CHAPTER 11

Combinatorial Auctions

Liad Blumrosen and Noam Nisan

Abstract

In combinatorial auctions, a large number of items are auctioned concurrently and bidders are allowed to express preferences on bundles of items. This is preferable to selling each item separately when there are dependencies between the different items. This problem has direct applications, may be viewed as a general abstraction of complex resource allocation, and is the paradigmatic problem on the interface of economics and computer science. We give a brief survey of this field, concentrating on theoretical treatment.

11.1 Introduction

A large part of computer science as well as a large part of economics may be viewed as addressing the "allocation problem": how should we allocate "resources" among the different possible uses of these resources. An auction of a single item may be viewed as a simple abstraction of this question: we have a single indivisible resource, and two (or more) players desire using it – who should get it? Being such a simple and general abstraction explains the pivotal role of simple auctions in mechanism design theory.

From a similar point of view, "combinatorial auctions" abstract this issue when multiple resources are involved: how do I allocate a collection of interrelated resources? In general, the "interrelations" of the different resources may be combinatorially complex, and thus handling them requires effective handling of this complexity. It should thus come as no surprise that the field of "combinatorial auctions" – the subject of this chapter – is gaining a central place in the interface between computer science and economics.

11.1.1 Problem Statement

The combinatorial auction setting is formalized as follows: There is a set of m indivisible items that are concurrently auctioned among n bidders. For the rest of this chapter we

will use n and m in this way. The combinatorial character of the auction comes from the fact that bidders have preferences regarding subsets – bundles – of items. Formally, every bidder i has a valuation function v_i that describes his preferences in monetary terms:

Definition 11.1 A valuation v is a real-valued function that for each subset S of items, $v(S)$ is the value that bidder i obtains if he receives this bundle of items. A valuation must have "free disposal," i.e., be monotone: for $S \subseteq T$ we have that $v(S) \leq v(T)$, and it should be "normalized": $v(\emptyset) = 0$.

The whole point of defining a valuation function is that the value of a bundle of items need not be equal to the sum of the values of the items in it. Specifically for sets S and T, $S \cap T = \emptyset$, we say that S and T are *complements* to each other (in v) if $v(S \cup T) > v(S) + v(T)$, and we say that S and T are *substitutes* if $v(S \cup T) < v(S) + v(T)$.

Note that implicit in this definition are two assumptions about bidder preferences: first, we assume that they are "quasi-linear" in the money; i.e., if bidder i wins bundle S and pays a price of p for it then his utility is $v_i(S) - p$. Second, we assume that there are "no externalities"; i.e., a bidder only cares about the item that he receives and not about how the other items are allocated among the other bidders.

Definition 11.2 An *allocation* of the items among the bidders is $S_1, \ldots, S_n$ where $S_i \cap S_j = \emptyset$ for every $i \neq j$. The *social welfare* obtained by an allocation is $\sum_i v_i(S_i)$. A socially efficient allocation (among bidders with valuations $v_1, \ldots, v_n$) is an allocation with maximum social welfare among all allocations.

In our usual setting the valuation function v_i of bidder i is private information – unknown to the auctioneer or to the other bidders. Our usual goal will be to design a mechanism that will find the socially efficient allocation. What we really desire is a mechanism where this is found in equilibrium, but we will also consider the partial goal of just finding the optimal allocation regardless of strategic behavior of the bidders. One may certainly also attempt designing combinatorial auctions that maximize the auctioneer's revenue, but much less is known about this goal.

There are multiple difficulties that we need to address:

- *Computational complexity:* The allocation problem is computationally hard (NP-complete) even for simple special cases. How do we handle this?
- *Representation and communication:* The valuation functions are exponential size objects since they specify a value for each bundle. How can we even represent them? How do we transfer enough information to the auctioneer so that a reasonable allocation can be found?
- *Strategies:* How can we analyze the strategic behavior of the bidders? Can we design for such strategic behavior?

The combination of these difficulties, and the subtle interplay between them is what gives this problem its generic flavor, in some sense encompassing many of the issues found in algorithmic mechanism design in general.

11.1.2 Some Applications

In this chapter we will undertake a theoretical study and will hardly mention specific applications. More information about various applications can be found in the references mentioned in Section 11.8. Here we will shortly mention a few.

"Spectrum auctions," held worldwide and, in particular, in the united states, have received the most attention. In such auctions a large number of licenses are sold, each license being for the use of a certain band of the electromagnetic spectrum in a certain geographic area. These licenses are needed, for example, by cell-phone companies. To give a concrete example, let us look at the next scheduled auction of the FCC at the time of writing (number 66), scheduled for August 2006. This auction is intended for "advanced wireless services" and includes 1,122 licenses, each covering a 10- or 20-MHz spectrum band (somewhere in the 1.7-GHz or 2.1-GHz frequency range) over a geographic area that contains a population of between 0.5 million to 50 million. The total of the *minimum bids* for all licenses is over 1 billion dollars. Generally speaking, in such auctions bidders desire licenses covering the geographic area that they wish to operate in, with sufficient bandwidth. Most of the spectrum auctions held so far escaped the full complexity of the combinatorial nature of the auction by essentially holding a separate auction for each item (but usually in a clever simultaneous way). In such a format, bidders could not fully express their preferences, thus leading, presumably, to suboptimal allocation of the licenses. In the case of FCC auctions, it has thus been decided to move to a format that will allow "combinatorial bidding," but the details are still under debate.

Another common application area is in transportation. In this setting the auction is often "reversed" – a procurement auction – where the auctioneer needs to *buy* the set of items from many bidding *suppliers*. A common scenario is a company that needs to buy transportation services for a large number of "routes" from various transportation providers (e.g., trucking or shipping companies). For each supplier, the cost of providing a bundle of routes depends on the structure of the bundle as the cost of moving the transportation vehicles between the routes in the bundle needs to be taken into account. Several commercial companies are operating complex combinatorial auctions for transportation services, and commonly report savings of many millions of dollars.

The next application we wish to mention is conceptual, an example demonstrating that various types of problems may be viewed as special cases of combinatorial auctions. Consider a communication network that needs to supply multiple "connection requests" – each requesting a path between two specified nodes in the network, and offering a price for such a path. In the simplest case, each network edge must be fully allocated to one of the requests, so the paths allocated to the requests must be edge-disjoint. Which requests should we fulfill, and which paths should we allocate for it? We can view this as a combinatorial auction: the items sold are the edges of the network. The players are the different requests, and the valuation of a

request gives the offered price for any bundle of edges that contains a path between the required nodes, and 0 for all other bundles.

11.1.3 Structure of This Chapter

We start our treatment of combinatorial auctions, in Section 11.2, by leaving aside the issue of representation and concentrating on bidders with simple "single-minded" valuations. For these bidders we address the twin questions of the computational complexity of allocation and strategic incentive compatibility. The rest of the chapter then addresses general valuations. Section 11.3 lays out mathematical foundations and introduces the notion of Walrasian equilibrium and its relation to the linear programming relaxation of the problem. Section 11.4 describes a first approach for computationally handling general valuations: representing them in various "bidding languages." Section 11.5 describes a second approach, that of using iterative auctions which repeatedly query bidders about their valuations. In Section 11.6 we show the limitations of the second approach, pointing out an underlying communication bottleneck. Section 11.7 studies a natural widely used family of iterative auctions – those with ascending prices. Bibliographic notes appear in Section 11.8, followed by a collection of exercises.

11.2 The Single-Minded Case

This section focuses on the twin goals of computational complexity and strategic behavior, while leaving out completely the third issue of the representational complexity of the valuation functions. For this, we restrict ourselves to players with very simple valuation functions which we call "single-minded bidders." Such bidders are interested only in a single specified bundle of items, and get a specified scalar value if they get this whole bundle (or any superset) and get zero value for any other bundle.

Definition 11.3 A valuation v is called *single minded* if there exists a bundle of items S^* and a value $v^* \in \Re^+$ such that $v(S) = v^*$ for all $S \supseteq S^*$, and $v(S) = 0$ for all other S. A single-minded bid is the pair (S^*, v^*).

Single-minded valuations are thus very simply represented. The rest of this section assumes as common knowledge that all bidders are single minded.

11.2.1 Computational Complexity of Allocation

Let us first consider just the algorithmic allocation problem among single-minded bidders. Recall that in general, an allocation gives disjoint sets of items S_i to each bidder i, and aims to maximize the social welfare $\sum_i v_i(S_i)$. In the case of single-minded bidders whose bids are given by (S_i^*, v_i^*), it is clear that an optimal allocation can allocate to every bidder either exactly the bundle he desires $S_i = S_i^*$ or nothing at all $S_i = \emptyset$. The algorithmic allocation problem among such bidders is thus given by the following definition.

Definition 11.4 The allocation problem among single-minded bidders is the following:
INPUT: (S_i^*, v_i^*) for each bidder $i = 1, \ldots, n$.
OUTPUT: A subset of winning bids $W \subseteq \{1, \ldots, n\}$ such that for every $i \neq j \in W$, $S_i^* \cap S_j^* = \emptyset$ (i.e., the winners are compatible with each other) with maximum social welfare $\sum_{i \in W} v_i^*$.

This problem is a "weighted-packing" problem and is NP-complete, which we will show by reduction from the INDEPENDENT-SET problem.

Proposition 11.5 *The allocation problem among single-minded bidders is NP-hard. More precisely, the decision problem of whether the optimal allocation has social welfare of at least k (where k is an additional part of the input) is NP-complete.*

PROOF We will make a reduction from the NP-complete "INDEPENDENT-SET" problem: given an undirected graph $G = (V, E)$ and a number k, does G have an independent set of size k? An independent set is a subset of the vertices that have no edge between any two of them. Given such an INDEPENDENT-SET instance, we will build an allocation problem from it as follows:

- The set of items will be E, the set of edges in the graph.
- We will have a player for each vertex in the graph. For vertex $i \in V$ we will have the desired bundle of i be the set of adjacent edges $S_i^* = \{e \in E | i \in e\}$, and the value be $v_i^* = 1$.

Now notice that a set W of winners in the allocation problem satisfies $S_i^* \cap S_j^* = \emptyset$ for every $i \neq j \in W$ if and only if the set of vertices corresponding to W is an independent set in the original graph G. The social welfare obtained by W is exactly the size of this set, i.e., the size of the independent set. It follows that an independent set of size at least k exists if and only if the social welfare of the optimal allocation is at least k. This concludes the NP-hardness proof. The fact that the problem (of whether the optimal allocation has social welfare at least k) is in NP is trivial as the optimal allocation can be guessed and then the social welfare can be calculated routinely. □

As usual when a computational problem is shown to be NP-complete, there are three approaches for the next step: approximation, special cases, and heuristics. We will discuss each in turn.

First, we may attempt finding an allocation that is approximately optimal. Formally, we say that an allocation $S_1, \ldots, S_n$ is a c-approximation of the optimal one if for every other allocation $T_1, \ldots, T_n$ (and specifically for the socially optimal one), we have that $\frac{\sum_i v_i(T_i)}{\sum_i v_i(S_i)} \leq c$. Perhaps a computationally efficient algorithm will always be able to find an approximately optimal allocation? Unfortunately, the NP-completeness reduction above also shows that this will not be possible. Not only is it known that the finding the maximum independent set is NP-complete, but it is known that approximating it to within a factor of $n^{1-\epsilon}$ (for any fixed $\epsilon > 0$) is NP-complete. Since in our reduction the

social welfare was exactly equal to the independent-set size, we get the same hardness here. Often this is stated as a function of the number of items m rather than the number of players n. Since $m \leq n^2$ (m is the number of edges, n is the number of vertices), we get:

Proposition 11.6 *Approximating the optimal allocation among single-minded bidders to within a factor better than $m^{1/2-\epsilon}$ is NP-hard.*

As we will see in the next subsection, this level of approximation can be reached in polynomial time, even in an incentive-compatible way (which is the topic of the next subsection).

Second, we can focus on special cases that can be solved efficiently. Several such cases are known. The first one is when each bidder desires a bundle of at most two items $|S_i^*| \leq 2$. This case is seen to be an instance of the weighted matching problem (in general nonbipartite graphs) which is known to be efficiently solvable. The second case is the "linear order" case. Assume that the items are arranged in a linear order and each desired bundle is for a continuous segment of items, i.e., each $S_i^* = \{j^i, j^i + 1, \ldots, k^i\}$ for some $1 \leq j^i \leq k^i \leq m$ (think of the items as lots along the sea shore, and assume that each bidder wants a connected strip of seashore). It turns out that this case can be solved efficiently using dynamic programming, which we leave as an exercise to the reader (see Exercise 11.1).

Third, an NP-completeness result only says that one cannot write an algorithm that is guaranteed to run in polynomial time and obtain optimal outputs on *all input instances*. It may be possible to have algorithms that run reasonably fast and produce optimal (or near-optimal) results on most natural input instances. Indeed, it seems to be the case here: the allocation problem can be stated as an "integer programming" problem, and then the large number of known heuristics for solving integer programs can be applied. In particular, many of these heuristics rely on the linear programming relaxation of the problem, which we will study in Section 11.3 in a general setting. It is probably safe to say that most allocation problems with up to hundreds of items can be practically solved optimally, and that even problems with thousands or tens of thousands of items can be practically approximately solved quite well.

11.2.2 An Incentive-Compatible Approximation Mechanism

After dealing with the purely algorithmic aspect in the last subsection, we now return to handling also strategic issues. Again, we still avoid all representation difficulties, i.e., focusing on single-minded bidders. That is, we now wish to take into account the fact that the true bids are private information of the players, and not simply available to the algorithm. We still would like to optimize the social welfare as much as possible. The approach we take is the standard one of mechanism design: *incentive compatibility*. We refer the reader to Chapter 9 for background, but in general what we desire is an allocation algorithm and payment functions such that each player always prefers reporting his private information truthfully to the auctioneer rather than any potential lie. This would ensure that the allocation algorithm at least works with the true information. We also wish everything to be efficiently computable, of course.

Definition 11.7 Let V_{sm} denote the set of all single-minded bids on m items, and let A be the set of all allocations of the m items between n players. A mechanism for single-minded bidders is composed of an allocation mechanism $f : (V_{sm})^n \to A$ and payment functions $p_i : (V_{sm})^n \to \Re$ for $i = 1, \ldots, n$. The mechanism is computationally efficient if f and all p_i can be computed in polynomial time. The mechanism is incentive compatible (in dominant strategies) if for every i, and every $v_1, ..., v_n, v_i' \in V_{sm}$, we have that $v_i(a) - p_i(v_i, v_{-i}) \geq v_i(a') - p_i(v_i', v_{-i})$, where $a = f(v_i, v_{-i})$, $a' = f(v_i', v_{-i})$ and $v_i(a) = v_i$ if i wins in a and zero otherwise.

The main difficulty here is the clash between the requirements of incentive compatibility and that of computational efficiency. If we leave aside the requirement of computational efficiency then the solution to our problem is simple: take the socially efficient allocation and let the payments be the VCG payments defined in Chapter 9. These payments essentially charge each bidder his "externality": the amount by which his allocated bundle reduced the total reported value of the bundles allocated to others. As shown in Chapter 9, this would be incentive compatible, and would give the exactly optimal allocation. However, as shown above, exact optimization of the social welfare is computationally intractable. Thus, when we return to the requirement of computational efficiency, exact optimization is impossible. Now, one may attempt using "VCG-like" mechanisms: take the best approximation algorithm you can find for the problem – which can have a theoretical guarantee of no better than $O(\sqrt{m})$ approximation but may be practically much better – and attempt using the same idea of charging each bidder his externality according to the allocation algorithm used. Unfortunately, this would not be incentive compatible! VCG-like payments lead to incentive compatibility if but only if the social welfare is exactly optimized by the allocation rule (at least over some subrange of allocations).

We thus need to find another type of mechanisms – non-VCG. While in general settings almost no incentive compatible mechanisms are known beyond VCG, our single-minded setting is "almost single-dimensional" – in the since that the private values are composed of a single scalar and the desired bundle – and for such settings this is easier. Indeed, the mechanism in Figure 11.1 is computationally efficient, incentive compatible, and provides a $\sqrt{m}$ approximation guarantee, as good as theoretically possible in polynomial time.

This mechanism greedily takes winners in an order determined by the value of the expression $v_1^*/\sqrt{|S_1^*|}$. This expression was taken as to optimize the approximation ratio obtained theoretically, but as we will see, the incentive compatibility result would apply to any other expression that is monotone increasing in v_i^* and decreasing in $|S_i^*|$. The intuition behind the choice of j for defining the payments is that this is the bidder who lost exactly because of i – if Bidder i had not participated in the auction, Bidder j would have won.

Theorem 11.8 *The greedy mechanism is efficiently computable, incentive compatible, and produces a $\sqrt{m}$ approximation of the optimal social welfare.*

The Greedy Mechanism for Single-Minded Bidders:

Initialization:

- Reorder the bids such that $v_1^*/\sqrt{|S_1^*|} \geq v_2^*/\sqrt{|S_2^*|} \geq ... \geq v_n^*/\sqrt{|S_n^*|}$.
- $W \leftarrow \emptyset$.

For i = 1...n do: if $S_i^* \cap \left(\bigcup_{j \in W} S_j^*\right) = \emptyset$ then $W \leftarrow W \cup \{i\}$.

Output:

Allocation: The set of winners is W.

Payments: For each $i \in W$, $p_i = v_j^*/\sqrt{|S_j^*|/|S_i^*|}$, where j is the smallest index such that $S_i^* \cap S_j^* \neq \emptyset$, and for all $k < j, k \neq i$, $S_k^* \cap S_j^* = \emptyset$ (if no such j exists then $p_i = 0$).

Figure 11.1. The mechanism achieves a $\sqrt{m}$ approximation for combinatorial auctions with single-minded bidders.

Computational efficiency is obvious; we will show incentive compatibility and the approximation performance in two separate lemmas. The incentive compatibility of this mechanism follows directly from the following lemma.

Lemma 11.9 *A mechanism for single-minded bidders in which losers pay 0 is incentive compatible if and only if it satisfies the following two conditions:*

(i) **Monotonicity:** *A bidder who wins with bid (S_i^*, v_i^*) keeps winning for any $v_i' > v_i^*$ and for any $S_i' \subset S_i^*$ (for any fixed settings of the other bids).*

(ii) **Critical Payment:** *A bidder who wins pays the minimum value needed for winning: the infimum of all values v_i' such that (S_i^*, v_i') still wins.*

Before we prove the lemma – or actually just the side that we need – let us just verify that our mechanism satisfies these two properties. Monotonicity is implied since increasing v_i^* or decreasing S_i^* can only move bidder i up in the greedy order, making it easier to win. The critical payment condition is met since notice that i wins as long as he appears in the greedy order before j. The payment computed is exactly the value at which the transition between i being before and after j in the greedy order happens.

Note that this characterization is different from the characterization given in Chapter 9 for general single-parameter agents, since single-minded bidders are not considered to have a single parameter, as their private data consists of both their value and their desired bundle.

PROOF We first observe that under the given conditions, a truthful bidder will never receive negative utility: his utility is zero while losing (losers pay zero), and for winning, his value must be at least the critical value, which exactly equals his payment. We will now show that a bidder can never improve his utility by reporting some bid (S', v') instead of his true values (S, v). If (S', v') is a losing bid or if S' does not contain S, then clearly reporting (S, v) can only help. Therefore we will assume that (S', v') is a winning bid and that $S' \supseteq S$.

We next show that the bidder will never be worse off by reporting (S, v') rather than (S', v'). Denote the bidder's payment for the bid (S', v') by p', and for the bid (S, v') by p. For every $x < p$, bidding (S, x) will lose since p is a critical value. By monotonicity, (S', x) will also be a losing bid for every $x < p$, and therefore the critical value p' is at least p. It follows that by bidding (S, v') instead of (S', v') the bidder still wins and his payment will not increase.

It is left to show that bidding (S, v) is no worse than the winning bid (S, v'): Assume first that (S, v) is a winning bid with a payment (critical value) $\widetilde{p}$. As long as v' is greater than $\widetilde{p}$, the bidder still wins with the same payment, thus misreporting his value would not be beneficial. When $v' < \widetilde{p}$ the bidder will lose, gaining zero utility, and he will not be better off.

If (S, v) is a losing bid, v must be smaller than the corresponding critical value, so the payment for any winning bid (S, v') will be greater than v, making this deviation nonprofitable. □

The approximation guarantee is ensured by the following lemma.

Lemma 11.10 *Let OPT be an allocation (i.e., set of winners) with maximum value of $\sum_{i \in OPT} v_i^*$, and let W be the output of the algorithm, then $\sum_{i \in OPT} v_i^* \leq \sqrt{m} \sum_{i \in W} v_i^*$.*

PROOF For each $i \in W$ let $OPT_i = \{j \in OPT, j \geq i \mid S_i^* \cap S_j^* \neq \emptyset\}$ be the set of elements in OPT that did not enter W because of i (in addition to i itself). Clearly $OPT \subseteq \bigcup_{i \in W} OPT_i$ and thus the lemma will follow once we prove the claim that for every $i \in W$, $\sum_{j \in OPT_i} v_j^* \leq \sqrt{m} v_i^*$.

Note that every $j \in OPT_i$ appeared after i in the greedy order and thus $v_j^* \leq \frac{v_i^* \sqrt{|S_j^*|}}{\sqrt{|S_i^*|}}$.Summing over all $j \in OPT_i$, we can now estimate

$$\sum_{j \in OPT_i} v_j^* \leq \frac{v_i^*}{\sqrt{|S_i^*|}} \sum_{j \in OPT_i} \sqrt{|S_j^*|}. \tag{11.1}$$

Using the Cauchy–Schwarz inequality, we can bound

$$\sum_{j \in OPT_i} \sqrt{|S_j^*|} \leq \sqrt{|OPT_i|} \sqrt{\sum_{j \in OPT_i} |S_j|}. \tag{11.2}$$

Every S_j^* for $j \in OPT_i$ intersects S_i^*. Since OPT is an allocation, these intersections must all be disjoint, and thus $|OPT_i| \leq |S_i^*|$. Since OPT is an allocation $\sum_{j \in OPT_i} |S_j| \leq m$. We thus get $\sum_{j \in OPT_i} \sqrt{|S_j^*|} \leq \sqrt{|S_i^*|}\sqrt{m}$, and plugging into Inequality 11.1 gives the claim $\sum_{j \in OPT_i} v_j^* \leq \sqrt{m} v_i^*$. □

11.3 Walrasian Equilibrium and the LP Relaxation

In this section we return to discuss combinatorial auctions with general valuations, and we will study the linear-programming relaxation of the winner-determination problem in such auctions. We will also define the economic notion of a competitive equilibrium

with item prices (or "Walrasian equilibrium"). Although these notions appear to be independent at a first glance, we will describe a strong connection between them. In particular, we will prove that the existence of a Walrasian equilibrium is a sufficient and necessary condition for having an integer optimal solution for the linear programming relaxation (i.e., no integrality gap). One immediate conclusion is that in environments where Walrasian Equilibria exist, the efficient allocation can be computed in polynomial time.

11.3.1 The Linear Programming Relaxation and Its Dual

The winner determination problem in combinatorial auctions can be formulated by an integer program. We present the linear programming relaxation of this integer program, and denote it by LPR (in the integer program Constraint (11.6) would be replaced with "$x_{i,S} \in \{0, 1\}$").

The Linear Programming Relaxation (LPR):

$$\textbf{Maximize} \quad \sum_{i \in N, S \subseteq M} x_{i,S}\, v_i(S) \tag{11.3}$$

$$\textbf{s.t.} \quad \sum_{i \in N,\ S | j \in S} x_{i,S} \leq 1 \qquad \forall j \in M \tag{11.4}$$

$$\sum_{S \subseteq M} x_{i,S} \leq 1 \qquad \forall i \in N \tag{11.5}$$

$$x_{i,S} \geq 0 \qquad \forall i \in N, S \subseteq M \tag{11.6}$$

In the integer program, each variable $x_{i,S}$ equals 1 if bidder i receives the bundle S, and zero otherwise. The objective function is therefore maximizing social welfare. Condition 11.4 ensures that each item is allocated to at most one bidder, and Condition 11.5 implies that each player is allocated at most one bundle. Solutions to the linear program can be intuitively viewed as fractional allocations: allocations that would be allowed if items were divisible. While the LP has exponentially (in m) many variables, it still has algorithmic implications. For example, in the case of single-minded bidders only a single variable X_{i,S_i^*} for each bidder i is required, enabling direct efficient solution of the LP. In Section 11.5.2 we will see that, assuming reasonable access to the valuations, the general LP can be solved efficiently as well.

We will also consider the dual linear program.

The Dual Linear Programming Relaxation (DLPR)

$$\textbf{Minimize} \quad \sum_{i \in N} u_i + \sum_{j \in M} p_j \tag{11.7}$$

$$\textbf{s.t.} \quad u_i + \sum_{j \in S} p_j \geq v_i(S) \qquad \forall i \in N, \quad S \subseteq M \tag{11.8}$$

$$u_i \geq 0, \quad p_j \geq 0 \qquad \forall i \in N, \ j \in M \tag{11.9}$$

The usage of the notations p_j and u_i is intentional, since we will later see that at the optimal solution, these dual variables can be interpreted as the prices of the items and the utilities of the bidders.

11.3.2 Walrasian Equilibrium

A fundamental notion in economic theory is the notion of a competitive equilibrium: a set of prices where the market clears, i.e., the demand equals the supply. We will now formalize this concept, that will be generalized later in Section 11.7.

Given a set of prices, the *demand* of each bidder is the bundle that maximizes her utility. (There may be more than one such bundle, in which case each of them is called *a* demand.) In this section we will consider a linear pricing rule, where a price per each *item* is available, and the price of each bundle is the sum of the prices of the items in this bundle.

Definition 11.11 For a given bidder valuation v_i and given item prices $p_1, \ldots, p_m$, a bundle T is called a *demand* of bidder i if for every other bundle $S \subseteq M$ we have that $v_i(S) - \sum_{j \in S} p_j \leq v_i(T) - \sum_{j \in T} p_j$.

A Walrasian equilibrium[1] is a set of "market-clearing" prices where every bidder receives a bundle in his demand set, and unallocated items have zero prices.

Definition 11.12 A set of nonnegative prices $p_1^*, \ldots, p_m^*$ and an allocation $S_1^*, \ldots, S_m^*$ of the items is a *Walrasian equilibrium* if for every player i, S_i^* is a demand of bidder i at prices $p_1^*, \ldots, p_m^*$ and for any item j that is not allocated (i.e., $j \notin \cup_{i=1}^n S_i^*$) we have $p_j^* = 0$.

The following result shows that Walrasian equilibria, if they exist, are economically efficient; i.e., they necessarily obtain the optimal welfare. This is a variant of the classic economic result known as the *First Welfare Theorem* but for environments with indivisible items. Here we actually prove a stronger statement: the welfare in a Walrasian equilibrium is maximal even if the items were divisible. In particular, if a Walrasian equilibrium exists, then the optimal solution to the linear program relaxation will be integral.

Theorem 11.13 (The First Welfare Theorem) *Let $p_1^*, \ldots, p_m^*$ and $S_1^*, \ldots, S_n^*$ be a Walrasian equilibrium, then the allocation $S_1^*, \ldots, S_n^*$ maximizes social welfare. Moreover, it even maximizes social welfare over all fractional allocations, i.e., let $\{X_{i,S}^*\}_{i,S}$ be a feasible solution to the linear programming relaxation. Then, $\sum_{i=1}^n v_i(S_i^*) \geq \sum_{i \in N,\, S \subseteq M} X_{i,S}^* v_i(S)$.*

[1] Walras was an economist who published in the 19th century one of the first comprehensive mathematical analyses of general equilibria in markets.

PROOF In a Walrasian equilibrium, each bidder receives his demand. Therefore, for every bidder i and every bundle S, we have $v_i(S_i^*) - \sum_{j \in S_i^*} p_j^* \geq v_i(S) - \sum_{j \in S} p_j^*$. Since the fractional solution is feasible to the LPR, we have that for every bidder i, $\sum_S X_{i,S}^* \leq 1$ (Constraint 11.5), and therefore

$$v_i(S_i^*) - \sum_{j \in S_i^*} p_j^* \geq \sum_{S \subseteq M} X_{i,S}^* \left(v_i(S) - \sum_{j \in S} p_j^* \right). \tag{11.10}$$

The theorem will follow from summing Inequality 11.10 over all bidders, and showing that $\sum_{i \in N} \sum_{j \in S_i^*} p_j^* \geq \sum_{i \in N,\, S \subseteq M} X_{i,S}^* \sum_{j \in S} p_j^*$. Indeed, the left-hand side equals $\sum_{j=1}^m p_j^*$ since $S_1^*, \ldots, S_n^*$ is an allocation and the prices of unallocated items in a Walrasian equilibrium are zero, and the right-hand side is at most $\sum_{j=1}^m p_j^*$, since the coefficient of every price p_j^* is at most 1 (by Constraint 11.4 in the LPR). □

Following is a simple class of valuations for which no Walrasian equilibrium exist.

Example 11.14 Consider two players, Alice and Bob, and two items $\{a, b\}$. Alice has a value of 2 for every nonempty set of items, and Bob has a value of 3 for the whole bundle $\{a, b\}$, and 0 for any of the singletons. The optimal allocation will clearly allocate both items to Bob. Therefore, Alice must demand the empty set in any Walrasian equilibrium. Both prices will be at least 2; otherwise, Alice will demand a singleton. Hence, the price of the whole bundle will be at least 4, Bob will not demand this bundle, and consequently, no Walrasian equilibrium exists for these players.

To complete the picture, the next theorem shows that the existence of an integral optimum to the linear programming relaxation is also a sufficient condition for the existence of a Walrasian equilibrium. This is a variant of a classic theorem, known as "The Second Welfare Theorem," that provided sufficient conditions for the existence of Walrasian equilibria in economies with divisible commodities.

Theorem 11.15 (The Second Welfare Theorem) *If an integral optimal solution exists for LPR, then a Walrasian equilibrium whose allocation is the given solution also exists.*

PROOF An optimal integral solution for LPR defines a feasible efficient allocation $S_1^*, \ldots, S_n^*$. Consider also an optimal solution $p_1^*, \ldots, p_n^*, u_1^*, \ldots, u_n^*$ to DLPR. We will show that $S_1^*, \ldots, S_n^*, p_1^*, \ldots, p_n^*$ is a Walrasian equilibrium.

Complementary-slackness conditions are necessary and sufficient conditions for the optimality of solutions to the primal linear program and its dual. Because of the complementary-slackness conditions, for every player i for which $x_{i,S_i^*} > 0$ (i.e., $x_{i,S_i^*} = 1$), we have that Constraint (11.8) is binding for the optimal dual solution, i.e.,

$$u_i^* = v_i(S_i^*) - \sum_{j \in S_i^*} p_i^*$$

Constraint 11.8 thus also shows that for any other bundle S we get

$$v_i(S_i^*) - \sum_{j \in S_i^*} p_i^* \geq v_i(S) - \sum_{j \in S} p_i^*$$

Finally, the complementary-slackness conditions also imply that for every item j for which Constraint (11.4) is strict, i.e., $\sum_{i \in N,\ S | j \in S} x_{i,S} < 1$ – which for integral solutions means that item j is unallocated – then necessarily $p_j^* = 0$. □

The two welfare theorems show that the existence of a Walrasian equilibrium is equivalent to having a zero integrality gap:

Corollary 11.16 *A Walrasian equilibrium exists in a combinatorial-auction environment if and only if the corresponding linear programming relaxation admits an integral optimal solution.*

11.4 Bidding Languages

This section concerns the issue of the *representation* of bids in combinatorial auctions. Namely, we are looking for representations of valuations that will allow bidders to simply encode their valuation and send it to the auctioneer. The auctioneer must then take the valuations (bids) received from all bidders and determine the allocation. Following sections will consider indirect, iterative ways of transferring information to the auctioneer.

Specifying a valuation in a combinatorial auction of m items requires providing a value for each of the possible $2^m - 1$ nonempty subsets. A naive representation would thus require $2^m - 1$ real numbers to represent each possible valuation. It is clear that this would be completely impractical for more than about two or three dozen items. The computational complexity can be effectively handled for much larger auctions, and thus the representation problem seems to be the bottleneck in practice.

We will thus be looking for languages that allow succinct representations of valuations. We will call these *bidding languages* reflecting their intended usage rather than the more precise "valuations languages." From the outset it is clear that due to information theoretic reasons it will never be possible to encode *all* possible valuations succinctly. Our interest would thus be in succinctly representing interesting or important ones.

When attempting to choose or design a bidding language, we are faced with the same types of trade-offs common to all language design tasks: *expressiveness vs. simplicity*. On one hand, we would like our language to express succinctly as many "naturally occurring" valuations as possible. On the other hand, we would like it to be as simple as possible, both for humans to express and for programs to work with. A well-chosen bidding language should aim to strike a good balance between these two goals.

The bottom line of this section will be the identification of a simple langauge that is rather powerful and yet as easily handled by allocation algorithms as are the single minded bids studied in Section 11.2.

11.4.1 Elements of Representation: Atoms, OR, and XOR

The common bidding languages construct their bids from *combinations* of simple *atomic bids*. The usual atoms in such schemes are the single-minded bids addressed in Section 11.2: (S, p) meaning an offer of p monetary units for the bundle S of items. Formally, the valuation represented by (S, p) is one where $v(T) = p$ for every $T \supseteq S$, and $v(T) = 0$ for all other T.

Intuitively, bids can be combined by simply offering them together. Still informally, there are two possible semantics for an offer of several bids. One considers the bids as totally independent, allowing any subset of them to be fulfilled, and the other considers them to be mutually exclusive and allows only one of them to be fulfilled. The first semantics is called an OR bid, and the second is called (somewhat misleadingly) a XOR bid.

Take, for example, the valuations represented by "$(\{a, b\}, 3)\ XOR\ (\{c, d\}, 5)$" and "$(\{a, b\}, 3)\ OR\ (\{c, d\}, 5)$." Each of them values the bundle $\{a, c\}$ at 0 (since no atomic bid is satisfied) and values the bundle $\{a, b\}$ at 3. The difference is in the bundle $\{a, b, c, d\}$, which is valued at 5 by the XOR bid (according to the best atomic bid satisfied), but is valued at 8 by the OR bid. For another example, look at the bid "$(\{a, b\}, 3)\ OR\ (\{a, c\}, 5)$." Here, the bundle $\{a, b, c\}$ is valued at 5 since both atomic bids cannot be satisfied together.

More formally, both OR and XOR bids are composed of a collection of pairs (S_i, p_i), where each S_i is a subset of the items, and p_i is the maximum price that he is willing to pay for that subset. For the valuation $v = (S_1, p_1)\ XOR, \ldots, XOR\ (S_k, p_k)$, the value of $v(S)$ is defined to be $max_{i|S_i \subseteq S}\ p_i$. For the valuation $v = (S_1, p_1)\ OR, \ldots, OR\ (S_k, p_k)$, one must be a little careful and the value of $v(S)$ is defined to be the maximum over all possible "valid collections" W, of the value of $\sum_{i \in W} p_i$, where W is a valid collection of pairs if for all $i \neq j \in W$, $S_i \cap S_j = \emptyset$.

It is not difficult to see that XOR bids can represent every valuation v: just XOR, the atomic bids $(S, v(S))$ for all bundles S. On the other hand, OR bids can represent only *superadditive* bids (for any two disjoint sets S, T, $v(S \cup T) \geq v(S) + v(T)$), since the atoms giving the value $v(S)$ are disjoint from those giving the value $v(T)$, and they will be added together for $v(S \cup T)$. It is not difficult to see that all superadditive valuations can indeed be represented by OR bids by ORing the atomic bids $(S, v(S))$ for all bundles S.

We will be more interested in the *size* of the representation, defined to be simply the number of atomic bids in it. The following basic types of valuations are good examples for the power and limitations of these two bidding languages.

Definition 11.17 A valuation is called *additive* if $v(S) = \sum_{j \in S}\ v(\{j\})$ for all S. A valuation is called *unit demand* if $v(S) = max_{j \in S}\ v(\{j\})$ for all S.

An additive valuation is directly represented by an OR bid:

$$(\{1\}, p_1)\ OR\ (\{2\}, p_2)\ OR\ \cdots\ OR\ (\{m\}, p_m)$$

while a unit-demand valuation is directly represented by an XOR bid:

$$(\{1\}, p_1)\ XOR\ (\{2\}, p_2)\ XOR\ \cdots\ XOR\ (\{m\}, p_m)$$

where for each item j, $p_j = v(\{j\})$. Additive valuations can be represented by XOR bids, but this may take exponential size: atomic bids for all $2^m - 1$ possible bundles will be needed whenever $p_j > 0$ for all j. (Since an atomic bid is required for every bundle S with $v(S)$ strictly larger than that of all its strict subsets, which is the case here for all S.) On the other hand, nontrivial unit-demand valuations are never superadditive and thus cannot be represented at all by OR bids.

11.4.2 Combinations of OR and XOR

While both the OR and XOR bidding languages are appealing in their simplicity, none of them are expressive enough to succinctly represent many desirable simple valuations. A natural attempt is to combine the power of OR bids and XOR bids. The most general way to allow this general form of combinations is to define OR and XOR as operations on valuations.

Definition 11.18 Let v and u be valuations, then $(v\ XOR\ u)$ and $(v\ OR\ u)$ are valuations and are defined as follows:

- $(v\ XOR\ u)(S) = \max(v(S), u(S))$.
- $(v\ OR\ u)(S) = \max_{R,T \subseteq S,\ R \cap T = \emptyset}\ v(R) + u(T)$

Thus a general "OR/XOR formula" bid will be given by an arbitrary expression involving the OR and XOR operations over atomic bids. For instance, the bid $((\{a, b\}, 3)\ XOR\ (\{c\}, 2))\ OR\ (\{d\}, 5)$ values the bundle $\{a, b, c\}$ at 3, but the bundle $\{a, b, d\}$ at 8. The following example demonstrates the added power we can get from such combinations just using the restricted structure of an OR of XORs of atomic bids.

Definition 11.19 A valuation is called *symmetric* if $v(S)$ depends only on $|S|$. A symmetric valuation is called *downward sloping* if it can be represented as $v(S) = \sum_{j=1..|S|} p_j$, with $p_1 \geq p_2 \geq \cdots \geq p_m \geq 0$.

It is easy to verify that every downward sloping valuations with $p_1 > p_2 > \cdots > p_m > 0$ requires XOR bids of size $2^m - 1$, and cannot be represented at all by OR bids.

Lemma 11.20 *OR-of-XORs bids can express any downward sloping symmetric valuation on m items in size m^2.*

PROOF For each $j = 1, \ldots, m$ we will have a clause that offers p_j for any single item. Such a clause is a simple XOR-bid, and the m different clauses are all connected by an OR. Since the p_j's are decreasing, we are assured that the first allocated item will be taken from the first clause, the second item from the second clause, etc. □

11.4.3 Dummy Items

General OR/XOR formulae seem very complicated and dealing with them algorithmically would appear to be quite difficult. Luckily, this is not the case and a generalization of the langauge makes things simple again. The main idea is to allow XORs to be represented by ORs. This is done by allowing the bidders to introduce *dummy items* into the bids. These items will have no intrinsic value to any of the participants, but they will be indirectly used to express XOR constraints. The idea is that an XOR bid $(S_1, p_1)\ XOR\ (S_2, p_2)$ can be represented as $(S_1 \cup \{d\}, p_1)\ OR\ (S_2 \cup \{d\}, p_2)$, where d is a dummy item.

Formally, we let each bidder i have its own set of dummy items D_i, which only he can bid on. An OR* bid by bidder i is an OR bid on the augmented set of items $M \cup D_i$. The value that an OR* bid gives to a bundle $S \subseteq M$ is the value given by the OR bid to $S \cup D_i$. Thus, for example, for the set of items $M = \{a, b, c\}$, the OR* bid $(\{a, d\}, 1)\ OR\ (\{b, d\}, 1)\ OR\ (\{c\}, 1)$, where d is a dummy item, is equivalent to $((\{a\}, 1)\ XOR\ (\{b\}, 1))\ OR\ (\{c\}, 1)$.

An equivalent but more appealing "user interface" is to let bidders report a set of atomic bids together with "constraints" that signify which bids are mutually exclusive. Each constraint can then be converted into a dummy item that is added to the conflicting atomic bids. Despite its apparent simplicity, this language can simulate general OR/XOR formulae.

Theorem 11.21 *Any valuation that can be represented by OR/XOR formula of size s can be represented by OR* bids of size s, using at most s^2 dummy items.*

PROOF We prove by induction on the formula structure that a formula of size s can be represented by an OR* bid with s atomic bids. We then show that each atomic bid in the final resulting OR* bid can be modified as to not to include more than s dummy items in it.

Induction: The basis of the induction is an atomic bid, which is clearly an OR* bid with a single atomic bid. The induction step requires handling the two separate cases: OR and XOR. To represent the OR of several OR* bids as a single OR* bid, we simply merge the set of clauses of the different OR* bids. To represent the XOR of several OR* bids as a single OR* bid, we introduce a new dummy item x_{ST} for each pair of atomic bids (S, v) and (T, v') that are in two different original OR* bids. For each bid (S, v) in any of the original OR* bids, we add to the generated OR* bid an atomic bid $(S \cup \{x_{ST} | T\}, v)$, where T ranges over all atomic bids in all of the other original OR* bids.

It is clear that the inductive construction constructs an OR* bid with exactly s clauses in it, where s is the number of clauses in the original OR/XOR formula. The number of dummy items in it, however, may be large. However, we can remove most of these dummy items. One can see that the only significance of a dummy item in an OR* bid is to disallow some two (or more) atomic bids to be taken concurrently. Thus we may replace all the existing dummy items with at most $\binom{s}{2}$ new dummy items, one for each pair of atomic bids that cannot be taken

together (according to the current set of dummy items). This dummy item will be added to both of the atomic bids in this pair. □

This simulation can be directly turned into a "compiler" that translates OR/XOR formulae into OR* bids. This has an extremely appealing implication for allocation algorithms: to any winner determination (allocation) algorithm, an OR* bid looks just like a regular OR-bid on a larger set of items. But an OR bid looks to an allocation algorithm just like a collection of atomic bids from different players. It follows that any allocation algorithm that can handle single-minded bids (i.e., atomic bids) can immediately also handle general valuations represented as OR* bids or as general OR/XOR formulae. In particular, the various heuristics mentioned in Section 11.2 can all be applied for general valuations represented in these languages.

11.5 Iterative Auctions: The Query Model

The last section presented ways of encoding valuations in bidding languages as to enable the bidders to directly send their valuation to the auctioneer. In this section we consider indirect ways of sending information about the valuation: *iterative auctions*. In these, the auction protocol repeatedly interacts with the different bidders, aiming to adaptively elicit enough information about the bidders' preferences as to be able to find a good (optimal or close to optimal) allocation. The idea is that the adaptivity of the interaction with the bidders may allow pinpointing the information that is relevant to the current auction and not requiring full disclosure of bidders' valuations. This may not only reduce the amount of information transferred and all associated complexities but also preserve some privacy about the valuations, only disclosing the information that is really required. In addition, in many real-life settings, bidders may need to exert efforts even for determining their own valuation (like collecting data, hiring consultants, etc.); such iterative mechanisms may assist the bidders with realizing their valuations by guiding their attention only to the data that is relevant to the mechanism.

Such iterative auctions can be modeled by considering the bidders as "black-boxes," represented by oracles, where the auctioneer repeatedly queries these oracles. In such models, we should specify the types of queries that are allowed by the auctioneer. These oracles may not be truthful, of course, and we will discuss the incentive issues in the final part of this section (see also Chapter 12). The auctioneer would be required to be computationally efficient in two senses: the number of queries made to the bidders and the internal computations. Efficiency would mean polynomial running time in m (the number of items) even though each valuation is represented by 2^m numbers. The running time should also be polynomial in n (the number of bidders) and in the number of bits of precision of the real numbers involved in the valuations.

11.5.1 Types of Queries

Our first step is to define the types of queries that we allow our auctioneer to make to the bidders. Probably the most straightforward query one could imagine is where a bidder reports his value for a specific bundle.

Value query: *The auctioneer presents a bundle S, the bidder reports his value $v(S)$ for this bundle.*

It turns out that value queries are pretty weak and are not expressive enough in many settings. Another natural and widely used type of queries is the *demand query*, in which a set of prices is presented to the bidder, and the bidder responds with his most valuable bundle under the published prices.

***Demand query** (with item prices[2])*: *The auctioneer presents a vector of item prices $p_1, \ldots, p_m$; the bidder reports a demand bundle under these prices, i.e., some set S that maximizes $v(S) - \sum_{i \in S} p_i$.*

How difficult it is for a bidder to answer such a demand query or a value query depends on his internal representation of his valuation. For some internal representations this may be computationally intractable, while for others it may be computationally trivial. It does seem though that in many realistic situations the bidders will not really have an explicit internal representation, but rather "know" their valuation only in the sense of being able to answer such queries.

The first observation that we should make is that demand queries are strictly more powerful than value queries.

Lemma 11.22 *A value query may be simulated by mt demand queries, where t is the number of bits of precision in the representation of a bundle's value.*

PROOF We first show how to answer "marginal value" queries using demand queries: given a bundle S and an item $j \notin S$, compute the marginal value of j relative to S: $v(S \cup \{j\}) - v(S)$ (the items are denoted, w.l.o.g., by $1, \ldots, m$). For all $i \in S$ we set $p_i = 0$, for all $i \notin S \cup \{j\}$, we set $p_i = \infty$, and then run a binary search on p_j. The highest value p_j for which the demand under these prices contains j is the marginal value of j relative to S.

Once we can solve marginal value queries, any value query can be solved by $v(S) = \sum_{j \in S}(v(\{i \in S | i \leq j\}) - v(\{i \in S | i < j\})$. □

Lemma 11.23 *An exponential number of value queries may be required for simulating a single demand query.*

The proof of Lemma 11.23 is left for Exercise 11.3.

11.5.2 Solving the Linear Program

Many algorithms for handling combinatorial auctions or special cases of combinatorial auctions start by solving the linear programming relaxation of the problem, shown in Section 11.3.1. A very useful and surprising property of demand queries is that they allow solving the linear-programming relaxation efficiently. This is surprising since the linear program has an exponential number of variables. The basic idea is

[2] In Section 11.7 we consider more general demand queries where a price of a bundle is not necessarily the sum of the prices of its items.

to solve the dual linear program using the Ellipsoid method. The dual program has a polynomial number of variables, but an exponential number of constraints. The Ellipsoid algorithm runs in polynomial time even on such programs, provided that a "separation oracle" is given for the set of constraints. Surprisingly, such a separation oracle can be implemented by presenting a single demand query to each of the bidders.

Consider the linear-programming relaxation (LPR) for the winner determination problem in combinatorial auctions, presented in Section 11.3.

Theorem 11.24 *LPR can be solved in polynomial time (in n, m, and the number of bits of precision t) using only demand queries with item prices.*[3]

PROOF Consider the dual linear program, DLPR, presented in Section 11.3 (Equations 11.8–11.9). Notice that the dual problem has exactly $n + m$ variables but an exponential number of constraints.

Recall that a separation oracle for the Ellipsoid method, when given a possible solution, either confirms that it is a feasible solution, or responds with a constraint that is violated by the possible solution. Consider a possible solution $(\overrightarrow{u}, \overrightarrow{p})$ for the dual program. We can rewrite Constraint 11.8 of the dual program as $u_i \geq v_i(S) - \sum_{j \in S} p_j$. Now, a demand query to bidder i with prices p_j reveals exactly the set S that maximizes the RHS of the previous inequality. Thus, in order to check whether $(\overrightarrow{u}, \overrightarrow{p})$ is feasible it suffices to (1) query each bidder i for his demand D_i under the prices p_j; (2) check only the n constraints $u_i + \sum_{j \in D_i} p_j \geq v_i(D_i)$ (where $v_i(D_i)$ can be simulated using a polynomial sequence of demand queries as was previously observed). If none of these are violated then we are assured that $(\overrightarrow{u}, \overrightarrow{p})$ is feasible; otherwise, we get a violated constraint.

What is left to be shown is how the *primal* program can be solved. (Recall that the primal program has an exponential number of variables.) Since the Ellipsoid algorithm runs in polynomial time, it encounters only a polynomial number of constraints during its operation. Clearly, if all other constraints were removed from the dual program, it would still have the same solution (adding constraints can only decrease the space of feasible solutions). Now take the "reduced dual" where only the constraints encountered exist, and look at its dual. It will have the same solution as the original dual and hence of the original primal, but with a polynomial number of variables. Thus, it can be solved in polynomial time, and this solution clearly solves the original primal program, setting all other variables to zero. □

11.5.3 Approximating the Social Welfare

The final part of this section will highlight some of the prominent algorithmic results for combinatorial auctions. Some of these results are obtained by solving the LP relaxation. Figure 11.5.2 lists state-of-the-art results for the point in time in which this chapter

[3] The solution will have a polynomial-size support (nonzero values for $x_{i,S}$), and thus we will be able to describe it in polynomial time.

Class	Queries	Approx	IC approx	Lower bound
Gen	Any	$\sqrt{m}$	$\frac{m}{\sqrt{\log m}}$ $\sqrt{m}$ (rand)	$m^{\frac{1}{2}-\epsilon}$ Section 1.6, [NS06]
	Value	$\frac{m}{\sqrt{\log m}}$	$\frac{m}{\sqrt{\log m}}$ [HKDMT04]	$\frac{m}{\log m}$ [BN05a, DS05]
	Demand	$\sqrt{m}$ [BN05a]	$\frac{m}{\sqrt{\log m}}$ $\sqrt{m}$ (rand) [LS05, DNS06]	$m^{\frac{1}{2}-\epsilon}$
SubA	Value	$\sqrt{m}$	$\sqrt{m}$ [DNS05]	$m^{\frac{1}{4}}$
	Demand	2 (rand) [Fei06]	$\sqrt{m}$	2 [DNS05]
XOS	Value	$\sqrt{m}$	$\sqrt{m}$	$m^{\frac{1}{4}}$ [DS06]
	Demand	2 [DNS05] $\frac{e}{e-1}$ (rand) [Fei06]	$\sqrt{m}$ $\log^2 m$ (rand) [DNS06]	$\frac{e}{e-1}$ [DNS05]
SubM	Value	2 [LLN06]	$\sqrt{m}$	$\frac{e}{e-1}$ [KLMM05]
	Demand	2 $\frac{e}{e-1}$-10^{-4} (rand) [FV06]	$\sqrt{m}$ $\log^2 m$ (rand)	$\frac{276}{275}$ [FV06]
Subs	Value	1 [Ber05]	1	
	Demand	1 [GS99, BM97]	1	
k**Dup**	Demand	$m^{\frac{1}{k+1}}$ [BKV05, DS05]	$k \cdot m^{\frac{1}{k-2}}$ [BGN03]	$m^{\frac{1}{k+1}-\epsilon}$ [BGN03, DS05]
Proc	Any	$\ln n$ [NS06]	-	$\log n$ [Nis02]

Figure 11.2. It describes the best algorithmic results, incentives compatible approximation results and lower bounds which are currently known for different classes of combinatorial-auction valuations. All results apply for a polynomial number of queries of the specified type. Results without references can be trivially derived from other entries in this table. The word "rand" implies that the result is achieved by a randomized algorithm; otherwise, the results correspond to deterministic algorithms only. Results that use ϵ hold for any $\epsilon > 0$. For the simplicity of the presentation, we ignore the constants of the asymptotic results (i.e., we drop the big-Oh and Ω notations). [NS06]: Nisan and Segal, 2006; [BN05a]: Blumrosen and Nisan, 2005; [DS05]: Dobzinski and Schapira, 2005; [LS05]: Lavi and Swamy, 2005; [DNS06]: Dobzinski et al., 2006; [Fei06]: Feige, 2006; [DNS05]: Dobzinski et al., 2005; [DS06]: Dobzinski and Schapira, 2006; [LLN06]: Lehmann et al., 2006; [KLMM05]: Khot et al., 2005; [FV06]: Feige and Vondrak, 2006; [Ber05]: Bertelsen, 2005; [GS99]: Gul and Stacchetti, 1999; [BM97]:Bikhchandani and Mamer, 1997; [BKV05]: Briest et al., 2005; [BGNO3]: Bartal et al., 2003; [Nis02]: Nisan, 2002.

was written. For each class of bidder valuations, we mention the best currently known polynomial-time approximation ratio, the optimal ratio that is currently achievable by ex-post Nash incentive-compatible mechanisms that run in polynomial time, and the best computational hardness result for the algorithmic problem (under standard computational assumptions). We also classify the results according to the queries they

use: unrestricted, value queries, or demand queries. In the figure, we refer the reader to the papers that established these results for more details. In particular, a randomized incentive-compatible mechanism that achieves a $O(\sqrt{m})$-approximation for general combinatorial auctions is discussed in Chapter 12. Below are the classes of valuations that we consider and their abbreviations:

Gen – General (unrestricted) valuations.
SubA – Subadditive valuations, i.e., where $v(S \cup T) \leq v(S) + v(T)$ for all S, T.
XOS – All valuations that can be represented by XOR-of-ORs bids with singleton atomic bundles (see Section 11.4).
SubM - Submodular valuations, i.e., where for every two bundles S and T we have that $v(S) + v(T) \geq v(S \cup T) + v(S \cap T)$.
Subs – (Gross-) substitutes valuations, see Definition 11.28 in Section 11.7.
k**Dup** – Combinatorial auctions with k duplicates of each good. Each bidder desires at most a single item of each good.
Proc – Procurement auctions, where a single buyer needs to buy a set of m items from n suppliers. The suppliers have privately known costs for bundles of items. The buyer aims to minimize the total cost paid.

It is known that Gen $\supset$ SubA $\supset$ XOS $\supset$ SubM $\supset$ Subs.

11.6 Communication Complexity

We already saw in Section 11.2.1 that solving the optimal allocation problem is NP-complete even for single-minded bidders and thus certainly for more general types of bidders. However, as mentioned, in practice one can usually solve problems with thousands or tens-of-thousands of items and bids optimally of near-optimally. Will it be possible to do the same for general valuations using some type of queries to the bidders? In other words: is the problem of representing the valuations an obstacle beyond the computational hardness? In this section we provide an affirmative answer: even if the auctioneer had unlimited computational power, then eliciting sufficient information from the bidders as to determine the optimal allocation would require an exponential amount of queries to the bidders – for any query type. We present this lower bound in a very general model – Yao's two-party communication complexity model – and thus it holds for essentially any model of iterative combinatorial auctions with any type of queries. Let us first introduce this model formally.

11.6.1 The Model and Statement of Lower Bound

The lower bound is obtained in Yao's standard model of two-player communication complexity. In this model we consider two players, Alice and Bob, each holding a valuation function. We can restrict ourselves to the special case where the value of each set is either 0 or 1. Thus, the inputs are monotone functions $v_1, v_2 : 2^M \rightarrow \{0, 1\}$. Alice and Bob must embark on a communication protocol whose final outcome is the declaration of an allocation (S, S^c) that maximizes $v_1(S) + v_2(S^c)$. The protocol specifies rules for exchanging bits of information, where Alice's message at each point

may depend only on v_1 and on previous messages received from Bob, while Bob's message at each point may depend only on v_2 and on previous messages received from Alice. No computational constraints are put on Alice and Bob – only communication is measured. The main result shows that:

Theorem 11.25 *Every protocol that finds the optimal allocation for every pair of 0/1 valuations v_1, v_2 must use at least $\binom{m}{m/2}$ bits of total communication in the worst case.*

Note that $\binom{m}{m/2}$ is exponential in m.[4] Since Yao's communication model is very powerful, the lower bound immediately applies to essentially all settings where v_1 and v_2 reside in "different places." In particular, to the case where the bidders reply to queries of the auctioneer (since a protocol with an auctioneer can be converted into one without an auctioneer, by sending all replies directly to each other and having Alice and Bob simulate the auctioneer's queries) and to any larger number of bidders (since the 2-bidder case is a special case where all bidders but two have null valuations.)

11.6.2 The Proof

Fix a communication protocol that for every input valuation pair (v_1, v_2) finds an optimal allocation S, S^c. We will construct a "fooling set": a set of valuation pairs with the property that the communication patterns produced by the protocol must be different for different valuation pairs. Specifically, for every 0/1 valuation v, we define the *dual valuation* v^* to be $v^*(S) = 1 - v(S^c)$. Note that (i) v^* is indeed a monotone 0/1 valuation, and (ii) for every partition (S, S^c), $S \subseteq M$, we have that $v(S) + v^*(S^c) = 1$.

Lemma 11.26 *Let $v \neq u$ be arbitrary 0/1 valuations. Then, in a welfare maximizing combinatorial auction, the sequence of bits transmitted on inputs (v, v^*) is not identical to the sequence of bits transmitted on inputs (u, u^*).*

Before we prove the lemma, let us see how the main theorem is implied. Since different input valuation pairs lead to different communication sequences, we see that the total possible number of communication sequences produced by the protocol is at least the number of valuation pairs (v, v^*), which is exactly the number of distinct 0/1 valuations v. The number of 0/1 valuations can be easily bounded from below by $2^{\binom{m}{m/2}}$ by counting only valuations such that $v(S) = 0$ for all $|S| < m/2$, $v(S) = 1$ for all $|S| > m/2$, and allowing $v(S)$ to be either 0 or 1 for $|S| = m/2$; there are $\binom{m}{m/2}$ sets of size $m/2$, so the total number of such valuations is exponential in this number. The protocol must thus be able to produce $2^{\binom{m}{m/2}}$ different communication sequences. Since these are binary sequences, at least one of the sequences must be of length at least $\binom{m}{m/2}$.

[4] More precisely, by Stirling's formula, $\binom{m}{m/2} \sim \sqrt{2/(\pi \cdot m)} \cdot 2^m$.

PROOF (of lemma) Assume, by way of contradiction, that the communication sequence on (v, v^*) is the same as on (u, u^*). We first show that the same communication sequence would also be produced for (v, u^*) and for (u, v^*). Consider the case of (v, u^*); i.e., Alice has valuation v and Bob has valuation u^*. Alice does not see u^* so she behaves and communicates exactly as she would in the (v, v^*) case. Similarly, Bob behaves as he would in the (u, u^*) case. Since the communication sequences in the (v, v^*) and the (u, u^*) cases are the same, neither Alice nor Bob ever notices a deviation from this common sequence, and thus never deviates themselves. In particular, this common sequence is followed also on the (v, u^*) case. Thus, the same allocation (S, S^c) is produced by the protocol in all four cases: (v, v^*), (u, u^*), (v, u^*), (u, v^*). We will show that this is impossible, since a single allocation cannot be optimal for all four cases.

Since $u \neq v$, we have that for some set T, $v(T) \neq u(T)$. Without loss of generality, $v(T) = 1$ and $u(T) = 0$, and so $v(T) + u^*(T^c) = 2$. The allocation (S, S^c) produced by the protocol must be optimal on the valuation pair (v, u^*), thus $v(S) + u^*(S^c) \geq 2$. However, since $(v(S) + v^*(S^c)) + (u(S) + u^*(S^c)) = 1 + 1 = 2$, we get that $u(S) + v^*(S^c) \leq 0$. Thus (S, S^c) is not an optimal allocation for the input pair (u, v^*) – contradiction to the fact that the protocol produces it as the output in this case as well. □

More complex lower bounds on communication allow us to prove tight lower bounds for iterative auctions in various setting. The above lower bound on communication can be extended to even approximating the social welfare.

Theorem 11.27 *For every $\epsilon > 0$, approximating the social welfare in a combinatorial auction to within a factor strictly smaller than $\min\{n, m^{1/2-\epsilon}\}$ requires exponential communication.*

Note that this is tight: achieving a factor of n is always trivial (by bundling all items together and selling them in a simple single-item auction), and for $n \geq \sqrt{m}$ there exists an $O(\sqrt{m})$ approximation (see Figure 11.5.2). Actually, most of the lower bounds described in Figure 11.5.2 are communication-complexity results.

11.7 Ascending Auctions

This section concerns a large class of combinatorial auction designs which contains the vast majority of implemented or suggested ones: *ascending auctions*. These are a subclass of iterative auctions with demand queries in which the prices can only increase. In this class of auctions, the auctioneer publishes prices, initially set to zero (or some other minimum prices), and the bidders repeatedly respond to the current prices by bidding on their most desired bundle of goods under the current prices. The auctioneer then repeatedly updates the prices by increasing some of them in some manner, until a level of prices is reached where the auctioneer can declare an allocation. There are several reasons for the popularity of ascending auctions, including their intuitiveness,

An item-price ascending auction for substitutes valuations:

Initialization:

For every item $j \in M$, set $p_j \leftarrow 0$.
For every bidder i let $S_i \leftarrow \emptyset$.

Repeat

For each i, let D_i be the demand of i at the following prices:
p_j for $j \in S_i$ and $p_j + \epsilon$ for $j \notin S_i$.
If for all i $S_i = D_i$, exit the loop;
Find a bidder i with $S_i \neq D_i$ and update:

- For every item $j \in D_i \setminus S_i$, set $p_j \leftarrow p_j + \epsilon$
- $S_i \leftarrow D_i$
- For every bidder $k \neq i$, $S_k \leftarrow S_k \setminus D_i$

Finally: Output the allocation $S_1, ..., S_n$.

Figure 11.3. An item-price ascending auction that ends up with a nearly optimal allocation when bidders' valuations have the (gross) substitutes property.

the fact that private information is only partially revealed, that it is clear that they will terminate, and that they may increase the seller's revenue in some settings.

We will describe auctions that belong to two families of ascending auctions. One family uses a simple pricing scheme (item prices), and guarantees economic efficiency for a restricted class of bidder valuations. The second family is socially efficient for every profile of valuations, but uses a more complex pricing scheme – prices for bundles – extending the demand queries defined in Section 11.5.

11.7.1 Ascending Item-Price Auctions

Figure 11.3 describes an auction that is very natural from an economic point of view: increase prices gradually, maintaining a tentative allocation, until no item that is tentatively held by one bidder is demanded by another. Intuitively, at this point demand equals supply and we are close to a Walrasian equilibrium discussed earlier in Section 11.3, which, by the first welfare theorem, is socially efficient.

Of course, we know that a Walrasian equilibrium does not always exist in a combinatorial auction, so this cannot always be true. The problem is that the auction does not ensure that items are not underdemanded: it may happen that an item that was previously demanded by a bidder is no longer so. The following class of valuations are those in which this cannot happen.

Definition 11.28 A valuation v_i satisfies the *substitutes (or gross-substitutes)* property if for every pair of item-price vectors $\overrightarrow{q} \geq \overrightarrow{p}$ (coordinate-wise comparison), we have that the demand at prices q contains all items in the demand at prices p whose price remained constant. Formally, for every $A \in argmax_S\{v(S) - \sum_{j \in S} p_j\}$, there exists $D \in argmax_S\{v(S) - \sum_{j \in S} q_j\}$, such that $D \supseteq \{j \in A | p_j = q_j\}$.

That is, the only items that could drop from the demand when prices change from $\overrightarrow{p}$ to $\overrightarrow{q}$ are those whose price has strictly increased. The substitutes property rules out any form of complementarities. For example, a single-minded bidder who is willing to pay 10 for the complete bundle $\{a, b\}$ will demand both items at prices (3, 3), but if the price of b is raised to 8, this bidder will no longer demand any item – contrarily to the requirement of a substitutes valuation. Exercise 11.6 shows that, in general, substitutes valuations must be submodular. It is not difficult to see that this class of valuations contains the classes of additive valuations, unit-demand valuations, and downward-sloping valuations (see Definitions 11.17 and 11.19). With such valuations, the auction maintains the property that every item is demanded by some bidder. The auction terminates when all the bidders receive their demanded bundles, and consequently, the auction converges to a (nearly) Walrasian equilibrium.

Definition 11.29 An allocation $S_1, \ldots, S_n$ and a prices $p_1, \ldots, p_m$ are an ϵ-Walrasian equilibrium if $\bigcup_i S_i \supseteq \{j | p_j > 0\}$ and for each i, S_i is a demand of i at prices p_j for $j \in S_i$ and $p_j + \epsilon$ for $j \notin S_i$.

Theorem 11.30 *For bidders with substitutes valuations, the auction described in Figure 11.3 ends with an ϵ-Walrasian equilibrium. In particular, the allocation achieves welfare that is within $n\epsilon$ from the optimal social welfare.*

PROOF The theorem will follow from the following key claim:

Claim 11.31 *At every stage of the auction, for every bidder i, $S_i \subseteq D_i$.*[5]

First notice that this claim is certainly true at the beginning. Now let us see what an update step for some bidder i causes. For i itself, S_i after the step is exactly equal to D_i (note that the changes in prices of items just added to S_i exactly matches those defining D_i). For $k \neq i$, two changes may occur at this step: first, items may have been taken from S_k by i, and second the prices of items outside of S_k may have increased. The first type of change makes S_k smaller while not affecting D_k. The second type of change does not affect S_k and the substitutes property directly implies that the only items that can be removed from D_k are those whose price strictly increased and are thus not in S_k.

Once we have this claim, it is directly clear that no item that was ever demanded by any player is ever left unallocated; i.e., $\bigcup_i S_i$ always contains all items whose price is strictly positive. Since the auction terminates only when all $D_i = S_i$ we get an ϵ-Walrasian equilibrium. The fact that an ϵ-Walrasian equilibrium is close to socially optimal is obtained just as in the proof of the first welfare theorem (Theorem 11.13). □

Since prices are only going up, the algorithm terminates after at most $m \cdot v_{\max}/\epsilon$ stages, where $v_{\max}$ is the maximum valuation. It may also be useful to view this auction

[5] For simplicity of presentation, the algorithm assumes that D_i is unique. In the general case, the claim is that S_i is contained in some demand bundle D_i, and the auction is required to pick such a D_i.

as implementing a *primal-dual* algorithm. The auction starts with a feasible solution to the dual linear program (here, zero prices), and as long as the complementary-slackness conditions are unsatisfied proceeds by improving the solution of the dual program (i.e., increasing some prices).

Finally, we will address the strategic behavior of the bidders in such ascending auctions. Will strategic bidders act myopically and truthfully reveal their demand in these auctions? If the valuation functions have complementarities, then bidders will clearly have strong incentives not to report their true preferences, due to a problem known as the *exposure problem*: Bidders who bid for a complementary bundle (e.g., a pair of shoes), are exposed to the risk that part of the bundle (the left shoe) may be taken from them later, and they are left liable for the price of the rest of the bundle (the right shoe) that is worthless for them.

However, even for substitutes preferences the incentive issues are not solved. The prices in Walrasian equilibria are not necessarily VCG prices, and therefore truthful bidding is not an ex-post equilibrium.[6] The strategic weakness of Walrasian equilibria is that bidders may have the incentive to demand smaller bundles of items (*demand reduction*), in order to lower their payments. The following example illustrates such a scenario.

Example 11.32 Consider two items a and b and two players, Alice and Bob, with the following substitutes valuations:

	v(a)	**v(b)**	**v(ab)**
Alice	4	4	4
Bob	5	5	10

For these valuations, the auction in Figure 11.3 will terminate at the Walrasian equilibrium prices $p_a = 4$, $p_b = 4$, where Bob receives both items, and earning him a payoff of 2. If Bob placed bids only on a during the auction, then the auction would stop at zero prices, allocating a to Bob and b to Alice. With this demand reduction, Bob improves his payoff to 5.

11.7.2 Ascending Bundle-Price Auctions

As we saw, not every profile of valuations has a Walrasian equilibrium. The next type of auction that we describe will reach an equilibrium that involves a more complex pricing scheme. We start by describing this extended notion of equilibrium, allowing *personalized bundle prices* – a distinct price per each possible bundle and for each bidder. That is, personalized bundle prices specify a price $p_i(S)$ per each bidder i and every bundle S. We can naturally generalize the notion of the demand of bidder i under such prices to $argmax_S(v_i(S) - p_i(S))$.

[6] When we further restrict the class of substitutes valuations such that each bidder desires at most one item ("unit-demand" valuations, see Definition 11.17), then it is known that a similar auction reaches the lowest possible Walrasian-equilibrium prices that are also VCG prices, and hence these auctions are ex-post Nash incentive compatible (see Chapter 9).

A bundle price auction:

Initialization: For every player i and bundle S, let $p_i(S) \leftarrow 0$.

Repeat

- Find an allocation $T_1, ..., T_n$ that maximizes revenue at current prices, i.e., $\sum_{i=1}^{n} p_i(T_i) \geq \sum_{i=1}^{n} p_i(Y_i)$ for any other allocation $Y_1, ..., Y_n$. (Bundles with zero prices will not be allocated, i.e., $p_i(T_i) > 0$ for eve ry i.)
- Let L be the set of losing bidders, i.e., $L = \{i | T_i = \emptyset\}$.
- For every $i \in L$ let D_i be a demand bundle of i under the prices $\overrightarrow{p_i}$.
- If for all $i \in L$, $D_i = \emptyset$ then terminate.
- For all $i \in L$ with $D_i \neq \emptyset$, let $p_i(D_i) \leftarrow p_i(D_i) + \epsilon$.

Figure 11.4. A bundle price auction which terminates with the socially efficient allocation for any profile of bidders.

Definition 11.33 Personalized bundle prices $\overrightarrow{p} = \{p_i(S)\}$ and an allocation $S = (S_1, \ldots, S_n)$ are called a *competitive equilibrium* if:

- For every bidder i, S_i is a demand bundle, i.e., for any other bundle $T_i \subseteq M$, $v_i(S_i) - p_i(S_i) \geq v_i(T_i) - p_i(T_i)$.
- The allocation S maximizes *seller's revenue* under the current prices, i.e., for any other allocation $(T_1, \ldots, T_n)$, $\sum_{i=1}^{n} p_i(S_i) \geq \sum_{i=1}^{n} p_i(T_i)$.

It is easy to see that with personalized bundle prices, competitive equilibria always exist: any welfare-maximizing allocation with the prices $p_i(S) = v_i(S)$ gives a competitive equilibrium. This may be viewed as the Second Welfare Theorem (see Theorem 11.15) for this setting. Even this weak notion of equilibrium, however, guarantees optimal social welfare:

Proposition 11.34 *In any competitive equilibrium* $(\overrightarrow{p}, S)$ *the allocation maximizes social welfare.*

PROOF Let $(\overrightarrow{p}, S)$ be a competitive equilibrium, and consider some allocation $T = (T_1, \ldots, T_n)$. Since S_i is a demand bundle under the prices $\overrightarrow{p_i}$ for every bidder i, we have that $v_i(S_i) - p_i(S_i) \geq v_i(T_i) - p_i(T_i)$. Summing over all the bidders, together with $\sum_{i=1}^{n} p_i(S_i) \geq \sum_{i=1}^{n} p_i(T_i)$, we get that the welfare in the allocation S exceeds the welfare in T. □

Several iterative auctions are designed to end up with competitive equilibria. Figure 11.4 describes a typical one. At each stage the auctioneer computes a tentative allocation that maximizes his revenue at current prices – which we view as the current bids. All the losing bidders then "raise their bids" on their currently demanded bundle. When no losing bidder is willing to do so, we terminate with an approximately competitive equilibrium.

Definition 11.35 A bundle S is an ϵ-*demand* for a player i under the bundle prices $\overrightarrow{p_i}$ if for any other bundle T, $v_i(S) - p_i(S) \geq v_i(T) - p_i(T) - \epsilon$. An

ϵ-competitive equilibrium is similar to a competitive equilibrium (Definition 11.33), except each bidder receives an ϵ-demand under the equilibrium prices.

Theorem 11.36 *For any profile of valuations, the bundle-price auction described in Figure 11.4 terminates with an ϵ-competitive equilibrium. In particular, the welfare obtained is within $n\epsilon$ from the optimal social welfare.*

PROOF At each step of the auction at least one price will be raised. Since a bundle price will clearly never exceed its value, the auction will terminate eventually (although this may take exponentially many steps). Since the allocation at each step is clearly revenue maximizing, it suffices to show that, upon termination, each bidder receives an ϵ-demand.

Losing bidders will clearly receive their demand, the empty set, since this is the condition of termination. A winning bidder i gets an ϵ-demand bundle since the auction maintains the property that every bundle T_i with $p_i(T_i) > 0$ is an ϵ-demand. To see this notice that $p_i(T_i) > 0$ implies that at some previous round T_i was the demand of bidder i. At that point, T_i was the exact demand, and thus, an ϵ-demand bundle after the price increment. Since the last time that the bidder demanded (the current) T_i, only prices of other bundles have increased, clearly maintaining the property.

Finally, the near optimality of the social welfare in an approximate competitive equilibrium follows the same arguments as in Proposition 11.34. □

Notice that while the auction always terminates with a (near) optimal allocation, this may require exponential time in two respects: first, the number of stages may be exponential, and, second, each stage requires the auctioneer to solve an NP-hard optimization problem. Of course, we know that this is unavoidable and that, indeed, exponential communication and computation are required in the worst case. Variants of this auction may be practically faster by allowing bidders to report a *collection* of demand bundles at each stage and increase the prices of all of them (in particular, prices of supersets of a reported demand bundle can be, w.l.o.g., maintained to be at least as high as that of the bundle itself.).

The prices generated by this auction are not VCG prices and thus players are not strategically motivated to act myopically and truthfully report their true demand at each stage.[7] One weak positive equilibrium property is achieved when each bidder is committed in advance to act according to a fixed valuation ("proxy bidding"). Then, the auction admits ex-post Nash equilibria, but these equilibria require the participants to possess considerable knowledge of the preferences of the other bidders.

More complex variants of the auction may charge VCG prices from the bidders rather then the equilibrium prices obtained. While this will have the obvious advantage that truthful bidding will be an ex-post Nash equilibrium, it turns out that this will lose some nice properties possessed by the equilibrium prices reached (like resistance to bidder collusion and to false-name bids in some settings).

[7] When bidders have substitutes valuations (Definition 11.28); however, the auction does terminate at VCG prices.

11.8 Bibliographic Notes

This chapter gives only the very basics of the theoretical treatment of combinatorial auctions. Much more information appears in the recently published books (Cramton et al., 2006; Milgrom, 2004). Information about spectrum auctions can be found, for example, in (FCC auctions home page; Cramton, 2002, 2006), and a nice description of industrial applications can be found in (Sandholm, 2006a).

The earliest work on the computational complexity of the winner determination problem in combinatorial auctions is Rothkhof et al. (1998),which contains algorithms for various special cases. Other early work on algorithms for winner determination is due to Sandholm (2002),who also noted the NP-hardness of the problem and of its approximation. The hardness of approximation is based on the hardness of approximation of clique size of Håstad (1999), with the strong version as stated appearing in Zuckerman (2006). Recent surveys on winner determination algorithms appear in (Lehmann et al., 2006b, Muller, 2006; Sandholm, 2006b). The single-minded case was studied in Lehmann et al. (2002) on which Section 11.2.2 is based. Additional results for the single-minded case and generalizations of it can be found in Babaioff et al. (2005) and the references within.

The LP formulation of the problem and the relation of its integrality gap to Walrasian equilibria were studied in Bikhchandani and Mamer (1997) and Bikhchandani and Ostroy (2002).

Bidding languages were studied in a general and formal way in Nisan (2000) on which Section 11.4 is based. Dummy items were suggested in Fujishima et al. (1999). A detailed survey of bidding languages appears in Nisan (2006).

A systematic study of the query model can be found in Blumrasen and Nisan (2005a). The fact that the linear program can be solved in polynomial time using demand queries appears in Nisan and Segal (2006) and Blumfosen and Nisan (2005a). Applications of this fact for various approximation algorithms can be found in Dobzinski et al. (2005), Lavi and Swamy (2000), and Feige and Vondrak (2006). Relations of the query model to machine-learning theory is described in Blum et al. (2004) and Lehaie and Parkes (2004) and the references within.

The analysis of the communication complexity of combinatorial auctions was initiated in Nisan and Segal (2006) on which Section 11.6 is based. A more comprehensive treatment of this subject can be found in the survey (Segal, 2006). A detailed exposition of the theory of communication complexity can be found in Kushilevitz and Nisan (1997).

Ascending item-price combinatorial auctions for the (gross)-substitutes case were first suggested by Demange et al. (1986), extending their use for matching Kelso and Crawford (1982). These were further studied in Bikhchandani and Mamer (1997), Gul and Stacchetti (1999, 2000), Milgrom (2004), and Ausubel (2006). Socially-efficient ascending bundle-price auctions were suggested in Parkes and Ungar (2000) and Ausubel and Milgrom (2002), and hybrid designs that use both item- and bundle prices appear in Kelly and Steinberg (2000) and Cramton et al. (2006). Ausubel and Milgrom (2002) also discussed connections to coalitional games and their core. A detailed study of ascending auctions and their limitations may be found in Blumrosen and Nisan (2005b). A comprehensive survey can be found in Parkes (2006).

Exercise 11.1 is from Rothkhof et al. (1998). A proof for Exercise 11.2 can be found in Muller (2006). Exercise 11.3 is from Blumrosen and Nisan (2005a). Exercise 11.4 is from Dobzinski et al. (2005). Exercise 11.5 is from Nisan (2000). Exercise 11.6 is from Gul and Stacchetti (1999). Exercise 11.7 is from Parkes (2001) and Blumrosen and Nisan (2005b). Exercise 11.8 is from Blumrosen and Nisan (2005b). The algorithm in exercise 11.9 is the classic one for SET-COVER by Lovasz (1975), see also Nisan (2002).

Acknowledgments

The authors thank Shahar Dobzinki, Jason Hartline, and David Parkes for their valuable comments on an earlier draft of this chapter.

Bibliography

L.M. Ausubel. An efficient dynamic auction for heterogeneous commodities. *Amer. Econ. Rev.*, 96(3):602–629, 2006.

L.M. Ausubel and P.R. Milgrom. Ascending auctions with package bidding. *Front. Theor. Econ.*, 1:1–42, 2002.

M. Babaioff, R. Lavi, and E. Pavlov. Mechanism design for single-value domains. In *20th Ntl. Conf. Artificial Intelligence*, pp. 241–247, 2005.

Y. Bartal, R. Gonen, and N. Nisan. Incentive compatible multi unit combinatorial auctions. In *9th Conf. Theor. Aspects of Rationality and Knowledge*, pp. 72–87, 2003.

A. Bertelsen. *Substitutes Valuations and $m^\natural$-Concavity*. M.Sc. Thesis, The Hebrew University of Jerusalem, 2005.

S. Bikhchandani and J.W. Mamer. Competitive equilibrium in an exchange economy with indivisibilities. *J. Economic Theory*, 74:385–413, 1997.

S. Bikhchandani and J.M. Ostroy. The package assignment model. *J. Economic Theory*, 107:377–406, 2002.

A. Blum, J.C. Jackson, T. Sandholm, and M.A. Zinkevich. Preference elicitation and query learning. *J. Mach. Learn. Res.*, 5:649–667, 2004.

L. Blumrosen and N. Nisan. On the computational power of iterative auctions I: demand queries. Discussion paper no. 381, The Center for the Study of Rationality, The Hebrew University, 2005a. An extended abstract in EC'05 contained preliminary results.

L. Blumrosen and N. Nisan. On the computational power of iterative auctions II: Ascending auctions. Discussion paper no. 382, The Center for the Study of Rationality, The Hebrew University, 2005b. An extended abstract in EC'05 contained preliminary results.

P. Briest, P. Krysta, and B. Vöcking. Approximation techniques for utilitarian mechanism design. In *the 37th ACM Symp. Theor. Comp.*, pp. 39–48, 2005.

P. Cramton. In Martin Cave, Sumit Majumdar, and Ingo Vogelsang, eds., *Handbook of Telecommunications Economics. Chapter 14: Spectrum auctions*. Elsevier Science B.V., 2002.

P. Cramton. In P. Cramton, Y. Shoham, and R. Steinberg, eds., *Combinatorial Auctions. Chapter 5. Simultaneous Ascending Auctions*. MIT Press, 2006.

P. Cramton, L.M. Ausubel, and P.R. Milgrom. In P. Cramton, Y. Shoham, and R. Steinberg, eds., Combinatorial Auctions. Chapter 5. The Clock-Proxy Auction: A Practical Combinatorial Auction Design. MIT Press, 2006.

P. Cramton, Y. Shoham, and R. Steinberg (Editors). *Combinatorial Auctions*. MIT Press, 2006.

G. Demange, D. Gale, and M. Sotomayor. Multi-item auctions. *J. Political Econ.*, 94:863–872, 1986.

S. Dobzinski, N. Nisan, and M. Schapira. Approximation algorithms for combinatorial auctions with complement-free bidders. In *37th ACM Symp. Theory Computing*, pp. 610–618, 2005.

S. Dobzinski, N. Nisan, and M. Schapira. Truthful randomized mechanisms for combinatorial auctions. In *38th Annual ACM Symp. Theory of Computing*, pp. 644–652, 2006.

S. Dobzinski and M. Schapira. Optimal upper and lower approximation bounds for k-duplicates combinatorial auctions. Working paper, the Hebrew University, 2005.

S. Dobzinski and M. Schapira. An improved approximation algorithm for combinatorial auctions with submodular bidders. In *Proc. 17th Annual ACM-SIAM Symp. Disc. Algo.*, pp. 1064–1073, 2006.

FCC auctions home page. http://wireless.fcc.gov/auctions.

U. Feige. On maximizing welfare where the utility functions are subadditive. In *38th ACM Symp. Theory of Computing*, pp. 41–50, 2006.

U. Feige and J. Vondrak. Approximation algorithms for allocation problems: Improving the factor of 1-1/e. In *47th Annual IEEE Symp. Foundations of Computer Science*, pp. 667–676, 2006.

Y. Fujishima, K. Leyton-Brown, and Y. Shoham. Taming the computational complexity of combinatorial auctions: Optimal and approximate approaches. In *16th Intl. Joint Conf. Artificial Intelligence*, Stockholm, Sweden, 1999.

F. Gul and E. Stacchetti. Walrasian equilibrium with gross substitutes. *J. Econ. Theor.*, 87:95–124, 1999.

F. Gul and E. Stacchetti. The English auction with differentiated commodities. *J. Econ. Theor.*, 92(3):66–95, 2000.

J. Håstad. Clique is hard to approximate to within $n^{1-\epsilon}$. *Acta Mathematica*, 182, 1999.

R. Holzman, N. Kfir-Dahav, D. Monderer, and M. Tennenholtz. Bundling equilibrium in combinatrial auctions. *Games Econ. Behav.*, 47:104–123, 2004.

F. Kelly and R. Steinberg. A combinatorial auction with multiple winners for universal service. *Management Sci.*, 46:586–596, 2000.

A.S. Kelso and V.P. Crawford. Job matching, coalition formation, and gross substitutes. *Econometrica*, 50:1483–1504, 1982.

S. Khot, R.J. Lipton, E. Markakis, and A. Mehta. Inapproximability results for combinatorial auctions with submodular utility functions. In *1st Workshop on Internet and Network Economics*, pp. 92–101, 2005.

E. Kushilevitz and N. Nisan. *Communication Complexity*. Cambridge University Press, 1997.

S. Lahaie and D.C. Parkes. Applying learning algorithms to preference elicitation. In *5th ACM Conf. Elect. Commerce*, pp. 180–188, 2004.

R. Lavi and C. Swamy. Truthful and near-optimal mechanism design via linear programming. In *46th Annual IEEE Symp. Fdns. of Computer Science*, pp. 595–604, 2005.

B. Lehmann, D. Lehmann, and N. Nisan. Combinatorial auctions with decreasing marginal utilities. *Games Econ. Behav.*, 55(2):270–296, 2006a.

D. Lehmann, R. Müller, and T. Sandholm. In P. Cramton, Y. Shoham, and R. Steinberg, eds., *Combinatorial Auctions. Chapter 12. The Winner Determination Problem*. MIT Press, 2006b.

D. Lehmann, L.I. O'Callaghan, and Y. Shoham. Truth revelation in approximately efficient combinatorial auctions. *J. ACM*, 49(5):577–602, 2002.

L. Lovasz. On the ratio of optimal integral and fractional covers. *Discrete Mathematics*, 13:383–390, 1975.

P. Milgrom. Putting Auction Theory to Work: the simultaneous ascending auction. *J. Political Econ.*, 018(2):245–272, 2000.

P. Milgrom. *Putting Auction Theory to Work*. Cambridge University Press, 2004.

R. Muller. In P. Cramton, Y. Shoham, and R. Steinberg, eds., *Combinatorial Auctions. Chapter 13. Tractable Cases of the Winner Determination Problem*. MIT Press, 2006.

N. Nisan. Bidding and allocation in combinatorial auctions. In *ACM Conf. on Elect. Commerce*, 2000.

N. Nisan. The communication complexity of approximate set packing and covering. In *29th Intl. Colloq. Auto., Langs. Progr.*, pp. 868–875, 2002.

N. Nisan. In P. Cramton, Y. Shoham, and R. Steinberg, eds., *Combinatorial Auctions. Chapter 1. Bidding Languages*. MIT Press, 2006.

N. Nisan and I. Segal. The communication requirements of efficient allocations and supporting prices. *J. Econ. Theor.*, 129(1):192–224, 2006.

D.C. Parkes. *Iterative Combinatorial Auctions: Achieving Economic and Computational Efficiency.* PhD Thesis, Department of Computer and Information Science, University of Pennsylvania, 2001.

D.C. Parkes. In P. Cramton, Y. Shoham, and R. Steinberg, eds., *Combinatorial Auctions. Chapter 3. Iterative Combinatorial Auctions*. MIT Press, 2006.

D.C. Parkes and L.H. Ungar. Iterative combinatorial auctions: Theory and practice. In *17th Ntl. Conf. on Artificial Intelligence*, pp. 74–81, 2000.

M.H. Rothkhof, A. Peke**c**, and R.M. Harstad. Computationally manageable combinatorial auctions. *Management Sci.*, 44(8):1131–1147, 1998.

T. Sandholm. Algorithm for optimal winner determination in combinatorial auctions. *Artif. Intellig.*, 135:1–54, 2002.

T. Sandholm. Expressive commerce and its application to sourcing. In *Innovative Applications of Artificial Intelligence*, 2006.

T. Sandholm. In P. Cramton, Y. Shoham, and R. Steinberg, eds., *Combinatorial Auctions. Chapter 14. Optimal Winner Determination Algorithms*. MIT Press, 2006.

I. Segal. In P. Cramton, Y. Shoham, and R. Steinberg, eds., *Combinatorial Auctions. Chapter 11. The Communication Requirements of Combinatorial Allocation Problems*. MIT Press, 2006.

D. Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. In *38th Annual ACM Symp. Theory of Computing*, pp. 681–690, 2006.

Exercises

11.1 Consider an auction for items $1,\ldots,m$ where each bidder is single minded and desires an interval of consecutive items, i.e., $S_i = \{j | k_i \leq j \leq l_i)$ where $1 \leq k_i \leq l_i \leq m$. Prove that in this case the socially efficient allocation can be determined in polynomial time.

11.2 Consider combinatorial auctions for m items among n bidders, where each valuation is represented simply as a vector of $2^m - 1$ numbers (a value for each subset of items). Prove that the optimal allocation can be computed in time that is polynomial in the input length: $n(2^m - 1)$. (An immediate conclusion is that when $m{=}O(\log n)$ then the optimal allocation can be computed in polynomial time in n.)
Hint: Use dynamic programming

11.3 Show a class of valuations for bidders in combinatorial auctions for which a single demand query can reveal enough information for determining the optimal allocation, but this task may require an exponential number (in the number of items) of value queries. (This actually proves Lemma 11.23 from Section 11.5.1.)
Hint: Use the fact that the number of distinct bundles of size $\frac{m}{2}$, out of m items, is exponential in m.

11.4 A valuation v is called *subadditive* if for every two bundles S, T, $v(S) + v(T) \geq v(S \cup T)$. Prove that for any $\epsilon > 0$, achieving a $2 - \epsilon$ approximation in a combinatorial auction with sub additive bidders requires exponential communication.
Hint: Construct a reduction from Theorem 11.27 in Section 11.6.

11.5 The *majority* valuation assigns a value of 1 to any bundle of at least $\frac{m}{2}$ items, and 0 to all other bundles. Prove that representing this valuation using an OR^* formula requires size of at least $\binom{m}{\frac{m}{2}}$.

11.6 Prove that every (gross) substitutes valuation is submodular.

11.7 Consider an *anonymous-price* variant of the bundle-price ascending auctions described in Figure 11.4): The same ascending-price process is performed, except that at every stage, all bidders observe the same bundle prices $\{p(S)\}_{S\subseteq M}$. At each stage, the prices of bundles that are demanded by at least one losing bidder are raised by ϵ.

Show that when all the valuations are *super additive* such an auction terminates with the socially efficient allocation. (A valuation is super additive if for every two bundles S,T, $v(S)+v(T)\leq v(S\cup T)$.)

Hint: First show that if bidder i receives the bundle T_i in the optimal allocation, then $v_i(T_i)\geq v_j(T_i)$ for every bidder j.

11.8 Consider a pair of valuations with the following form (where $0<\alpha,\beta<1$ are unknown to the seller):

	v(**ab**)	**v**(**a**)	**v**(**b**)
Alice	2	α	β
Bob	2	2	2

Prove that no item-price ascending auction can reveal enough information for determining the socially efficient allocation for such valuations.

11.9 In a *procurement auction* with single-minded bidders, a single buyer needs to buy a set of m items from n possible suppliers. Each supplier i can provide a single set of items S_i for a privately known price v_i. The buyer needs to buy all items, and aims to minimize the total price paid.

(a) Prove that the following greedy algorithm finds a $(1+\ln m)$-approximation to the optimal procurement:
- *Initialize R to contain all m items, and $W\leftarrow\emptyset$.*
- *Repeat until $R=\emptyset$: Choose $j\in argmax_k \frac{v_k}{|R\cap S_k|}$, and let $W=W\cup\{j\}$ and $R=R\setminus S_j$.*

(b) Deduce an incentive-compatible polynomial-time $(1+\ln m)$-approximation mechanism for procurement auctions among single-minded bidders. Show first that the allocation scheme defined by the algorithm is monotone, and identify the "critical values" to be paid by the winning suppliers.