language the algorithm described in section 3.4.2 RSOP mechanism for the multi-price case. We used `rand()` function for bid values of the bidders, and for the partition of the bidders to the two sets. Finally we have as an output the competitive ratio with respect to the revenue of the auctioneer. As optimal revenue for the untruthful case we mean $F^{(2)}$ (see section 3.2.2 for definitions).

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define n 30

int bid[n+1];
int select[n+1];
int seta[n+1];
int setb[n+1];
int numa,numb;
int main()
{
    int i,j,temp;
    srand( time(NULL) );
    for (i=1;i<=n;i++)
    {
        bid[i] = rand();
        j = rand()%2;
        if (j==0)
        {numa++;
        seta[numa] = bid[i];}
        else
        {numb++;
        setb[numb] = bid[i];}
    }

    for (i=1;i<=n;i++)
    for (j=1;j<i;j++)
    {
        if (bid[i]>bid[j])
        {
```

```

    temp = bid[i];
    bid[i] = bid[j];
    bid[j] = temp;
}
}
int fbench=0;

for (i=2;i<=n;i++)
{
    if (fbench < i*bid[i])
    fbench = i*bid[i];
}

for (i=1;i<=numa;i++)
for (j=1;j<i;j++)
{
    if (seta[i]>seta[j])
    {
        temp = seta[i];
        seta[i] = seta[j];
        seta[j] = temp;
    }
}

for (i=1;i<=numb;i++)
for (j=1;j<i;j++)
{
    if (setb[i]>setb[j])
    {
        temp = setb[i];
        setb[i] = setb[j];
        setb[j] = temp;
    }
}

int thresa,thresb;
int opta=0,optb=0;
for (i=1;i<=numa;i++)
{
    if (opta < i*seta[i])
    {opta = i*seta[i];
    thresb = seta[i];}
}

```

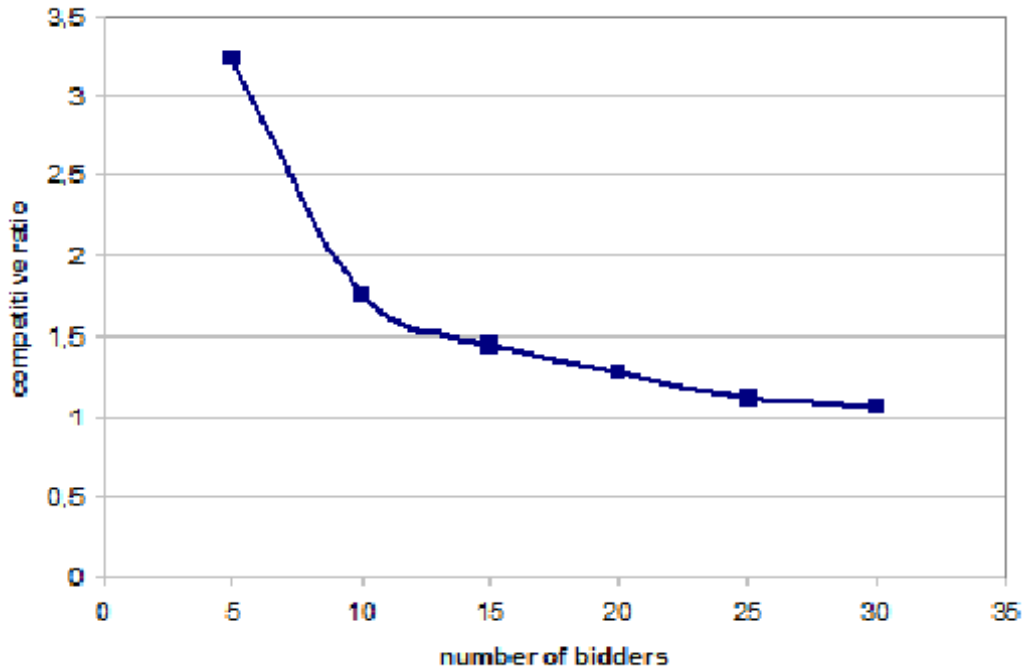


```

for (i=1;i<=numb;i++)
{
  if (optb < i*setb[i])
    {optb = i*setb[i];
    thresa = setb[i];}
}
int sol =0;
i=1;
while(seta[i] >= thresa)
i++;
sol+= thresa*(i-1);
i=1;
while(setb[i] >= thresb)
i++;
sol+= thresb*(i-1);
printf("%lf",((double)fbench / (double)sol));
return 0;
}

```

For the number of bidders n varying, we have the following graph.



A.3 Online iterative auction

We implemented in C programming language the online iterative algorithm described in section A.2. We used `rand()` function for utility functions, arrival and departure time of bidders. We assume that $d = 5$ and we took cases for $10 \leq M \leq 100$ with step 10 and also we took $n = 5000$ the number of bidders. The expected competitive ratio was ≤ 3 (if we subtract the $2 \cdot M \cdot d$).

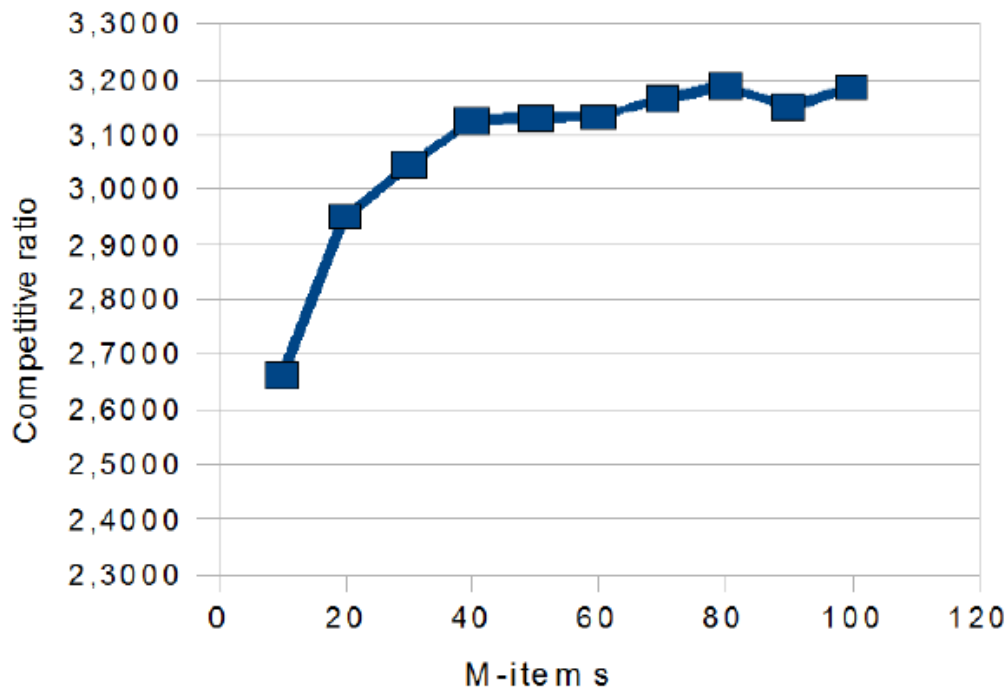
```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define n 5000
#define d 5
#define M 100

int utility[n+1];
int departure[n+1];
int arrival[n+1];
```

```

int sort[n+1];
int winners[n+1];
int itemtakes[n+1];
int main()
{
    int i,j,t,oldplayer,argmax,temp;
    double cpt=0,welfare=0,optwelfare=0,maxprice,secmaxprice;
    srand(time(NULL));
    for (i=1;i<=n;i++)
    {
        utility[i] = rand();
        arrival[i] = rand()%M;
        departure[i] = arrival[i] + (rand()%M+1);
    }
    for (t=1;t<=M;t++)
    {
        for (j=1;j<=n;j++)
            if (arrival[j]<= t && departure[j]>=t && utility[j]>= cpt
&& winners[j]==0)
            {
                cpt+=d;
                oldplayer = itemtakes[t];
                winners[oldplayer] = 0;
                winners[j] = 1;
            }
        welfare += cpt - d;
    }
    for (i=1;i<=n;i++)
    for (j=1;j<i;j++)
    if (utility[i] > utility[j])
    {
        temp = utility[i];
        utility[i] = utility[j];
        utility[j] = temp;
    }
    optwelfare = M*utility[M+1];
    printf("%lf %lf %lf",welfare,optwelfare,((double)(optwelfare-
2.0*M*d)/(double)welfare));
    return 0;
}

```



A.4 Price Sampling Algorithm

We implemented in C programming language the price sampling algorithm described in section 2.3. We used `rand()` function to create the bipartite graph, with max degree of left nodes $\leq d$ and the utility functions of the agents. The graph is constructed in a way that the optimal solution is $\sum_{j=n-m+1}^n j$, where m is the number of items (right set). Below you see the implementation and a graph that gives the approximation ratio to the optimal output. We expected a $4d$ -approximation ratio.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define n 2000
#define m 300

int matrix[n+1][m+1];
int vl[n+1];
```

```

int price[m+1];
int akmes[n+1];
int visited[m+1];

int main()
{
    int i,j,optsol=0,k,d;
    scanf("%d",&d);
    srand( time(NULL) );

    for (i=1;i<=n;i++)
        for (j=1;j<=m;j++)
            {
                k = rand()%2;
                if (k==0)
                    matrix[i][j] = 0;
                else
                {
                    if (akmes[i]<d-1)
                    {matrix[i][j] = 1;
                    akmes[i]++;}
                }
            }

    for (i=1;i<=n;i++)
        vl[i] = i;

    for (i=n-m+1;i<=n;i++)
    {
        matrix[i][i+m-n]=1;
        optsol+=vl[i];
    }

    int s = n/2,t,minim,mini,sol=0;

    for (j=1;j<=m;j++)
    {
        int maxx = 0;
        for (i=1;i<=s;i++)
        {
            if (matrix[i][j]==1 && maxx < vl[i])
                maxx = vl[i];
        }
    }
}

```

```

    }
    price[j] = maxx;
}

for (i=s+1;i<=n;i++)
{
    minim = 1000000;
    for (j=1;j<=m;j++)
        if (visited[j]==0 && vl[i]>price[j] && minim > price[j])
            {minim = price[j];
             mini = j;}
    if (minim == 1000000)
        minim = 0;
    visited[mini] = 1;
    sol +=minim;
}
printf("%lf",((double)optsol / (double)sol));
return 0;
}

```

