

Market Structure and Multidimensional Auction Design for Computational Economies

by

Peter R. Wurman

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in The University of Michigan
2000

Doctoral Committee:

Associate Professor Michael P. Wellman, Chair
Associate Professor Edmund H. Durfee
Professor Jeffrey K. MacKie-Mason
Professor Ennio S. Stacchetti

Abstract

Market Structure and Multidimensional Auction Design for Computational Economies

by

Peter R. Wurman

Chair: Michael P. Wellman

Research in multiagent systems and electronic commerce applications often involves the task of designing a mechanism to allocate scarce resources among self-interested agents. Microeconomics has developed an extensive theory on the use of markets and auctions to solve such resource allocation problems. However, in many of the problems that arise in computer science applications, resources are discrete and the agents often realize positive synergies, or complementarities, between resource types. These problem features pose particular challenges to mechanism designers.

This thesis supports the application of microeconomic theory to multiagent systems and electronic commerce problems in three ways. First, it defines a structural view of markets that allows clustering of resources into multidimensional auctions. This structural view facilitates the exploration of tradeoffs between communication, computation, and convergence properties when only a subset of resources exhibit complementarities.

Second, this thesis provides a parametrized view of the space of auction designs. This parametrization adds considerable structure to the design problem, and provides a concise and consistent language with which to communicate the rules to participating agents.

Finally, this thesis extends the state of the art in multidimensional auctions for both discrete and continuous resources. Perhaps most significantly, this dissertation introduces the Ascending k -Bundle Auctions, a family of single-sided auctions for discrete resources that associate payments with bundles. These auctions have a desirable equilibrium property—no agent prefers some other bundle at the posted payments to the one it is allocated by the mechanism.

*To Nancy, Timothy, Erik and Benjamin,
and the things that bring us joy.*

Acknowledgements

This dissertation would not have come into being without the guidance of my advisor, Professor Michael Wellman. Mike has been, and will continue to be, my academic role model. His technical skills are unmatched, and the time and care he puts into feedback has greatly improved my work. In my mind, he is an exemplar for all research advisors.

This dissertation also owes its existence to my wife Nancy, who supported us financially, and believed in me even when I doubted myself. I may never be able to fully repay her, but I intend to try.

This document is not the only thing that owes its existence to Nancy. When I started graduate school, we had one son—Timothy. During my tenure as a graduate student, Nancy has blessed me with two more: Erik and Benjamin. Although it has been challenging juggling family and work schedules, I would not trade a single father-son moment for anything in the world. After a long day banging my head against a particularly difficult proof, I would come home and read “There’s a Monster at the End of this Book,” and everything would fall back into perspective.

I would also like to thank my parents for their confidence, and their lifelong emphasis on education. I only hope I can pass such gifts on to my children.

My dissertation committee naturally played an important role. First and foremost, they set the standard by which I measure my work. Jeff MacKie-Mason, in particular, has been very supportive, and is the de facto co-chair of this dissertation. Ennio Stacchetti reviewed my work on bundle pricing, and recognized that a big element of my original proof of the existence of equilibrium payments reduced to the assignment problem. I thank Ed Durfee for his willingness to participate in a dissertation of such mixed heritage.

I am indebted to Stuart Feldman and IBM’s Institute for Advanced Commerce for providing both outside validation and financial support by means of the 1998 IBM/IAC fellowship for the best E-commerce dissertation proposal. This work has also been funded by the DARPA program in Information Survivability, and a U-M Rackham Research

grant.

Several other faculty have played a significant role in my journey. Carl Berger was a shining light during my many years at the Office of Instructional Technology. His obvious love for his job and his role in the community inspired me to pursue a Ph.D. Dennis Severance provided key advice at several junctures. John Laird has been a model of how to balance an academic career and a family, and nurture both.

Much of the content of this thesis was inspired by the development of the Michigan Internet AuctionBot. The original AuctionBot architecture and parametrization was developed jointly by myself, Bill Walsh, Brian Joh, and Michael Wellman. Kevin O'Malley took over development and maintenance of the AuctionBot system, and without his help I would still be years from completing this thesis. Many other students and staff have contributed significantly to the project, including Chris Wong, Stuart Blacklock, Jonathan Mayer, Jeff Kopmanis, Tracy Mullen, Terence Kelly, Dan Reeves, and still others.

The administrative staff at the Artificial Intelligence Laboratory deserve credit for shielding students from administrative headaches—and I'm sure my situation generated more than its share of headaches. Thanks to Kathy Dewitt, Cindy Watts, Deb Stark, and Karen Alexa.

Several other young researchers at remote institutions deserve thanks for reading my papers and sharing ideas. In particular, I'd like to thank Wolfram Conen, David Parkes, and Fredrick Ygge.

Finally, I would like to thank the friends I have made during my time at the Artificial Intelligence Lab who I haven't mentioned in any of the above categories: David Pennock, David Pynadath, Mike van Lent, and Bob Wray provided necessary diversions as office-mates or racquetball partners. Karl Kluge, Karsten Nielsen, Peter Weinstein, and Scott Wood were valued lunch companions. I thank these friends, and the many others I don't have space to list, for the enjoyable times I have spent here at Michigan, and I wish them all luck in their future endeavors.

– Pete

July 23, 1999

Table of Contents

Dedication	ii
Acknowledgements	iii
List of Tables	viii
List of Figures	ix
List of Appendices	x
Chapter	
1 Introduction	1
1.1 The Space of Market Designs	3
1.2 Multidimensional Auctions	5
1.3 Organization of this Thesis	5
2 Distributed Resource Allocation	7
2.1 Overview	7
2.2 A General Formulation of Distributed Resource Allocation	8
2.2.1 Mechanisms	10
2.2.2 Price Systems	12
2.2.3 Exchanges	15
2.3 Performance Criteria	15
2.3.1 Efficiency	16
2.3.2 Equilibrium	16
2.3.3 Stability	18
2.3.4 Individual Rationality	18
2.3.5 Convergence	19
2.3.6 Incentive Compatibility	19
2.3.7 Privacy Preservation	20
2.3.8 Computational Costs	20
2.4 Standard Results about Price Systems	20
2.4.1 When Markets Succeed	20
2.4.2 When Markets Fail	21
2.5 Distributed Scheduling: An Example Domain	23

2.6	Summary	25
3	Market Design	26
3.1	Overview	26
3.2	Common Auction Characteristics	27
3.2.1	Classic Auction Types	27
3.2.2	Three Core Activities	28
3.2.3	The Semantics of Bids	29
3.3	The Auction Parameter Space	33
3.3.1	Bidding Rules	34
3.3.2	Clearing Policy	41
3.3.3	Information Revelation	44
3.4	Parametrization of Well-Known Auctions	48
3.5	Conclusion	49
4	One-Dimensional Auctions for Discrete Resources	50
4.1	Overview	50
4.1.1	Matching Functions in the Literature	51
4.1.2	Matching Functions Online	52
4.2	Generalized, One-Dimensional Matching Functions	54
4.2.1	Uniform-Price Matching Functions	54
4.2.2	Discriminatory-Price Matching Functions	55
4.2.3	Comparisons	57
4.3	The 4-HEAP Algorithm	57
4.3.1	Description of the Algorithm	58
4.3.2	Complexity Analysis	60
4.3.3	Discriminatory Price Auctions	61
4.3.4	4-HEAP and Multiple-Unit Auctions	62
4.3.5	Tradeoffs between Operation Costs	62
4.4	Matching Functions with Indivisible Bids	63
4.5	Conclusion	64
5	Multidimensional Auctions for Discrete Resources	65
5.1	Overview	65
5.2	Existing Results	66
5.2.1	Existence of Price Equilibria	66
5.2.2	Generalized Vickrey Auction	66
5.2.3	Combinatorial Auction Designs	68
5.3	Equilibrium Payments	72
5.3.1	Construction	72
5.3.2	Properties	74
5.3.3	Example	78
5.4	The Family of k -Bundle Auctions	79
5.4.1	Sealed-Bid k -Bundle Auctions	79
5.4.2	Ascending k -Bundle Auction (A1BA)	80
5.5	A1BA Simulations with Myopic Agents	82
5.5.1	The Myopic Strategy	82
5.5.2	Design of the Simulation	82

5.5.3	Results of the Simulation	83
5.6	Conclusion	85
6	Multidimensional Auctions for Continuous Resources	86
6.1	Fixed-Point Algorithms	86
6.2	The Partial-Equilibrium Fixed-Point Algorithm for Multidimensional Auc- tions	88
6.2.1	The Partial-Equilibrium Fixed-Point Procedure	89
6.2.2	Relationship to Tatonnement and Centralized Computation	90
6.3	Conclusion	91
7	Summary and Future Work	92
7.1	Summary of Contributions	92
7.2	Future Work	94
Appendix		
A	URLs for Online Auction Sites	97
B	Parametrized Descriptions of Familiar Auctions	98
C	Summary of Notation	100
Bibliography		104

List of Tables

Table

2.1	Two agents with two discrete goods.	22
3.1	An example with four bids.	42
3.2	An example with indivisible bids without an efficient clearing price.	46
3.3	Summary of the parameters and their possible values for divisible-bid, discrete-good auctions.	48
4.1	A sequence of five bids.	55
4.2	Transaction prices for the six matching functions.	57
4.3	In discriminatory auctions, agent 4's transaction price depends on its trading partner.	57
4.4	A set of bids that can lead to arbitrarily bad outcomes when the greedy algorithm is used.	64
5.1	An example without equilibrium prices for individual goods.	66
5.2	An example in which the GVA payments are discriminatory.	68
5.3	An example in which charging bid payments is not an equilibrium.	72
5.4	An example from Bikhchandani & Ostroy.	77
5.5	Third-order equilibrium payments.	77
5.6	An example with three agents.	78
5.7	The assignment subproblem.	78
5.8	Equilibrium payments.	78
5.9	An example with multiple equilibrium bundle payment vectors.	80
5.10	Distribution of (normalized) solutions when $\beta = 1.5$	83
5.11	An example in which it is in agent 1's interest to not bid on A.	84
B.1	Parameter choices that describe three online auctions.	98
B.2	Parameter choices that describe three classic auctions.	99
B.3	Parameter choices that describe some other interesting auctions.	99

List of Figures

Figure

2.1	Possible partitions of a mediated mechanism for four resources.	13
2.2	An example distributed scheduling problem.	24
3.1	A classification of classic auction types.	28
3.2	Examples of continuous bids that are (a) monotone, and (b) nonmonotone in prices. The shaded area in (a) indicates the correspondence when the bid is divisible.	30
3.3	Examples of discrete bids that are (a) monotone and divisible, (b) monotone but not divisible, (c) not monotone.	31
3.4	Iso-payment curves on a lattice.	32
3.5	Nonmonotone iso-payment curves on a lattice.	33
3.6	Examples of continuous bids that are superior, inferior, and mixed with respect to the original bid.	37
3.7	The four price-space bids from Figure 3.6 mapped into payment space. . .	38
4.1	Schematic diagram of bids arranged in the four heaps.	59
4.2	Pseudocode for receiving a new sell bid.	60
5.1	The space of possible equilibrium payments for the example in Table 5.6. The dashed line indicates the range of k -bundle prices.	79

List of Appendices

Appendix

A	URLs for Online Auction Sites	97
B	Parametrized Descriptions of Familiar Auctions	98
C	Summary of Notation	100

Chapter 1

Introduction

Computer science has drawn a great deal of inspiration from microeconomic theory in recent years. In hindsight, this seems like a natural collaboration given that economics is the study of the allocation of scarce resources—a description that characterizes many computer science problems. The influence of economic theory is visible on at least three fronts:

- **Multiagent systems** (MAS) describes the branch of artificial intelligence research focused on systems of interacting software agents. Researchers in this field have recognized that *game theory* is often a useful framework in which to analyze systems of self-interested agents (Rosenschein and Zlotkin, 1994).
- **Market-oriented programming** is an approach to multiagent systems in which activities are defined by the resources they consume, and agents negotiate for resources in market mechanisms (Wellman, 1993; Ygge, 1998).
- **Electronic commerce** has rapidly become a significant force in the global economy. The reduced costs of communication made possible by a pervasive network promise to bring society closer to the idealized *frictionless* economy of economists' dreams.

In all three of these situations, computer scientists often gain leverage on their problems from the appropriate application of the tools of economic analysis. At the same time, economists have much to gain by collaborating with computer scientists. Many of the problems that arise in computer science applications violate assumptions that are fundamental to many theoretical results. The new problem domains motivate research on less developed aspects of economic theory. In addition, the computational power of modern computers and the broad reach of the Internet make possible solutions that were impractical a decade ago.

The focus of this dissertation is market mechanisms in general, and auctions in particular. Auctions are the infrastructure that support negotiation in any market-based approach, whether it be internal to a computational system, or a component in the automation of commercial activity.

The importance of auctions in the digital economy is evidenced by their manifest popularity on the Internet. Following a reported \$500 million in retail sales by online auctions in 1997, Forrester Research projected that business-to-business transactions through online auctions would total \$8.7 billion in 1998. In fact, there are an estimated 1,000 electronic “exchanges” online today, and that number is expected to grow to 10,000 within a year (Dalton, 1999).

While the consumer oriented auctions tend to use simple variations of the traditional English auction, in the business-to-business arena auctions are likely to be significantly more complex. One reason is that the new exchanges need to be integrated into existing business practices, and be sensitive to existing relationships between firms. In addition, these auctions tend to be for higher stakes—thousands of reams of office paper, hundreds of tons of steel, etc. In some cases, the resources being traded are inputs to manufacturing processes, and their usefulness depends upon the availability of other resources, which are negotiated on other exchanges. Finally, many of these business-to-business exchanges will have multiple buyers and sellers.

The majority of these exchanges will be designed by people with computer science and business backgrounds. These entrepreneurs are asking the research community for a “cookbook”, indexed by the application features, in which they can look up an auction mechanism that satisfies their design goals. I argue that a goal of mechanism design research should be to construct such a cookbook. Although it is unlikely we will identify an appropriate auction design for every problem specification, the act of making this goal more explicit will add valuable structure to the research field.

The traditional approach in mechanism design research is to focus on specific problem types, and develop auctions that “solve” that problem. Economists have a deep understanding of several fundamental auction mechanisms, and the conditions under which they succeed and fail. However, analytical results are currently tractable for only relatively simple protocols. Moreover, the theory often does not address practical issues, such as the computational costs, the perceived fairness, reputation effects, and the process by

which an auction attracts participants.¹ Thus, it is likely that the range of designs used in practice will be considerably broader than auction theorists have studied.

In this dissertation, I present a framework that supports the mechanism designers task of mapping problem descriptions and design goals to auction types. This dissertation suggests an enumeration of the space of possible auctions. The particular approach I propose is to identify the base components of auctions, and examine the manner in which they can be combined to produce particular designs.

This approach has the additional benefit that it helps to identify holes in our coverage of the solution space, and focuses our attention on components which need further development. In particular, Chapters 5 and 6 of this thesis propose innovative components to fill gaps in the space of multidimensional auction designs.

The particular contributions of this thesis can be organized into two categories. The first category is the formal characterization of the space of market designs. The second category is the development of multidimensional auctions to fill gaps in the design space.

1.1 The Space of Market Designs

My description of the market design space is the result of the convergence of two threads in my research. The first is the development of the Michigan Internet AuctionBot, a highly configurable auction server that supports interactions with both humans (via a web interface) and software agents (via an agent programming interface).

During the construction of the AuctionBot, I advocated the view that auctions can be parametrized. This approach greatly facilitated the development of the AuctionBot, and resulted in a server capable of supporting millions of different auction types. A broader adoption of the parametrized view would benefit the research and industrial communities in several ways. First, parametrization isolates the common features of auctions, and, by identifying reusable components, greatly facilitates the construction of flexible auction servers. This flexibility will enable online exchanges to quickly adapt the auction infrastructure to the rapidly evolving world of electronic commerce. In addition, it is widely anticipated that “shopping agents”—programs that search for items across multiple ven-

¹For example, online consumer auction sites use a variant of the English auction in which the auction closes at the later of a fixed time and a period of bidding inactivity. Typically, the fixed time is several days to a week after the inception of the auction. This period of time serves to attract bidders—a necessary feature in a distributed environment like the Internet. Because bidding is allowed during the startup period, it must be considered part of the game.

dor sites (examples include BargainFinder, Jango (Doorenbos et al., 1997), and Amazon’s “Shop the Web”)²—will eventually be extended to interact through auctions (Guttman et al., 1998). Parametrization provides the basis for an auction description language that enables software agents to determine the rules of the auction before they synthesize a strategy and begin negotiation. Finally, as mentioned above, parametrization of auction rules enumerates the design space, and, in the process, highlights regions whose theoretical properties have not been investigated.

The second motivation behind this description of the design space is the result of investigations of various market designs to particular applications, such as the distributed scheduling problem discussed in Section 2.5. These investigations confirmed our belief that there are often a range of possible market designs, and that elements in this range make different tradeoffs between various desirable economic and computational properties.

Although particular points in the range have been thoroughly studied by economists, I am not aware of any research that has developed a consistent terminology for describing market structure. In this thesis, I present such a structured view, the cornerstone of which are *multidimensional auctions*. Multidimensional auctions accept bids on, and compute the allocation of, several resources at the same time.

This thesis combines the structural perspective of markets with the parametrization of auctions into a specification of the market design space that subsumes and extends previous such parametrizations. In particular, this thesis:

1. Develops a structural description that makes precise the range of market designs. The structural description allows resources to be aggregated into multidimensional auctions.
2. Develops an auction parametrization that supports the development of software agents and flexible auction servers, and covers a large percentage of common auction types.
3. Describes the 4-HEAP algorithm, a computationally efficient algorithm for managing bids in one-dimensional, iterative auctions for discrete goods.

The most significant of these contributions is the parametrization. It is defined in the context of multidimensional auctions and very general definitions of bid semantics. In addition to being very broad, it achieves a high level of orthogonality among parameters.

²URLs for all websites mentioned in this dissertation are located in Appendix A

Thus, the vast majority of the combinations of parameter values define meaningful auctions. In the few cases where parameters are not fully orthogonal, I define parallel version of the parameter for each value of the parameter on which it depends.

1.2 Multidimensional Auctions

The approach outlined above presumes the existence of appropriate multidimensional auctions. Historically, microeconomics has focused on the two extremes: fully separated or fully aggregated markets. Only recently have theorists begun to examine auction designs that could be used in the market structure defined herein.

This thesis contributes to the design of such multidimensional auctions. I present novel multidimensional auctions for both continuous and discrete resources. The latter is particularly interesting because the allocation of heterogeneous, discrete resources has become a major focus of research in auction theory.

The primary contributions of this thesis in the area of multidimensional auctions are:

1. Proof of the existence of payment equilibria that support the efficient allocation of bundles for general agent preferences.
2. Introduction of the A1BA, an ascending combinatorial auction which applies the algorithm for constructing equilibrium payments to bids. To my knowledge, A1BA is the first iterative auction that permits bundle bidding and guarantees that the final “payments” satisfy a particular equilibrium concept.
3. Proposal for an adaptation of fixed point algorithms that allows them to be used in multidimensional auctions for continuous resources. This algorithm will be a key component of future investigations of the convergence properties of market designs in which the complementary resources are clustered.

1.3 Organization of this Thesis

In Chapter 2, I present the formal model used in this thesis. I introduce a structural description of negotiation mechanisms in which (multidimensional) mediators are the structural elements. I also list standard economic and game theory desiderata for evaluating proposed designs, and present a distributed scheduling domain as a sample problem.

Chapter 3 identifies and generalizes rules found in common auctions. It begins with

an overview of classic auction types, and then defines bids as a correspondence between quantities of resources and transfers of money. The bulk of the chapter parametrizes orthogonal auction rules in the context of multidimensional auctions. In Appendix B, I show how these rules can be combined to describe standard auctions found in the literature and in practice.

Chapters 4-6 focus on a key element of the auction rules—the matching function. Chapter 4’s purview is one-dimensional matching functions for discrete resources. I define generalized versions of well known matching policies, and present the 4-HEAP algorithm, a computationally effective algorithm for managing iterative, one-dimensional auctions.

The focus of Chapter 5 is multidimensional auctions for discrete resources. I show that, when we forgo linear pricing and instead associate net payments with bundles, a form of equilibrium always exists that supports the socially-preferred allocation in one-sided auctions. I use the method of constructing equilibrium prices in an iterative, ascending auction, and report on a simulation in which agents follow a myopic bidding strategy. These results provide insights into the strengths and weaknesses of the auction design.

Chapter 6 briefly discusses how standard algorithms for computing equilibrium when resources are continuous in nature can be adapted for multidimensional auctions. As an example, I present an enhancement that allows fixed point algorithms to be used to solve for the partial equilibrium prices in a multidimensional auction.

Finally, Chapter 7 summarizes the contributions of the thesis and discusses directions for future work.

Chapter 2

Distributed Resource Allocation

2.1 Overview

Distributed resource allocation problems are characterized by *self-interested agents* and *scarce resources*. An agent derives value, or *utility*, from owning or consuming a subset of the resources. The agent's utility function is known only to the agent. Thus, the information needed to solve the global optimization problem is distributed among the agents in the form of their utility function.

Agents are endowed with some of the resources available in the system. Often, they can improve their overall utility by exchanging resources with other agents. Computing these resource exchanges is the business of an *allocation mechanism*. An allocation mechanism defines the set of messages that are allowed as part of the negotiation process, and the allocations that result.

Mechanisms can be *mediated* or *unmediated*. In an unmediated mechanism, agents send messages directly to each other. For example, a Saturday morning farmers' market is an unmediated mechanism—messages take the form of offers of money for produce, and are communicated directly between individual participants. A mediator, on the other hand, sits between the negotiating parties and controls the flow of information. The agents communicate with the mediator, who reveals information about the state of the negotiation according to its internal rules. The NYSE, for example, has market *specialists* who receive buy and sell offers from the floor and follow a specific set of rules to determine which offers trade and at what price.

Both of these examples involve the use of *money* to facilitate the exchanges. Money is commonly used in human economies to measure the relative worth of resources, and to store value between transactions. Resource allocation mechanisms that reveal information

in the form of relative values are called *price systems*. During the last decade, price systems have been increasingly used in multiagent systems work (Clearwater, 1995; Mullen and Wellman, 1995; Wellman, 1993; Ygge, 1998). This thesis is motivated by that previous research.

The formal characterization of the problem that follows is primarily derived from microeconomics and game theory. Wherever possible, I make use of standard terminology, concepts, and results. However, this thesis is oriented more towards implementation than is usual in economic treatments.

Section 2.2 formalizes the distributed resource allocation problem, and provides a structural description of mediated mechanisms. The model describes the interaction of agents and mediators by the messages they send. Parts of this thesis are focused on auction implementation, and so I define an auction’s behavior as a pair of functions. One function governs the signal the auction generates, and the other determines the exchanges necessary to implement the reallocation.

Section 2.3 reviews the performance criteria used to evaluate mechanisms. A great deal is known about the conditions under which price systems work. Unfortunately, many MAS problems are characterized by discrete resources, an area of economic theory that is not well-developed. Section 2.4 briefly describes standard results in the theory of price systems. Section 2.5 presents a distributed scheduling problem, which serves as an example domain throughout this thesis.

2.2 A General Formulation of Distributed Resource Allocation

A distributed resource allocation problem is characterized by the agents and the resources available. The set of m agents, \mathcal{I} , is indexed by i . The set of n resource types, \mathcal{J} , is indexed by j . Often, I restrict attention to a subset of agents, I , or a subset of resources, J , where $I \subseteq \mathcal{I}$ and $J \subseteq \mathcal{J}$.¹

Let $x_{i,j}$ be the amount of resource j assigned to agent i . The domain of $x_{i,j}$ is X_j , an ordered set whose elements are in \mathfrak{R}_+ . When X_j is the set of non-negative integers, I describe the resource as *discrete*. When X_j is an interval of \mathfrak{R}_+ , I refer to the resource as *continuous*.

Let \mathbf{x}_i be the vector of resources assigned to agent i . The domain of \mathbf{x}_i is $X =$

¹Appendix C summarizes the mathematical notation used in this dissertation.

$\prod_{j \in \mathcal{J}} X_j$ —a notation which allows a mixture of continuous and discrete resources. Let \mathbf{x} be the vector of resource assignments for all agents.

Each agent is endowed with an initial allocation of resources, denoted \mathbf{e}_i , and composed of elements $e_{i,j}$. Let \mathbf{e} denote the vector of m endowment vectors.

The particular details of a distributed resource allocation problem restrict the range of allocations that are *feasible*. At a minimum, feasibility requires that allocations satisfy the *material balance constraint*—resources cannot be added or removed from the system.² That is, for all j ,

$$\sum_{i \in \mathcal{I}} x_{i,j} = \sum_{i \in \mathcal{I}} e_{i,j}.$$

An allocation that satisfies the feasibility constraints is called a *solution*, and denoted \mathbf{f} . The vector of resources allocated to i in the solution \mathbf{f} is denoted by \mathbf{f}_i . Let F denote the set of all feasible solutions. If the only feasibility constraint is material balance, then it follows from the definition that $\mathbf{e} \in F$.

In Chapter 5, I consider the special case where $X = \{0, 1\}^{|\mathcal{J}|}$.³ In this setting, it is often convenient to refer to the resources using set notation. A *bundle*, b , is a subset of resources, $b \subseteq \mathcal{J}$. \mathcal{B} denotes the power set of \mathcal{J} . Let $b(j)$ indicate whether resource j is a component of b : $b(j) = 1$ if j is an element of b , and $b(j) = 0$ otherwise. If for all j , $f_{i,j} = b(j)$, we say that agent i is *assigned* bundle b .

Let agent i 's *net allocation*, \mathbf{z}_i , be the difference between its allocation and its endowment. $\mathbf{z}_i = \mathbf{x}_i - \mathbf{e}_i$. The domain of i 's net allocation is $Z^i = \{\mathbf{x}_i - \mathbf{e}_i \mid \mathbf{x}_i \in X\}$.

An agent's preference relation over all possible combinations of resources is specified by its utility function, u_i . The utility that an agent gets from the allocation \mathbf{x}_i , is $u_i(\mathbf{x}_i)$. Throughout this thesis I assume that u_i is nondecreasing in X .

Let \mathbf{y} , \mathbf{y}^1 , and \mathbf{y}^2 be real-valued vectors. The \wedge operator produces the *meet* of two vectors, defined as the vector whose elements are maximal among the corresponding elements of the original vectors. That is, $\mathbf{y} = \mathbf{y}^1 \wedge \mathbf{y}^2$ if, for all j ,

$$\mathbf{y}_j = \max(\mathbf{y}_j^1, \mathbf{y}_j^2).$$

²When we are discussing distributed resource allocation problems that include *production*—agents have the capability to convert one resource type into another—a more general version of the material balance constraint holds.

³We can often cast multi-unit discrete resource allocation problems as single-unit problems by considering multiple units of an indivisible resource type as distinct, singleton resources.

The complementary operator \vee produces the *join* of two vectors. $\mathbf{y} = \mathbf{y}^1 \vee \mathbf{y}^2$, if for all j ,

$$\mathbf{y}_j = \min(\mathbf{y}_j^1, \mathbf{y}_j^2).$$

A function, η , is *submodular* if, for all \mathbf{y}^1 and \mathbf{y}^2 ,

$$\eta(\mathbf{y}^1) + \eta(\mathbf{y}^2) \geq \eta(\mathbf{y}^1 \wedge \mathbf{y}^2) + \eta(\mathbf{y}^1 \vee \mathbf{y}^2).$$

A function is *supermodular* if the inequality is reversed.

If X contains the join and meet of every pair of its elements, then it is a *lattice*. If X_j is also finite for all j , then X is a *finite lattice*. In Chapter 5, I make use of the fact that \mathcal{B} is a finite lattice.

2.2.1 Mechanisms

A *mechanism* specifies the procedure by which agents can realize a solution. In game-theoretic terms, a mechanism defines the rules of the game. Game-theoretic descriptions specify every agents' space of *strategies* and, for every combination of strategies, each agent's *payoff*. When agents do not know each other's utility functions, distributed resource allocation problems fall in the class of games of *incomplete information*.

Let ψ^i be agent i 's strategy, The set of strategies available to the agent is denoted Ψ^i , and the joint of all agent strategy spaces is denoted Ψ . A mechanism, \mathcal{M} , maps the strategies of the agents, $\boldsymbol{\psi} = (\psi^1, \dots, \psi^m)$, into a solution, $\mathcal{M} : \Psi \rightarrow F$, which is the realization of the agents' payoffs. Let $\boldsymbol{\psi}^{-i}$ denote the set of strategies of all agents other than i . In general, an agent's strategy can be a complex function of its beliefs, other agents actions (or evidence thereof), the rules of the mechanism, and temporal considerations. The game-theoretic description fully captures the environment and provides a framework within which to analyze agent strategies.

When the space of agent strategies is sufficiently large, a game theoretic analysis may not be tractable. One methodology we can apply in such situations is to posit a particular agent behavior and study the performance of the mechanism under that assumption. A *protocol* is the combination of a mechanism and a set of agent behaviors.

Mechanism design has been a subject of study in microeconomics for three decades, inaugurated by Vickrey's pioneering work (Vickrey, 1961). More recently, its applicability has been recognized by researchers in the field of multiagent systems (MAS). I find it useful

to classify the related work in MAS by whether the mechanism is *mediated* or *unmediated*.

This dissertation is focused on mediated mechanisms, but a short review of unmediated mechanisms in multiagent systems is in order. One of the first unmediated, agent-oriented protocols was ContractNet (Davis and Smith, 1983), which was proposed as an approach to distributed problem solving. In ContractNet, agents send messages (requests for subcontracts) directly between one another. Sandholm (1993) extended ContractNet to allow the agents to account for marginal costs when accepting (or decline) contract offers, however the mechanism remained unmediated. The same holds for Kasbah (Chavez and Maes, 1996), an agent-based marketplace built at MIT to facilitate web-based transactions. The game-theoretic interactions studied by Rosenschein and Zlotkin (1994) also fall into this category. In general, unmediated mechanisms do not perform well as the number of agents increases because the number of (potential) bilateral negotiations necessary to find the “best” trades (or contracts) increases exponentially. For this reason, mediated mechanisms have attracted more attention in recent years.

In a mediated mechanism, agents communicate with system components, called mediators, which instantiate formal negotiation rules and control the flow of information. A mediator, α_J , is an element of the mechanism that manages the negotiation over the subset of resources J . The mediator’s *scope*, σ , is that subset of resources under its control. That is, $J = \sigma(\alpha_J)$. The *dimensionality* of α_J is $|J|$.

Agents interact with a mediator by sending and receiving messages. Agent i ’s message to the mediator with scope J is denoted w_i^J . The space of messages admissible to α_J is denoted W^J .

The mediator’s task is to map the agent messages, \mathbf{w} , into net resource exchanges: $\alpha_J : W^J \rightarrow F^J$. In this framework, mediators are not agents—they have no self-interest and merely follow a publically stated algorithm to determine allocations based on messages.⁴

A mediator is *iterative* if it reveals information about messages sent by other agents, as individuals or in aggregate, and allows an agent to submit new messages after receiving this feedback. To be fully general, we should augment messages with time, as in $w_i^J(t)$. However, in this thesis I consider only mediators that consider the most recent messages from each agent, and so stick to the more direct notation, w_i^J .

An iterative mechanism produces a signal $q(t)$ at time t . Once again the simpler notation suffices for the mediators discussed herein, and I will use q to denote the most recent signal produced by the mediator. If all agents receive the same signal, we say the signal

⁴This assumption does not preclude them from collecting revenues, a fact I touch on in Chapter 3.

is *anonymous*. If signals are customized for the agents, we say they are *discriminatory*, and denote them q^i . The component of a mediator that generates q is the *quote function*, designated ρ .⁵ That is,

$$q = \rho(\mathbf{w}). \quad (2.1)$$

In an iterative mechanism, agent i 's strategy ψ^i is conditioned on q (or q^i). Typically, the mechanism's signal provides evidence regarding potential outcomes of the negotiation, in the form of a tentative allocation. Let $\chi_{i,j}(q, w)$ denote the amount of j that q implies agent i would receive had it sent w to the mediator.

A signal is *separating* if all agents can correctly infer their tentative allocation. If q is *noisy*, then $\chi_{i,j}(q, w)$ may not be the allocation the agent would actually receive. In such cases, the agent may be able to infer a distribution over potential allocations, and modify its strategy appropriately.

A mediated mechanism is *partitioned* if each resource type is within the scope of exactly one mediator. Within the class of partitioned mechanisms, there are many possible designs. For instance, at the two structural extremes, we could define $\mathcal{M}^1 \equiv \{\alpha_{\{1, \dots, n\}}\}$ and $\mathcal{M}^n \equiv \{\alpha_{\{1\}}, \dots, \alpha_{\{n\}}\}$. The former has one mediator that manages all of the resources, while the latter has one mediator for each resource. Figure 2.1 depicts the possible partitioned mechanism structures for an allocation problem with four resource types.

2.2.2 Price Systems

The particular mechanisms I am interested in are based on price systems. *Prices* are nonnegative real numbers that determine the exchange rate of resources. Resource j can be exchanged for resource k at the ratio of their prices, p_j/p_k .

When prices are announced as part of the quote, they provide a concise summary of global information that agents can use to make local decisions. In fact, it has been shown that, for convex problems, the *competitive protocol*, in which the mediator announces prices and the agents respond with the quantity of resources they would buy (or sell) at those prices, minimizes the dimensionality of the message space required to determine the efficient allocation (Mount and Reiter, 1974). Moreover, this protocol is uniquely minimal

⁵In principle, ρ can use the entire history of messages to compute q , in which case (2.1) would be adjusted accordingly.

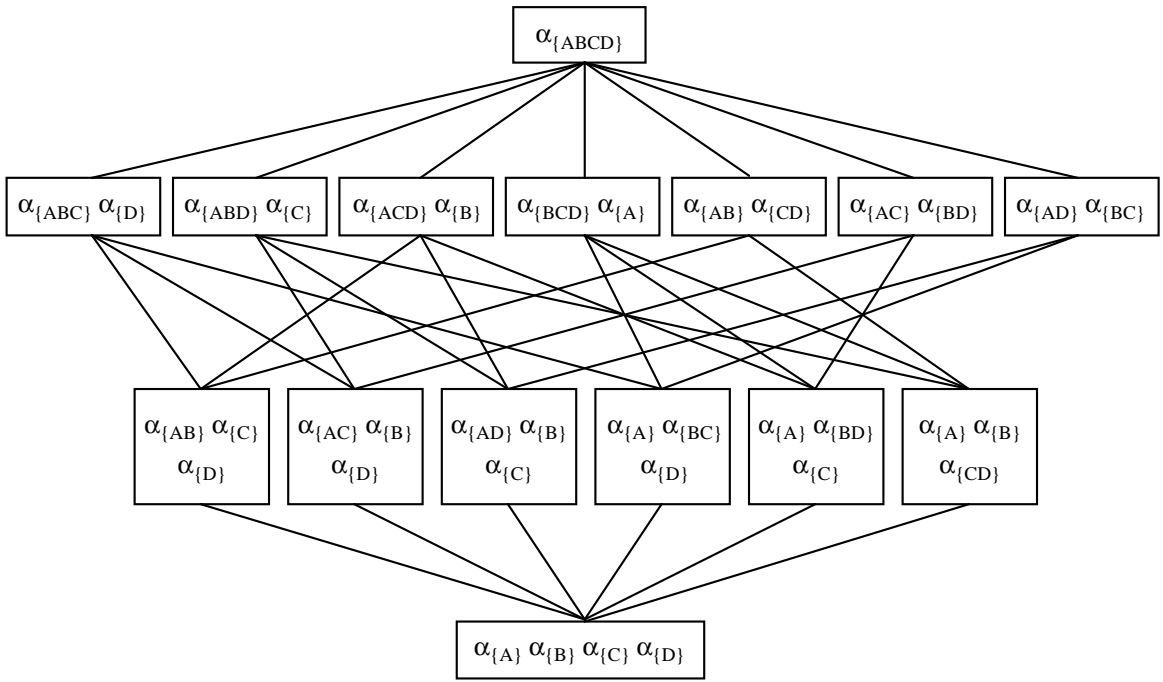


Figure 2.1: Possible partitions of a mediated mechanism for four resources.

(Jordan, 1982).

We can attribute prices to any distributed resource allocation problem simply by designating one resource a *numeraire*, fixing its price at one, and expressing the relative values of other goods in terms of the numeraire. However, it is convenient to introduce a special resource—denoted μ and called *money*—to serve this role. In this thesis, I treat μ separately from the elements of \mathcal{J} .

A common justification for the introduction of money into a system is that it facilitates the transfer of value between mediators, across time, or to aspects of the economy beyond the consideration of the model. Money serves this role in human economies, and is therefore a natural element of any e-commerce application.

Because we are not treating money as an element of \mathcal{J} , an agent's utility must account for it explicitly, and is denoted $u_i(\mathbf{x}, \mu)$. Throughout most of this thesis, I assume that agents' utility functions are *quasilinear* with respect to money. That is, agent i 's utility function is of the form

$$u_i(\mathbf{x}, \mu) = v_i(\mathbf{x}) + \mu.$$

This permits an agent to express its value for any \mathbf{x} directly in units of money. Agent i 's *valuation*, or *willingness to pay*, for \mathbf{x} is $v_i(\mathbf{x})$. Let $f_{i,\mu}$ be agent i 's monetary allocation

in the solution \mathbf{f} . Similarly, let $e_{i,\mu}$ denote i 's original endowment of money. When we require that $f_{i,\mu} \geq 0$, we say that agents are *budget constrained*.

Agent i 's *surplus*, s_i , is the improvement in utility that it experiences by moving from \mathbf{e}_i to \mathbf{f}_i ,

$$s_i = u_i(\mathbf{f}_i, f_{i,\mu}) - u_i(\mathbf{e}_i, e_{i,\mu}),$$

which, given quasilinear utility, can be written,

$$s_i = [v_i(\mathbf{f}_i) - v_i(\mathbf{e}_i)] + [f_{i,\mu} - e_{i,\mu}].$$

The total surplus, s , is the aggregate improvement created by the solution, \mathbf{f} ,

$$s = \sum_i s_i = \sum_i [u_i(\mathbf{f}_i, f_{i,\mu}) - u_i(\mathbf{e}_i, e_{i,\mu})].$$

When utility functions are quasilinear, we can write

$$s = \sum_i [v_i(\mathbf{f}_i) - v_i(\mathbf{e}_i)] + \sum_i [f_{i,\mu} - e_{i,\mu}].$$

Money is conserved, so $\sum_i f_{i,\mu} = \sum_i e_{i,\mu}$. Thus we can simplify the above expression,

$$s = \sum_i [v_i(\mathbf{f}_i) - v_i(\mathbf{e}_i)]. \quad (2.2)$$

Because the value of the initial allocation is fixed, an allocation that maximizes the total surplus also maximizes the total value,

$$V(\mathbf{f}) = \sum_i v_i(\mathbf{f}_i). \quad (2.3)$$

Let $\mathbf{z}_i \equiv \mathbf{f}_i - \mathbf{e}_i$, represent the change in allocation that agent i experiences between its endowment and the solution \mathbf{f} . Let p_j be the price of j expressed in monetary units, and \mathbf{p} denote the vector of all prices. The net *payment*, π^i , by agent i is simply the product of the prices and the change in the quantities of resources in i 's allocation.

$$\pi^i = \mathbf{p} \cdot \mathbf{z}_i. \quad (2.4)$$

Notice that, whereas prices are nonnegative, net payments can be positive or negative. A

negative payment indicates that money is flowing *to* the agent.

Nonlinear prices relax (2.4). A complete specification of nonlinear prices associates a payment, $\pi_{\mathbf{z}}$, for every allocation vector, \mathbf{z} . The aggregation of these payments forms a *payment lattice*, $\boldsymbol{\pi}$. We require that payments be nondecreasing in Z , and the payment associated with $\mathbf{z}_i = 0$ be zero. Like prices, payments can be either anonymous or discriminatory.

Implementing nonlinear prices requires a means to enforce them. For example, if the price of buying two units is more than twice the price of buying one unit, an agent that wants two units has an incentive to purchase the two units separately (perhaps under two different names). Likewise, if the price of two units is less than twice the price of one unit, two agents who want one unit each would prefer to form a purchasing cartel, buy a two-unit bundle, and divide it later. I do not deal with the enforcement issue in this thesis. In the portions of this thesis that deal with nonlinear prices, I implicitly assume there is an outside agency able to ensure that agents pay the payment associated with their net allocation.

2.2.3 Exchanges

One allocation of resources is transformed into another by a sequence of *exchanges*, Θ . The individual exchange θ transforms \mathbf{f} into $\hat{\mathbf{f}}$, expressed $\hat{\mathbf{f}} = \theta(\mathbf{f})$. Exchanges must conserve feasibility—if $\mathbf{f} \in F$ then $\theta(\mathbf{f}) \in F$. I introduce more precise notation for the special case of exchanges that involve only two agents (bilateral trades). A bilateral trade is defined by the tuple $\langle i, \mathbf{z}, h, \pi \rangle$, where agent i sells \mathbf{z} to agent h for a net payment π .

The set of agents that participate in the exchanges Θ is designated \mathcal{I}_Θ . The set of agents that do not participate in an exchange are designated $\mathcal{I}_{-\Theta}$.

Let γ be a procedure that determines a set of exchanges as a function of messages: $\Theta = \gamma(\mathbf{w})$. γ is called a *matching function*, and is a key component of a mediator. Chapters 4–6 discuss matching functions for various types of resources and mediators.

2.3 Performance Criteria

The following list of performance criteria for solutions and mechanisms is compiled from standard treatments of game theory and mechanism design theory (Campbell, 1987; Mas-Colell et al., 1995; Fudenberg and Tirole, 1996).

2.3.1 Efficiency

The measure of solution quality is *efficiency*. For the case of noncooperative agents with unrestricted utility functions, the appropriate measure is *Pareto efficiency*. A solution is Pareto efficient if there is no other solution in which some agent could improve its utility without making another agent worse off. Formally, \mathbf{f} is Pareto efficient if there is no solution $\hat{\mathbf{f}}$ and agent i for which $u_i(\hat{\mathbf{f}}_i) > u_i(\mathbf{f}_i)$ and for all agents h , $u_h(\hat{\mathbf{f}}_h) \geq u_h(\mathbf{f}_h)$.

The quasilinear utility assumption enables us to effectively transfer utility among agents, and allows us to quantify the social value, or *welfare*, of an allocation in monetary units. By the argument above, the efficient solution maximizes (2.3):

$$\mathbf{f}^* = \arg \max_{\mathbf{f}} \sum_{i \in \mathcal{I}} v_i(\mathbf{f}_i).$$

The value of the optimal allocation is $V(\mathbf{f}^*)$.

2.3.2 Equilibrium

An equilibrium is a state from which no agent wishes to deviate. There are a variety of equilibrium concepts, and each is appropriate under different assumptions of agent knowledge and behavior. An investigation of equilibrium properties often asks the following questions:

- Do equilibria exist? If so, how many?
- Are equilibrium allocations efficient?

The Prisoner's Dilemma is a classic game in which the Nash equilibrium, in which both players defect, is not a Pareto efficient solution (see, for example, Fudenberg and Tirole (1996)).

The following standard equilibrium concepts are used in this thesis.

Nash Equilibrium. A Nash equilibrium is a vector of strategies, $\boldsymbol{\psi}$, such that, for all i , $\hat{\boldsymbol{\psi}}^i$,

$$u_i(\mathcal{M}(\boldsymbol{\psi})) \geq u_i(\mathcal{M}(\boldsymbol{\psi}^{-i}, \hat{\boldsymbol{\psi}}^i)).$$

That is, given the other agents' strategies, agent i cannot receive any more utility

from a strategy other than ψ^i . Note that in order to compute its Nash strategies, the agent must know the other agents' payoffs (i.e. their utility functions) as well as \mathcal{M} .

A *mixed strategy* is a probability distribution over strategies. Let $\Pr_i(\psi^i)$ be the probability that i plays the strategy ψ^i . $\sum_{\psi^i \in \Psi^i} \Pr_i(\psi^i) = 1$. A primary result of game theory is that every finite strategic-form game has a mixed-strategy equilibrium (Nash, 1950).

Bayes-Nash Equilibrium. When an agent's knowledge is incomplete, we generally assume that it has some belief, expressed in terms of probabilities, regarding the possible types of other agents (where a type specifies an agents strategies and payoffs). A *Bayes-Nash Equilibrium* is a set of (possibly mixed) strategies from which no agent would deviate given its beliefs about the distribution of other agent types.

Dominant Strategy Equilibrium. A strategy is dominant if it maximizes an agent's utility regardless of the strategies played by other agents. In a dominant strategy equilibrium all agents are playing dominant strategies.

Price Equilibrium. In a price equilibrium, agents behave as if prices are determined exogenously and are affected by the agent's own messages. Under quasilinear preferences, a solution, \mathbf{f} , and price vector, \mathbf{p} , form a price equilibrium if, for all i ,

$$v_i(\mathbf{f}_i) - \mathbf{p} \cdot \mathbf{f}_i = \max_{\mathbf{x} \in X} [v_i(\mathbf{x}) - \mathbf{p} \cdot \mathbf{x}]. \quad (2.5)$$

If (2.5) holds for a price vector, \mathbf{p} , we say \mathbf{p} *supports* the equilibrium.

Notice that (2.5) is defined such that it is independent of the agent's endowment (i.e. it is in terms of \mathbf{x} rather than \mathbf{z}). If \mathbf{f}_i maximizes i 's utility given prices \mathbf{p} , then it does so regardless of i 's initial allocation. It is as if the agent sold its endowment to the system, and then repurchased it.

Payment Equilibrium. I adopt a relaxation of price equilibrium for mechanisms with nonlinear prices. This form of equilibrium will be used in Chapter 5 in situations where price equilibria do not exist. Let \mathbf{z}_i be agent i 's net allocation in \mathbf{f} . Under quasilinear preferences, the solution, \mathbf{f} , and payment lattice, $\boldsymbol{\pi}$, form a payment

equilibrium if, for all i ,

$$v_i(\mathbf{z}_i + \mathbf{e}_i) - \pi_{\mathbf{z}_i} = \max_{\mathbf{z} \in Z} [v_i(\mathbf{z} + \mathbf{e}_i) - \pi_{\mathbf{z}}]. \quad (2.6)$$

If (2.6) holds for a payment lattice, $\boldsymbol{\pi}$, we say $\boldsymbol{\pi}$ *supports* the equilibrium.

There is an important distinction between the payment and price equilibrium concepts. (2.5) can be transformed into (2.6) by adding the cost of the agent's endowments to both sides. However, the inverse transformation does not hold because, in the nonlinear case, $\pi_{\mathbf{z}_i} + \pi_{\mathbf{e}_i} \neq \pi_{\mathbf{z}_i + \mathbf{e}_i}$.

2.3.3 Stability

We say a solution is stable, or in the *core*, if there is no subset of agents that could have done better by exchanging their endowments outside of the mechanism (Mas-Colell et al., 1995; Tesler, 1994). Let $I \subset \mathcal{I}$, and \mathbf{g}^I be the optimal solution to the subproblem of allocating the endowments of I among I (assuming quasilinearity). \mathbf{f} is in the core if there does not exist some subset of agents, I , such that for all $i \in I$,

$$u_i(\mathbf{g}_i^I, g_{i,\mu}^I) \geq u_i(\mathbf{f}_i, f_{i,\mu}).$$

If a solution is in the core, then it is Pareto efficient, although the converse is not true.

2.3.4 Individual Rationality

A mechanism is *individually rational* if it produces solutions that are stable with respect to coalitions of size one. In other words, no agent would decline to participate in the reallocation because it was better off with its initial allocation. This rules out, for instance, the allocation that gives all of the resources to a single agent (assuming that agent did not have all of them in the first place). It would be irrational for any agent other than the beneficiary to participate in this mechanism.

In some cases we can guarantee only that the expected payoff to an agent is positive, *ex ante* (Mas-Colell et al., 1995). Since any particular allocation may make an agent worse off, we must construct social instruments to enforce the allocation.

2.3.5 Convergence

When equilibrium exists, we would like to determine whether a particular protocol is guaranteed to converge to it. If such a guarantee does not exist, we would like to determine whether convergence can be established for useful subclasses of the general problem.

2.3.6 Incentive Compatibility

The following description is summarized from Mas-Colell, et al. (Mas-Colell et al., 1995). A social choice function is a mapping of agent *types* to allocations, where an agent's type is simply its utility function. A mechanism is a *direct revelation mechanism* if each agents' messages states a type. The *revelation principle* states that for every equilibrium of some indirect mechanism, there is a direct revelation mechanism in which it is an equilibrium for agents to truthfully report their types. A social choice function is *incentive compatible* if it is an equilibrium for agents to truthfully report their type in the corresponding direct revelation mechanism.

The precise statement of the revelation principle and incentive compatibility depends on the equilibrium concept employed. Two important specializations are the revelation principle for dominant strategy equilibrium and the revelation principle for Bayesian Nash equilibrium (Mas-Colell et al., 1995, Chapter 23). A principal advantage of a dominant strategy mechanism is that an agent's decision depends on only its local information, and it gains no advantage by expending effort to model other agents. Unfortunately, the Gibbard-Satterthwaite Theorem shows that, for a very general class of social choice functions, the only mechanisms capable of implementing solutions in dominant strategies are dictatorial (Gibbard, 1973; Satterthwaite, 1975).

The set of *Bayesian incentive compatible* mechanisms is larger, and includes all dominant strategy incentive compatible mechanisms. In fact, when agents' utilities are quasi-linear and independent, there always exists at least one direct revelation mechanism that is Bayesian incentive compatible and individually rational. However, even if the mechanism is Bayesian incentive compatible, an agent may still benefit by modeling the other agents. A Bayes-Nash equilibrium states that an agent cannot improve its expected utility given its beliefs of other agents types. In an iterative mechanism, an agent can use the revealed information to update those beliefs.

2.3.7 Privacy Preservation

Privacy is measured with respect to personal information. Mechanisms that allow agents to learn other's private information do not respect privacy. If a mechanism preserves privacy, it may reduce the ability of agents to gain strategic advantage over others. In some practical applications, the participants may not trust the auctioneer, and may be resistant to revealing private information to the auction that is not directly relevant to the allocation.

2.3.8 Computational Costs

To analyze the computational costs associated with a mechanism, we must quantify the computational complexity of both the agents' and the mediators' individual optimization problems, the size and number of messages, and the number of iterations. System designers need to understand the costs and benefits of decentralization, and the tradeoffs among computational effort, communication, and solution quality.

2.4 Standard Results about Price Systems

2.4.1 When Markets Succeed

General equilibrium theory provides some very strong results regarding solutions in fairly broad classes of allocation problems. In particular, the First Welfare Theorem states that, when resources are continuous and a market exists in every resource an agent cares about, a price equilibrium is Pareto efficient. The Second Welfare Theorem states that, under the assumption of convex preferences, for any Pareto efficient allocation there is some set of prices and reallocation of initial endowments for which the allocation is a price equilibrium.

Tatonnement is the price adjustment protocol originally proposed by Léon Walras in 1874. The auctioneer announces a set of prices and the agents, which are assumed to be price-taking, respond with their demands. The auctioneer incrementally adjusts prices in an attempt to balance aggregate demand. When *gross substitutability*—characterized as the condition where a price increase in one resource does not cause a decrease in the aggregate demand of another—holds among the resources, the protocol is guaranteed to converge to equilibrium.

Very useful results have also been established for the allocation of discrete resources,

albeit most of the precise results in auction theory concern the allocation of a single unit. For example, the *Revenue Equivalence Theorem* states that, when agents are risk neutral, and have independent valuations drawn from the same continuous distribution, all four basic auction types (English, Dutch, and the first- and second-price sealed bid) produce the same revenue to the seller (see McAfee and McMillan (1987) for a review). More general problems have been studied, but the theory for multi-unit allocation problems is significantly less mature. In fact, results are often of a negative nature, showing ways in which the standard price mechanisms fail. Myerson and Satterthwaite (1983) show that there does not exist any bargaining mechanism that is individually rational, efficient, Bayesian incentive compatible for both buyers and sellers, and does not require outside subsidies. Other examples of market failures are presented in the next section.

2.4.2 When Markets Fail

Many distributed resource allocation problems that arise in computer science do not satisfy all of the assumptions of the Walrasian protocol. In particular, we are often faced with resources that are discrete in nature, and utility functions that exhibit complementarities among resources. *Complementarity* is the opposite of substitutability, and is characterized by a decrease in the aggregate demand of one good when the price of another good increases. Supermodularity of utility functions is a sufficient condition for resources to exhibit *complementary* relationships (Topkis, 1998).

Markets fail to reach price equilibrium for two basic reasons: either no equilibrium exists or the protocol fails to converge to one. Failures of convergence are problems with the protocol's ability to search through the space of possible allocations (often due to misaligned incentives). Failures of existence are actually due to formulating the problem as a price system. Note that an existence failure does not imply that there is no efficient solution—for any finite problem there is always at least one efficient solution. Rather, it means the solution is not within the restricted solution space defined by the price system. As Section 2.5 illustrates, existence failures are not merely of theoretical interest—they occur in a broad class of common distributed scheduling problems.

Failures of Convergence

Tatonnement is, essentially, a distributed gradient search technique. As such, it is vulnerable to saddle points, plateaus, and local maxima. Scarf (1960) constructed an example of

	A	B	AB
Agent 1	0	0	3
Agent 2	2	2	2

Table 2.1: Two agents with two discrete goods.

three agents and three resources for which a price equilibrium exists but price adjustment schemes fail to converge. The utility functions of the three agents are identical functions of three parameters. Agent 1 has parameters $\langle \alpha, \beta, \gamma \rangle$, agent 2 has $\langle \beta, \gamma, \alpha \rangle$, and agent 3 has $\langle \gamma, \alpha, \beta \rangle$.⁶ If the agents all start with endowments cycled in a similar fashion to their utility parameters,⁷ then an equilibrium is one where the prices of all the resources are equal. However, because gross substitutability is violated, unless the initial set of prices is in equilibrium, the price adjustment process will simply cycle around the equilibrium point without getting any closer.

Failures in Existence

The Second Welfare Theorem requires that aggregate excess demand functions be continuous, strictly convex, and strongly monotone. When these conditions are violated, the existence of price equilibrium is no longer guaranteed. Two common nonconvexities are discreteness and increasing marginal returns. Increasing marginal returns and complementarities are both forms of supermodular preferences.

Complementarities and discreteness prevent the existence of equilibrium prices in a very simple example presented by McAfee and McMillan (1996). The example consists of two goods, A and B, and two agents with quasilinear utility functions. Agent 1 has no value for either good alone, and values the combination of goods (AB) at \$3. Agent 2 values either good at \$2, but gets no added value from getting both. Table 2.1 shows the valuations for the two agents over all possible combinations. The efficient solution is to allocate both goods to agent 1. However there is no set of prices for the individual goods that supports this outcome. In order to exclude agent 2, the individual prices of each good need to be set above \$2. However, this would put the price of the pair above agent 1's valuation.

As the next section shows, discreteness and complementarities are not uncommon in

⁶Scarf presents several example functions, the simplest of which is $u(x, y, z) = \min(x, y)$.

⁷This is not a necessary restriction. Differing endowments will change the shape of the cycle but will not guarantee convergence.

distributed resource allocation problems.

2.5 Distributed Scheduling: An Example Domain

The example in Table 2.1, despite its simplicity, is an example of a very large and important class of scheduling problems. These scheduling problems illustrate many of the issues presented in this chapter.

Consider a (much simplified) factory with limited capacity and a set of agents who want jobs performed in the factory. A job is described by its length, λ , and deadline, δ . There are at least two ways to define the resources for this problem: either the resources are jobs and the agents take bids from the factory to perform the job, or the resources are time slots which the factory tries to sell to the agents. The two formulations are symmetric—what is at issue is whether we create a market in jobs or time slots. For this discussion, we choose the latter formulation.

If an agent can purchase enough time slots to complete its job before the deadline, it receives positive value. If it fails to get enough slots, it gets zero value. Thus, when an agent's job requires more than one time slot, its utility function over time slots exhibits complementarity.

An example problem is depicted in Figure 2.2. In this illustration, there are four agents competing for eight one-hour time slots. Each agent has one job it wants completed, with the values, lengths, and deadlines shown. Each time slot is labeled by its completion time, and augmented with a price, and the set of prices support the allocation indicated by lines between agents and time slots. In this example, there is an optimal allocation supported by a price equilibrium.

We have investigated the application of some basic economic theory to this scheduling domain (Walsh et al., 1998; Wellman et al., 1999). In situations where agents have job lengths of one time slot, equilibrium exists and we have analyzed protocols that converge to within some tolerance of the efficient allocation.

When larger job lengths are allowed, the guarantees no longer hold. The example in Table 2.1 could be interpreted as a scheduling problem in which agent 1 had a job of length two and deadline of 10:00, and agent 2 has a job length of one and deadline of 10:00. One approach is to have a market in resources defined by length-deadline pairs. The transformed resources corresponding to the example are $\hat{\mathcal{J}} = \{\text{one slot by 9:00, one slot by 10:00, two slots by 10:00}\}$. The mechanism must ensure that the allocated

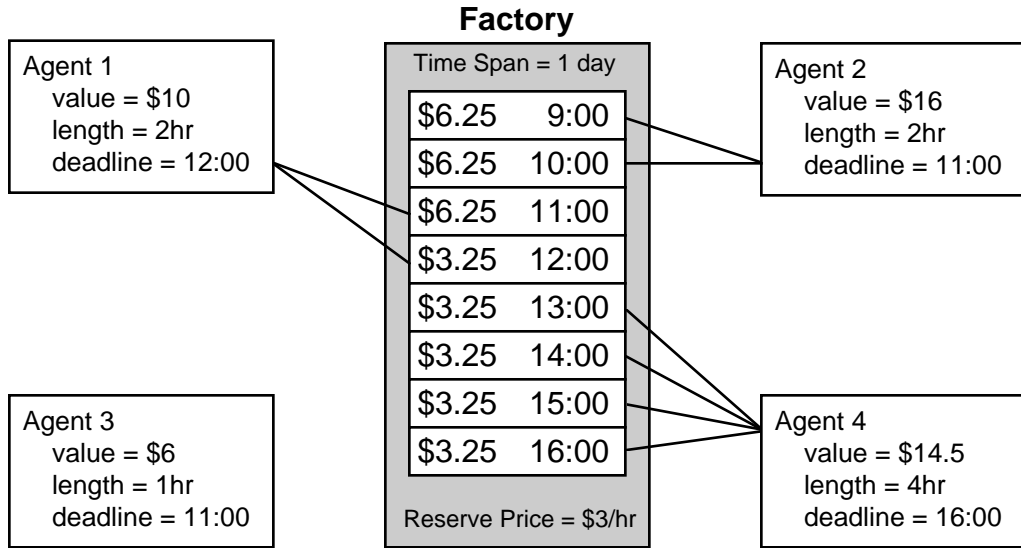


Figure 2.2: An example distributed scheduling problem.

length-deadline pairs satisfy the feasibility constraints of the original factory—it could not allocated both {one by 9:00} and {two by 10:00}. This transformation of resources allows agents to express their complementary demand for combinations of time slots. We need to specify payments for only $n\lambda_{\max}$ resources, where λ_{\max} is the largest job length, and n is the number of time slots. It is worth noting that price equilibria always exist for this transformed market.

One conclusion drawn from this investigation is that a range of mechanisms are available with different properties and performance characteristics. A goal of this dissertation is to provide a framework within which we can compare the performance characteristics of the various design choices. The following are four example structural designs that follow naturally from the presentation.

1. A separate auction (mediator) is created for each time slot. We would characterize the structure of this mechanism as $\mathcal{M}^n \equiv \{\alpha_{\{1\}}, \dots, \alpha_{\{n\}}\}$.
2. A direct revelation mechanism has the structural form $\mathcal{M}^{\text{DRM}} \equiv \{\alpha_{\{1, \dots, n\}}\}$.
3. An n -dimensional auction is run in which agents are allowed to bid on length-deadline pairs. The auctioneer is responsible for transforming these bids into an allocation that does not overallocate any base time slot. $\mathcal{M}^{\lambda^\delta} \equiv \{\alpha_{\{1, \dots, n\}}\}$. The auction may, for instance, be iterative and announce prices on length-deadline pairs, thus distinguishing it from \mathcal{M}^{DRM} .

4. An auction is run independently for each deadline. Agents bid on the number of time slots they want with a given deadline. In this case, the factory would have to be an active participant, and place bids in each auction in such a way that it is not overcommitted. This design is characterized

$$\mathcal{M}^\delta \equiv \{\alpha_{\delta=1}, \dots, \alpha_{\delta=n}\},$$

where (abusing notation slightly) $\alpha_{\delta=k}$ denotes the auction for packages of time slots that end at or before the k th. This is one possible design that supports the transformation of resources into length-deadline pairs.

2.6 Summary

In this chapter, I have formally defined the distributed resource allocation problem, and a general framework for evaluating its solutions. The rest of the dissertation comprises an in-depth study of particular elements of the price system model. Chapter 3 examines the potential rules that mediators can employ in price systems. In Chapter 4, I examine candidate matching functions for one-dimensional auctions of discrete resources. In Chapters 5 and 6, I present novel matching functions for multidimensional auctions for discrete and continuous resources, respectively.

Chapter 3

Market Design

3.1 Overview

Chapter 2 provides an abstract definition of an allocation mechanism. In this chapter I address, in more depth, the design of market-based allocation mechanisms. A mediator in a market-based mechanism is an *auction*.

In this and subsequent chapters, the messages that agents send to the auction are referred to as *bids*, and express offers to exchange resources for money. Unless stated otherwise, we assume that any agent can be either a buyer or a seller. The messages the auction produces are called *quotes*, and reveal intermediate information regarding the state of the auction, and, indirectly, the other agents' bids.

Many different auction types are discussed in the literature and used in practice. Rather than try to impose a hierarchical classification scheme on them, I propose a structured parametrization of the space of auction rules. This parametrization leads to a concise description of very large number of auctions, some of which have been studied in economic literature, and many of which have not. The general consensus among auction theorists is that no “one-size-fits-all” result is likely—the choice of a best auction for a particular need depends on details of the problem and the designer’s objectives.

This approach is particularly useful for two tasks that computer scientists face: building configurable auction infrastructure, and building software agents to participate in online markets.

The parametrization presented in this chapter evolved from the design of the Michigan Internet AuctionBot (Wurman et al., 1998b). Parametrization supported the original implementation, and facilitates the ongoing extensions, by separating the programmatic aspects of a particular family of auctions from the operational choices. For example, in

the Michigan Internet AuctionBot, particular auctioneer programs are built on underlying infrastructure that supports all of the operational tasks. New auctioneers need implement only their unique features, such as the matching and quote functions.

Secondly, the parametrization supports the task of building software agents by establishing the semantics of an auction description language. Such a language is crucial if we are to build infrastructure that enables software agents to participate in wide-scale electronic markets. Only if the agent is able to understand a particular market's rules will it be able to synthesize a bidding strategy.

This parametrization is influenced by two goals. The first is to define parameters, and the values they can take on, in a manner consistent with the multidimensional auction perspective taken in Chapter 2. The second is to make the parameters orthogonal—the more independent they are, the more easily they can be combined. Both of these goals are met to a high degree of success by the parametrization described in the following text.

The parametrization is necessarily incomplete—there are mechanisms in the literature, and in practice, which cannot be described by the parameters presented herein. Nevertheless, a broad range of auction types are covered—much broader than any competing parametrization. I illustrate the flexibility of the parametrization with selected examples, but do not attempt to identify every known auction type that fits in this framework.

3.2 Common Auction Characteristics

3.2.1 Classic Auction Types

Many different types of auctions are in common use. The English open-outcry auction is often used to sell art and other collectibles, for example. The Dutch auction is commonly used to sell perishables, such as fish or flowers. First-price sealed bid (FPSB) and second-price sealed bid (SPSB, or *Vickrey*) auctions are most often used in procurement situations. Call markets and continuous double auctions (CDAs) are favored institutions for trading securities and financial instruments. These institutions and others are discussed in various auction survey papers (Engelbrecht-Wiggans, 1980; Friedman, 1993; McAfee and McMillan, 1987; Milgrom, 1987; Milgrom and Weber, 1982).

Some authors organize auction designs in a hierarchical taxonomy (Engelbrecht-Wiggans, 1980; Friedman, 1993). The straw-man example depicted in Figure 3.1 classifies the auctions mentioned in the preceding paragraph by whether they are single or double sided,

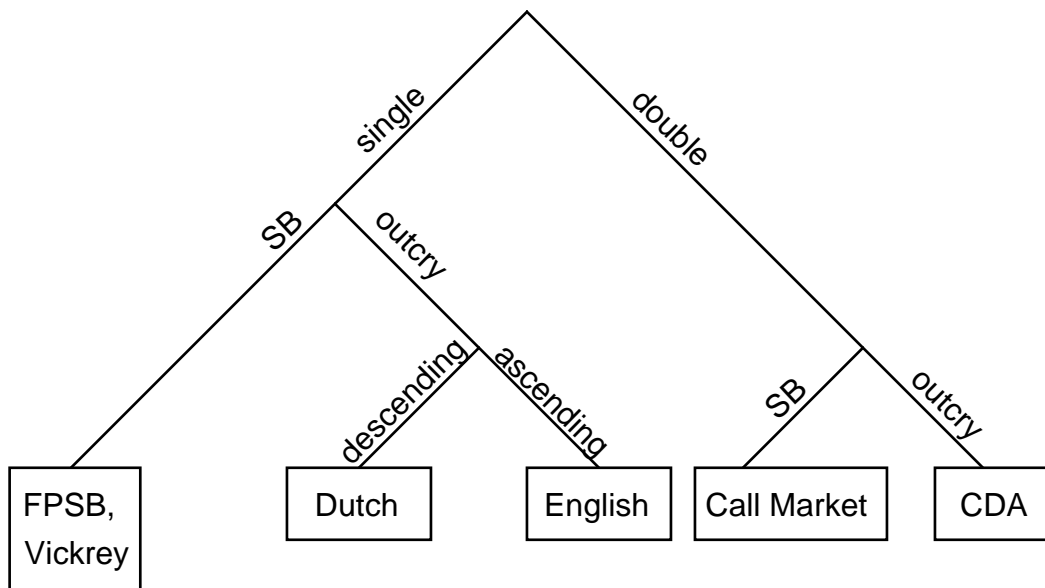


Figure 3.1: A classification of classic auction types.

and then by whether they are sealed-bid or open-outcry. The ascending/descending distinction differentiates the English and Dutch outcry auctions.

Explicit tree representations, however, introduce an artificial ordering on design decisions. In addition, any given tree obscures common features on different branches. Rather than use a hierarchical taxonomy, I focus on the features that define the commonalities and differences among various auctions.

3.2.2 Three Core Activities

The first step in organizing the design space is to recognize three core activities common to auctions. All auctions must perform the first two activities, and most perform the third as well. In the course of an auction, activities of these types may be interleaved and iterated any number of times, depending upon the auction rules.

- **Receive bids:** *Bids* are the messages sent by agents to indicate their willingness to participate in exchanges. On receiving a bid, the auction verifies that it satisfies the auction rules, and if so, *admits* it into the active set of bids.
- **Clear:** The central purpose of an auction is to *clear* the market, determining resource exchanges and corresponding payments between buyers and sellers.
- **Reveal intermediate information:** Auctions commonly supply agents with some form of intermediate status information, typically in the form of hypothetical results

were the auction to clear at that moment. I refer to these status reports generically as *quotes*.

This perspective leads naturally to a description of auction features along three axes: bidding rules, clearing policy, and information revelation (quote) policy. Section 3.3 elaborates on these three axes.

3.2.3 The Semantics of Bids

The canonical bid in an English open-outcry auction represents an offer by the bidder to purchase a unit item at the stated price. Presumably, the bidder will be happier to get the item at an even lower price. Above the stated price, the bidder has not expressed a willingness to buy anything.

The bid is a reflection of the agent’s *demand* for the resource. In the particular case of competitive equilibrium analysis, an agent’s *Walrasian demand correspondence* assigns the set of utility-maximizing allocations to every price vector \mathbf{p} (Mas-Colell et al., 1995, page 23). In Tatonnement-like protocols, such as the WALRAS algorithm (Wellman, 1993; Cheng and Wellman, 1998), a competitive agent’s bid in a (one-dimensional) auction with scope j , is its demand correspondence *assuming* the prices of the resources in $\mathcal{J} \setminus j$ are fixed.

Throughout this section, the net allocation, \mathbf{z} , and the price vector, \mathbf{p} , should be considered implicitly scoped to the auction, unless otherwise stated.

When we admit nonlinear pricing of multiple resources, it may not be convenient to express an agent’s valuation for a combination of resources in terms of prices on the individual resources. Consider, for example, an agent whose valuations over three allocations are $v_i((1, 1, 0)) = \$8$, $v_i((1, 0, 1)) = \$12$, $v_i((0, 1, 1)) = \$3$. It is quite complicated to express this agent’s valuations in terms of individual prices on each of the resources; it is much more compact to specify the agent’s valuations directly in terms of net payments.

Thus, I admit bids in both forms: as a correspondence between prices and quantities, denoted w_i , or between payments and net allocations, denoted ξ_i (see Section 2.2.2 for definitions of prices and payments). Note that the former is a special case of the latter—a vector of quantities specifies a net allocation and the product of quantities and prices determines a payment. I use the terms *bid* and *offer* interchangeably to refer to agent messages.

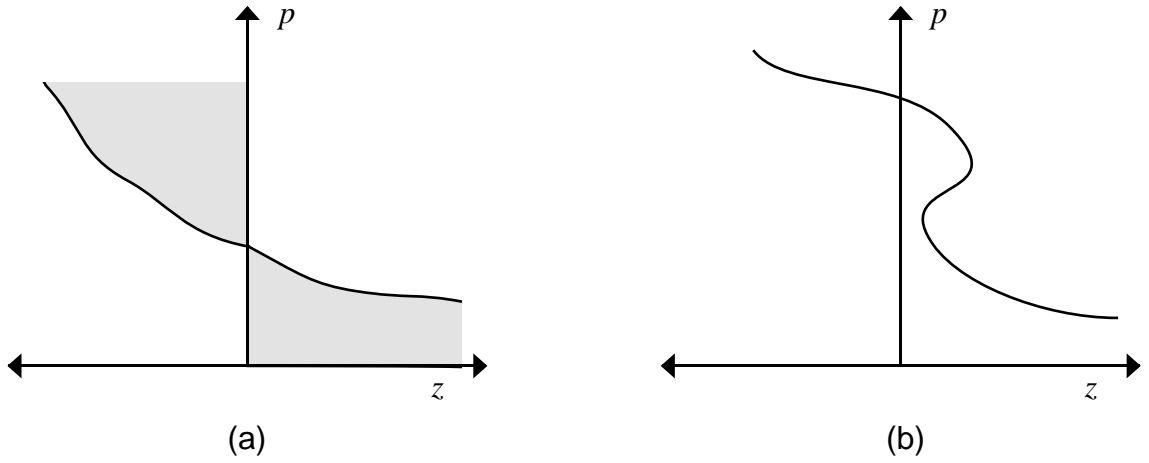


Figure 3.2: Examples of continuous bids that are (a) monotone, and (b) nonmonotone in prices. The shaded area in (a) indicates the correspondence when the bid is divisible.

Bids in Price Space

Let w_i be i 's bid, expressed as a correspondence between prices and quantities. The term $w_i(\mathbf{p})$ represents the set of net allocations that i expresses a willingness to accept at price vector \mathbf{p} . Its inverse, $w_i^{-1}(\mathbf{z})$, denotes the set of price vectors for which i would accept the net allocation \mathbf{z} .

In order to define bid relationships, we require some notation for comparison between sets. Let Ω be a set with a partial preorder, \geq , and let Ω' and Ω'' be subsets of Ω . We say that $\Omega' \gg \Omega''$ iff: (i) for all $\omega' \in \Omega'$, there is an element, $\omega'' \in \Omega''$ such that $\omega' \geq \omega''$, and (ii) for all $\omega'' \in \Omega''$, there is an element, $\omega' \in \Omega'$ such that $\omega' \geq \omega''$. In other words, every element of Ω' has a lesser element in Ω'' , and every element of Ω'' has a greater element in Ω' .

A bid is *monotone in prices* if its offer correspondence is nonincreasing in the sense of \gg .

Definition 3.1 w_i is monotone in prices iff $\hat{\mathbf{p}} > \mathbf{p}$ implies $w_i(\mathbf{p}) \gg w_i(\hat{\mathbf{p}})$.

Figure 3.2 illustrates monotone and nonmonotone continuous bids in a single price. The next example considers monotone discrete offers in two dimensions.

Example 3.1 Consider a mechanism for two resources, with price vector denoted (p_A, p_B) and net allocations denoted $(z_{i,A}, z_{i,B})$. Suppose that an agent's offer states that $w_i((1, 2)) = \{(4, 3), (5, 2)\}$. Values of the correspondence at the price vector $(2, 2)$ that are consistent with monotonicity include

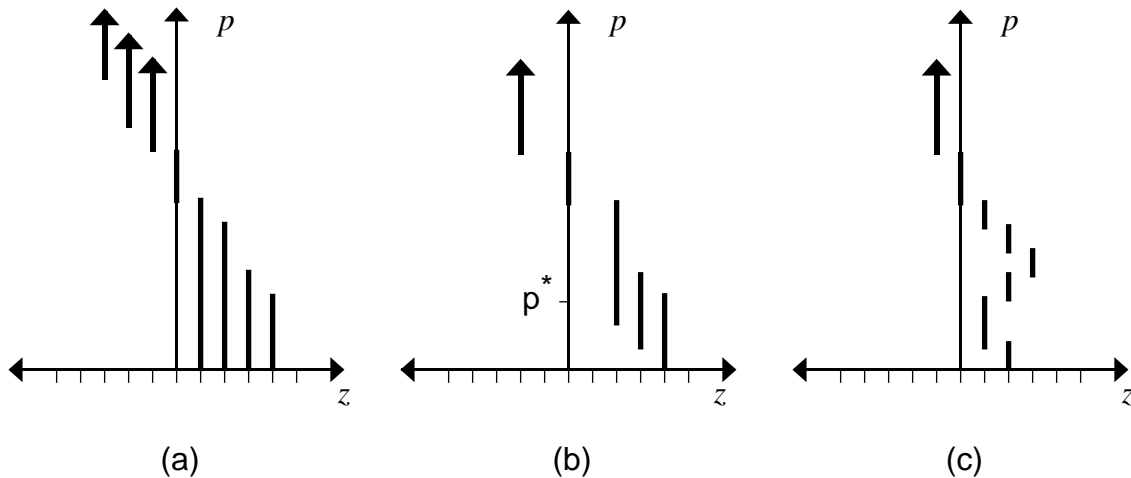


Figure 3.3: Examples of discrete bids that are (a) monotone and divisible, (b) monotone but not divisible, (c) not monotone.

$$w_i((2, 2)) = \{(4, 2)\} \text{ or } w_i((2, 2)) = \{(3, 3), (5, 1)\}.$$

A bid is *divisible* if whenever some allocation is acceptable at a price, all fractions of that allocation are also acceptable. To express this formally, let $\Lambda(\mathbf{z})$ be the set of vectors in Z between the zero vector and \mathbf{z} . That is,

$$\Lambda(\mathbf{z}) = Z \cap \{\alpha\mathbf{z} \mid \alpha \in [0, 1]\}.$$

Definition 3.2 *The bid w_i is divisible if, for all \mathbf{p} ,*

$$\mathbf{z} \in w_i(\mathbf{p}) \Rightarrow \Lambda(\mathbf{z}) \subseteq w_i(\mathbf{p}).$$

Examples of monotonicity and divisibility in discrete bids are shown in Figure 3.3. The offer in Figure 3.3(b) is indivisible: although $w_i(\mathbf{p}^*) = \{2, 3, 4\}$, it does not include $z = 1$. The continuous bid of Figure 3.2(a) is divisible if the shaded area is included in the correspondence, indivisible otherwise.

If a bid is monotone and divisible, then the offer correspondence can be specified conveniently in terms of its boundary, rather than enumerating the entire set of acceptable net allocations for each price.

Bids in Payment Space

I now consider bids expressed as a correspondence between payments and net allocation vectors, focusing on the monotonicity property. A bid is *monotone in payments* if its offer

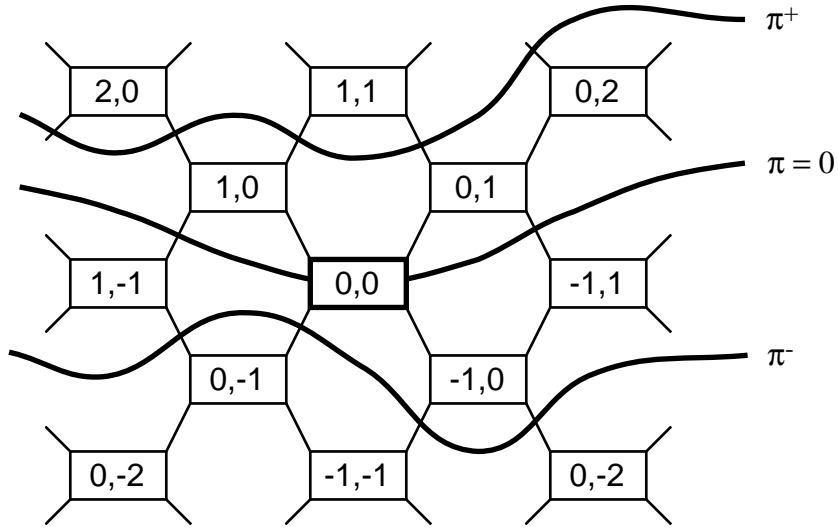


Figure 3.4: Iso-payment curves on a lattice.

correspondence requires greater allocations as payments increase.

Definition 3.3 *The offer ξ_i is monotone in payments, if, for all $\boldsymbol{\pi}$, \mathbf{z} , $\hat{\boldsymbol{\pi}} > \boldsymbol{\pi}$, and $\hat{\mathbf{z}} > \mathbf{z}$,*

$$\begin{aligned} \mathbf{z} \in \xi_i(\hat{\boldsymbol{\pi}}) &\Rightarrow \mathbf{z} \in \xi_i(\boldsymbol{\pi}), \text{ and} \\ \mathbf{z} \in \xi_i(\boldsymbol{\pi}) &\Rightarrow \hat{\mathbf{z}} \in \xi_i(\boldsymbol{\pi}). \end{aligned}$$

In other words, if \mathbf{z} is acceptable at some payment, then the agent is willing to take \mathbf{z} at a lesser payment, or a greater net allocation at the same payment. Figure 3.4 illustrates *iso-payment curves* (i.e., curves that connect vectors of equal valuations) on a two-dimensional lattice. The curve labeled π^+ represents a positive payment by the agent. Net allocations below this curve, such as $(0, 1)$, indicate that the agent is not willing to pay π^+ for that change in its allocation. Net allocations above this curve are acceptable to the agent for the payment π^+ . The curve labeled $\pi = 0$ represents a net payment of zero, and must intersect the zero allocation vector.

Figure 3.5 illustrates iso-payment curves on a two-dimensional lattice for a nonmonotone offer. The agent is willing to exchange $(1, -1)$ for a payment π^+ , but is not offering to take $(1, 0)$ at the same payment.

Note that monotonicity in prices does not imply monotonicity in payments, nor vice versa. A simple example with discrete offers illustrates the point.

Example 3.2 *Consider a one-dimensional offer for a discrete good. Suppose i offers to buy (up to) $z_{i,j}$ at a price p_j , and (up to) $\hat{z}_{i,j}$ at \hat{p}_j , where $\hat{p}_j < p_j$.*

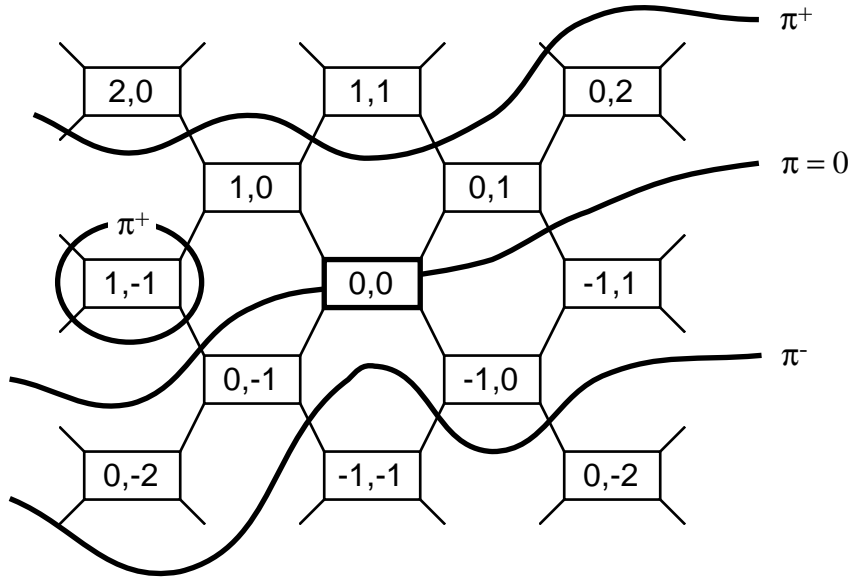


Figure 3.5: Nonmonotone iso-payment curves on a lattice.

The payments associated with $z_{i,j}$ and $\hat{z}_{i,j}$ are $\pi_{z_{i,j}} = p_j z_{i,j}$ and $\hat{\pi}_{z_{i,j}} = \hat{p}_j \hat{z}_{i,j}$, respectively. Monotonicity in prices requires $z_{i,j} \leq \hat{z}_{i,j}$, but it does not impose monotonicity in the payments. For instance, if $z_{i,j} = 10$, $p_j = 4$, and $\hat{p}_j = 2$, any $\hat{z}_{i,j} \geq 20$ is consistent with both monotonicity in prices and monotonicity in payments. However for $\hat{z}_{i,j}$ between 10 and 20, $\pi_{z_{i,j}} > \hat{\pi}_{z_{i,j}}$, and therefore $\hat{z}_{i,j} \notin \xi_i(\pi_{z_{i,j}})$.

3.3 The Auction Parameter Space

In this section, I present a parametrization of the auction design space that is broad enough to encompass most of the classic auctions, common commercial and online auctions, and many others. I have attempted to define the parameters in such a way as to make them orthogonal. One-dimensional versions of most of these parameters have been implemented in the AuctionBot system, providing users great flexibility in choosing the rules of the auctions they create. We are currently in the process of extending the AuctionBot to support various forms of multidimensional auctions.

Earlier, sparser versions of this parametrization appear in previous reports (Mullen and Wellman, 1996; Wurman et al., 1998b). Other researchers have also attempted to organize the space of auction designs. Engelbrecht-Wiggans (Engelbrecht-Wiggans, 1980) parametrizes a smaller set of auctions as part of a broad classification of auction re-

search. Friedman (Friedman, 1993) presents a taxonomic structure of the design space with particular emphasis on variations of the CDA. The FM96.5 (FishMarket) testbed (Rodríguez-Aguilar et al., 1998) is based on a detailed parametrization of the space of Dutch auctions. IBM developed an Internet auction server that implemented a subset of the parameters described below in the context of single-unit auctions (Kumar and Feldman, 1998). None of these previous works, including my own, considered multidimensional auctions or bids in the form of arbitrary correspondences.

My presentation is organized along the three axes introduced in Section 3.2.2: bidding rules, clearing policy, and information revelation policy.

3.3.1 Bidding Rules

Bidding rules determine under what conditions bids may be introduced, modified, or withdrawn, as a function of agent identity, current bid status, or even the entire auction history. If an incoming bid satisfies these rules, then the auction *admits* it into the set of current bids. If a bid fails to satisfy the admission criteria, the auction notifies the agent that the bid was rejected.

Because bids can express arbitrary correspondences, we can assume (without loss of generality) that each agent has at most one active bid in an auction at any time. An agent changes its bid by submitting a new one, if allowed by the auction rules.

Expressiveness

An auction dictates a *language* for bids, defining their syntax as well as expressive power. In this discussion I am concerned with semantics, and thus limit attention to expressive power. The bid semantics dictate whether offers are in terms of prices or payments, and the class of correspondences that may be expressed.

For example, classes of correspondences supported by bid languages in the Michigan Internet AuctionBot include the following:

- **Price-quantity schedules.** A bid schedule is a stepwise specification of offers to buy or sell various quantities at discrete price points. The price-quantity expression has two subclasses, corresponding to restrictive special cases:
 - **Single units.** An agent can express a single-unit offer concisely as a price and a sign indicating whether the offer is to buy or sell.
 - **Single price points.** This option restricts offers to a fixed number of units

(buy or sell) at a single, per-unit price.

- **Continuous.** Using analytical expressions, a bid may specify offers that are continuous functions of prices.
- **Combinatorial.** When the auction is multidimensional and mediates the allocation of discrete goods, agents make offers on bundles of resources. Bids are correspondences between bundles and payments.

General properties of correspondences, such as those mentioned in Section 3.2.3, can be used to further restrict bid expressions. For example, an auction might require that bids be monotone or divisible. When they are divisible (whether required or not), then a price-quantity schedule or continuous offer function would typically be interpreted as a boundary on the set of acceptable allocations in the offer correspondence.

Restrictions on bid expressions may reflect domain constraints (e.g. in the scheduling domain, bids can be expressed very concisely by the tuple $\langle \text{length, deadline, payment} \rangle$), or auction policies adopted for reasons of computation, communication, or incentive engineering. For example, single price points may be considered easier to specify than complete schedules. As another example, requiring that offers be divisible may simplify clearing calculations (discussed in Chapter 4). In the multidimensional context, Rothkopf et al. (1998) present several cases where restricting the expressive power of bids allows polynomial-time computation of optimal allocations in combinatorial auctions for discrete resources.

Buyers and Sellers

Typically, we classify auctions by whether they have one buyer or many buyers, and one seller or many sellers. The three combinations of interest are {one buyer:many sellers}, {many buyers:one seller}, and {many buyers:many sellers}. A restriction to “one” essentially means that the auction is one-sided, and the sole buyer or seller must be designated.

For example, a one-dimensional auction with a single buyer, designated h , imposes the restrictions

$$\begin{aligned} \forall i \neq h, \quad \forall p_j, z \in w_i(p_j) \quad z \leq 0, \\ \forall p_j, z \in w_h(p_j) \quad z \geq 0. \end{aligned}$$

Similarly, if h is the designated sole seller,

$$\begin{aligned} \forall i \neq h, \quad \forall p_j, z \in w_i(p_j) \quad z \geq 0, \\ \forall p_j, z \in w_h(p_j) \quad z \leq 0. \end{aligned}$$

The restrictions can be extended in a straightforward manner to bids expressed in payments. Note that in the {many buyers:many sellers} case, agents' bids can express offers to either buy or sell, or both.

In the multidimensional case, each agent's bid may be permitted to express willingness to buy or sell, or both, for each resource type. An even more general formulation of this rule allows the auction to restrict each agent's bids to certain ranges of the offer correspondence space.

Dominance

Bid dominance rules restrict the relationship of an agent's new offer to the bid, if any, it replaces. Both increasing and decreasing forms of dominance may apply, and, when bids are monotone, an auction can impose different restrictions on the buy and sell sides of the bid.

Let w_i and \hat{w}_i be two bids.

Definition 3.4 \hat{w}_i is superior to w_i in the range $[\underline{\mathbf{p}}, \bar{\mathbf{p}}]$ iff, for all $\mathbf{p} \in [\underline{\mathbf{p}}, \bar{\mathbf{p}}]$, $\hat{w}_i(\mathbf{p}) \gg w_i(\mathbf{p})$.

Definition 3.5 \hat{w}_i is inferior to w_i in the range $[\underline{\mathbf{p}}, \bar{\mathbf{p}}]$ iff w_i is superior to \hat{w}_i in that range.

Figure 3.6 illustrates a bid and three alternative bids. Bid w' is superior to w over all prices, w'' is inferior over all prices, and \tilde{w} is inferior in the prices that correspond to negative quantities, and superior in positive quantities.

The *ascending rule* requires that new bids be superior to old bids, while the *descending rule* requires new bids be inferior. Intuitively, the combination that requires that the buy-side of a bid increases and the sell-side of the bid decreases is the most natural, since it represents a strengthening of the offer on both sides. However, the rule that requires a seller to *increase* its sell bid has found use as part of a protocol for decentralized task allocation (Walsh and Wellman, 1998). It is an example of how an expanded view of the auction design space can lead to new, potentially interesting protocols.

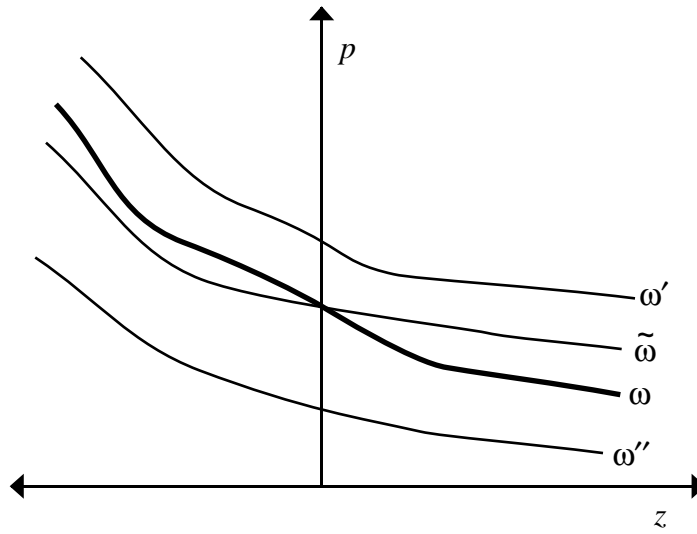


Figure 3.6: Examples of continuous bids that are superior, inferior, and mixed with respect to the original bid.

Note that the increasing restriction, when applied to the sell side, allows an agent to effectively withdraw its bids by offering to sell at infinity. Similarly, when the decreasing rule is applied to the buy side, the agent can effectively withdraw by offering to buy at a price of zero.

For bids expressed in payment space, $\hat{\xi}$ is superior to ξ if it expresses maximal payments at least as great on all net allocation vectors. For monotone bids, this is the case iff, for all π , $\hat{\xi}_i(\pi) \supseteq \xi_i(\pi)$. That is, the set of net allocations acceptable at the payment π expressed by $\hat{\xi}$ must be a superset of those expressed by ξ .

It is useful to compare the effects of increasing a bid in price space on the representation of that bid in payment space. Consider two points on a price-quantity schedule: (z^1, p^1) and (z^2, p^2) , where $z^1 < 0 < z^2$ and $p^1 > p^2$. A new bid \hat{w} , that is superior (in prices) to w , increases the price at which the agent is willing to exchange the respective quantities. Projected into payment space, $\hat{\pi}_{z^1} < \pi_{z^1}$ (because z^1 is negative), and $\hat{\pi}_{z^2} > \pi_{z^2}$. Thus, the ascending price rule causes the bid to *increase* on the buy side and *decrease* on the sell side in payment space.

Thus, because of the sign change, the direction of influence of the ascending rule in payments is the same as the mixed rule in prices. Figure 3.7 illustrates the four offers in Figure 3.6 mapped into payment space. $\tilde{\xi}$, which had mixed dominance in prices, is superior to ξ in payments.

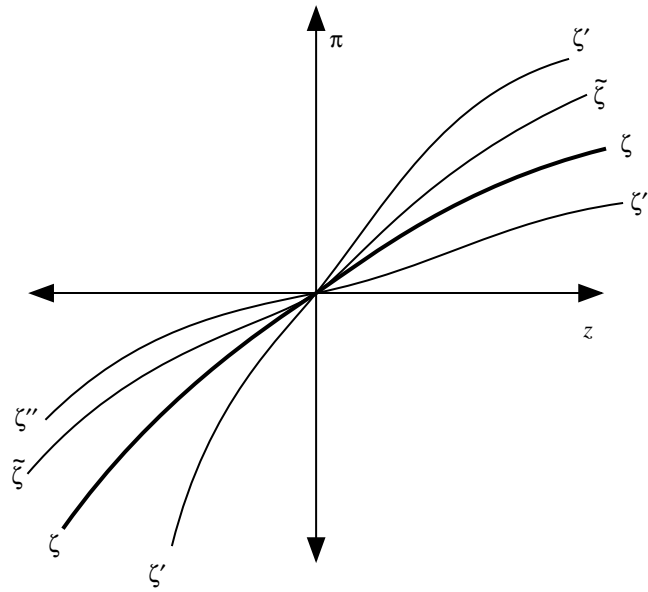


Figure 3.7: The four price-space bids from Figure 3.6 mapped into payment space.

Beat-the-Quote

The dominance rules of the previous section require that new bids bear some relation to previous bids by the same bidder. Many auctions require that a new bid satisfy some condition defined in terms of bids by other bidders. For example, in an English outcry auction, a new bid must beat the highest so far (perhaps by a specified increment). We capture this concept by defining analogous dominance rules with respect to *price quotes*, summary information about the bid state revealed by the auction (discussed in detail in Section 3.3.3). The purpose of such rules is to ensure a progression of prices, thus directing the process to a steady state, if not an actual equilibrium.

Typically, the auction's quote provides evidence regarding potential outcomes of the auction, in the form of the agent's tentative allocation. Recall that $\chi_i(q, w)$ denotes the allocation that price quote q entails agent i would receive had it sent the bid w to the auction.¹

Again, let w_i be the agent's current bid, and \hat{w}_i its new bid. The one-dimensional

¹This interpretation is typical, but not required. That is, a beat-the-quote rule is well-defined for any coherent interpretation of $\chi_i(q, w)$, as long as this function itself is well-defined.

beat-the-quote rule requires that:

$$\begin{aligned} &\text{if } \chi_i(q, w_i) > 0, \quad \chi_i(q, \hat{w}_i) \geq \chi_i(q, w_i), \\ &\text{else if } \chi_i(q, w_i) < 0, \quad \chi_i(q, \hat{w}_i) \leq \chi_i(q, w_i). \end{aligned}$$

In words, an agent's new bid must keep the same sign of, and must not decrease the magnitude of, its tentative allocation. The rule for payment space is the same, with ξ substituted for w . Note that it would not have been sufficient to define this rule in terms of simple price/payment increases (or decreases), since bids in general refer to various quantities at various prices/payments.

Example 3.3 *Consider a one-dimensional auction for discrete resources in which an agent has submitted an indivisible bid. Suppose the agent has a current offer to sell two units if the price is between \$5 and \$10, or sell four units if the price is \$10 or greater. A price quote of \$8 implies that the agent is winning two units. A new offer to sell two units if the price is between \$5 and \$7, or four units if the price is \$7 or greater, would increase the agent's tentative winnings to four units, and thus satisfies the beat-the-quote rule. However, a new offer to sell two units if the price is between \$3 and \$10, and four units if the price is \$10 or greater, would not satisfy the beat-the-quote rule.*

Notice that the bid dominance and beat-the-quote rules are complementary. Both potential new bids in Example 3.3 dominate the original bid, but only the first beats the quote. The next example demonstrates that a new bid can beat the quote without dominating the previous bid.

Example 3.4 *Consider an agent with a divisible offer to buy two units at \$2, or one at \$10. Suppose the price quote is \$6. A new bid, in which the two-unit offer is at \$5 and the one-unit offer at \$7, would beat the quote, but would not satisfy dominance because the offer decreased from one to zero units in the range [\$7, \$10].*

We can generalize this to the multidimensional case by requiring that it hold with respect to each resource. This requirement may be too strong, however, as evidenced by the following example.

Example 3.5 Consider a multidimensional auction for two single-unit resources. There are four possible net allocations: $(0, 0)$, $(1, 0)$, $(0, 1)$, and $(1, 1)$. Suppose an agent has bid \$2 on $(1, 0)$, \$4 on $(0, 1)$, and \$7 on $(1, 1)$. The auction announces the price quote $\{\pi_{(1,0)} = 3, \pi_{(0,1)} = 3, \pi_{(1,1)} = 8\}$. This quote implies that the agent is currently winning $(0, 1)$. It is quite possible that the agent would desire to raise its bid on $(1, 0)$ next. However, the version of the beat-the-quote rule described above would not permit this switch because the quantity of the second resource decreases.

An alternative multidimensional generalization would require only that the beat-the-quote condition hold with respect to *some* resource. This may seem rather weak, but in conjunction with a dominance rule, is often exactly what is desired. For example, several of the combinatorial auctions mentioned in Chapter 5 essentially apply this combination of bid restrictions.

The most common variant of the beat-the-quote rule requires that the new bid increase the price by some increment δ above (or below) the quoted prices. For example, in the English auction, when the agent is not winning the good, its new offer must not only make it the winning bidder, but it must do so at a price at least δ above the current winning bidder. We extend this concept to the more general case in the following manner. New bid \hat{w}_i beats the quote by δ iff there exists some w' such that:

1. w' satisfies the beat-the-quote rule (without δ),
2. \hat{w}_i is superior to w' , and
3. for some $z \geq 0$, $\hat{w}_i^{-1}(z) \geq w'^{-1}(z) + \delta$, or for some $z \leq 0$, $\hat{w}_i^{-1}(z) \leq w'^{-1}(z) - \delta$.

Such a rule could similarly be formulated for payment-space.

The beat-the-quote-by- δ rule is used to speed the progression of prices. There is a risk that the rule will result in a loss of efficiency when an agent values the object more than the winner, but not by enough to continue bidding. However, in many protocols—such as the multi-item auction of Demange et al. (1986)—the efficiency loss is bounded, typically by an amount linear in δ .

Withdrawal and Expiration Rules

Auction rules dictate whether bid withdrawals are allowed, and if so when. For example, one possible rule is that withdrawals are permitted only in conjunction with a clearing

operation. An expiration is a planned withdrawal, typically specified at bid time, also subject to permissibility by the auction rules.

Activity Rules

In many complex domains, agents can benefit from a strategy in which they withhold information while others reveal information. For example, the top two performing agents in the 1990-91 Santa Fe Institute double auction tournament employed a strategy of waiting in the background while others bid. When the market seemed about to converge, the agents would step in and “steal the deal” (Rust et al., 1994). Clearly, if everyone employed this strategy, the market would fail. Activity rules are designed to counteract such behavior.

DeMartini et al. (DeMartini et al., 1998) present a formal definition of one feasible activity rule.² In each round, an agent is eligible to place bids on as many items as it had (provisionally) winning bids, plus the number of bids it placed in the previous round. For example, if the agent was provisionally winning items A and B last round, and it chose to enhance its bid with offers on items C and D, then it is eligible to bid on up to four items.

Systematic parametrization of the space of activity rules would be quite useful, and is a topic for future work.

3.3.2 Clearing Policy

As mentioned in Chapter 2, the purpose of an auction is to compute a set of exchanges based on the bids it has received. I formalize the auction’s clearing policy in terms of the matching function, γ . This section details the parameters that determine how and when this exchanges are computed.

Matching function

We can view the matching function as determining the exchanges in two steps: first determining which agents will trade, and second, the exact terms of each exchange. A set of exchanges, Θ , *clears* the auction if no mutually beneficial trades exist among the bids remaining after the exchanges are executed. Although it is typical for the matching

²The rule described was employed in the Simultaneous Ascending Auction used by the FCC, and is part of the proposed RAD mechanism.

Agent	Offer
Agent 1	sell 1 unit at \$1
Agent 2	buy 1 unit at \$2
Agent 3	sell 1 unit at \$3
Agent 4	buy 1 unit at \$4

Table 3.1: An example with four bids.

function to clear the auction (hence the name of the axis), it is not necessary. An allocation is *locally efficient* if it maximizes the total surplus as represented by the bids.

Example 3.6 *Consider two candidate allocations for the single-unit bids listed in Table 3.1. In the first, agents 1 and 4 trade. This trade allocates the goods to the agents with the highest stated values—agent 4 buys one and agent 3 keeps one. In the second potential allocation, agent 4 trades with agent 3, and agent 2 trades with agent 1. The agents who end up with the goods have the first and third highest valuations. This allocation is inefficient because agents 2 and 3 could both be made better off by a further trade between them. Although both sets of exchanges clear the market (i.e., there are no trades available among the remaining bids), only the former is locally efficient.*

In general, there may be many allocations that satisfy local efficiency but differ in their allocation of money.³ For example, the exchange that results in the locally efficient solution for Example 3.6, namely, that agent 1 sells a unit to agent 4, would be mutually beneficial if agent 4 paid agent 1 any price between \$1 and \$4 for the object. At the lower end of the range, agent 4 captures all \$3 of the surplus. At the higher end of the acceptable price range, agent 1 captures the surplus.

The selection of matching functions depends on the nature of the resource, the dimensionality of the auction, the expected format of the agents' utility functions, and the designer's goals. Matching functions are the subject of the three chapters following the present one. Chapter 4 discusses matching functions, and computational algorithms that implement them, for one-dimensional, discrete-good auctions. Chapter 5 presents a novel quote and matching function for combinatorial auctions. Chapter 6 introduces an algorithm for computing a clearing price in multidimensional auctions for continuous resources.

³There may also be multiple locally efficient solutions that differ in their distribution of resources, for instance, if there are tie bids.

Clear Timing

This important parameter of the clearing policy determines *when* clears should occur. Some common forms of timing policy include:

Scheduled. Clears occur at a specified set of nominal times. The specification may take the form of an explicit enumeration, or some implicit description, for example, the frequency of a periodic clear.

Random. Clears are determined according to some random distribution. Memoryless distributions, such as the Poisson, are attractive candidates because they deter agents from applying complex time-dependent strategies. The relevant parameters of the distribution are typically made public, so the agents participate with some expectation of the timing of events.

Bidder activity. Clears occur whenever a new bid is admitted. This is the timing policy of *continual* auctions such as the CDA. A variation of this rule allows for *synchronized* auctions—rather than clearing when a bid is admitted, the auction can clear when a new bid has been received from each participant, or when a fixed number of bids has been received.

Bidder inactivity. The auction clears when no bid has been admitted for a specified period.

Auctions can also combine these schedules in various ways. For example, the online version of the English auction clears at a time determined by the latest of a scheduled time and a period of inactivity.

Closing Conditions

The closing conditions are logical tests that determine whether a clear should be the final clear. Auctions can close at a scheduled time, at a random time, after a period of inactivity, or when the bids of designated agents (e.g., the seller, in a single-seller auction) are matched. In some cases, we desire that auctions close when an external signal is received (e.g., indicating that some more global quiescence property has been achieved (Wellman and Walsh, 1999)).

Tie-breaking

A *tie* occurs whenever two agents express a willingness to take the same net allocation at the same price (or payment). The manner in which ties are resolved can also influence the outcome of an auction. Three common methods are to break ties arbitrarily, in favor of the earlier bids, or in favor of bids for larger quantities.

Auctioneer fees

The policies discussed thus far determine payments between the buyer and seller. In commercial auctions, it is common for the auctioneer to collect some fees, and these payments must be considered by the agents. A payment may be required of the buyer, seller or both. Common types of transaction fees include:

Entrance fee. A fixed fee for the agent's first bid. This is a generalization of a listing fee.

Bid fee. A fixed fee paid with every bid. Bid fees can provide a disincentive to price manipulation (McCabe and Smith, 1993).

Ad valorem. A percentage of the exchange price.

Nonlinear. A nonlinear function of the exchange price, such as those often used to provide quantity discounts.

These are just some of the possible fee structures. When an outside subsidy is present, these fees can flow the opposite direction—from auctioneer to agents. Anderson et al. (1999) explore the incentive properties of k -double auctions (Satterthwaite and Williams, 1989) with various fee structures.

3.3.3 Information Revelation

The parameters discussed below control the timing and content of quotes.

Price Quotes

The feature common to price quotes in many classic auctions is their information content. A price quote informs an agent of the range of offers that would have been in the exchange set had the auction cleared at the time the quote was issued. This definition is given in past hypothetical tense because a price quote is necessarily relative to the bidding state

at quote time—if the time necessary to transmit the price quote message varies between agents, the price quote may be stale before it is received.

This definition is consistent with the discussion of mechanism signals in Section 2.2.1. The quote function, ρ , is used to produce price quotes from bids, and if the quote is separating, then an agent can correctly determine its hypothetical allocation.

In many cases, the quote function ρ is an impotent version of the matching function γ : it calculates hypothetical exchanges and then announces the prices (or payments) without actually clearing. For example, the English auction uses the first-price rule to both generate price quotes and determine the exchange when the auction clears.

However, it is not necessary to use γ as a basis for ρ . Rassenti, et al. (1982) and DeMartini, et al. (1998) both present combinatorial auctions that use relaxed linear programming techniques for ρ , and integer programming for γ . In both cases, the information provided by the quote q is not always enough for an agent to correctly infer its allocation.⁴

In one-dimensional auctions with continuous, strictly monotone bids, the price quote, necessarily, must be the single clearing price, p^* , that balances aggregate supply and demand. In one-dimensional auctions for discrete resources with divisible bids, there is typically a range of clearing prices, specified by the endpoints \bar{p} and \underline{p} . The *bid quote*, \underline{p} , is the price an agent would have to bid under in order to place a winning sell bid. The *ask quote*, \bar{p} , is the price an agent would have to bid over to place a winning buy bid.

In the CDA, standing buy and sell offers never overlap, hence the bid and ask prices reflect the spread between the highest buyer and the lowest seller. More generally, the bid and ask prices have meaning even if the buy and sell offers do overlap. Consider a set of L single-unit bids, of which M are sell offers and the remaining $N = L - M$ are buy offers. Sort all bids by price, and count down the list of bids. The M th unit determines \bar{p} , the upper bound of the separating price range, and the $(M + 1)$ st unit determines \underline{p} , the lower bound (Wurman et al., 1998a).⁵ In fact, Vickrey’s seminal article (Vickrey, 1961) is an examination of the special case where $M = 1$: \bar{p} corresponds to the first price, and \underline{p} to the second price.

Example 3.7 *Consider one agent with an offer to buy one unit at \$10, and another with an offer to sell one unit at \$5. The buy and sell offers overlap,*

⁴Note that non-separating, non-noisy quotes can still be used in the beat-the-quote rule. The agent can infer a single tentative allocation, but can’t be sure that is what it would have really received.

⁵The ask quote is undefined if there are no sell offers, and the bid quote is undefined if there are no buy offers.

Agent	Offer	Divisible?
Agent 1	sell 3 units at \$1	Yes
Agent 2	buy 2 units at \$2	No
Agent 3	buy 1 unit at \$5	Yes
Agent 4	buy 3 units at \$4	No

Table 3.2: An example with indivisible bids without an efficient clearing price.

so if the auction cleared at this moment, it would form an exchange. A bid-ask quote, in this case, would report that a new buyer would need to outbid \$10 (displacing the current buyer), and a seller underbid \$5 (displacing the current seller), in order to be part of the tentative allocation.

Under some conditions, if an agent's bid is equal to the bid or ask quote, it cannot tell whether it is in the exchange set. Thus, it is sometimes useful to augment the bid-ask quote with information directly telling the agent its tentative allocation given the current bids.

When multi-unit indivisible bids are allowed, separating prices may not exist. That is, there may not be a single per-unit price above which an agent's buy offer is accepted and below which it is rejected. For example, two of the bids shown in Table 3.2 are indivisible. If the per-unit price is set at or below \$4, then the resources are overdemanded. If the price is set above \$4, the resources are underdemanded. For instance, consider the case where the price is set to \$4.5 and 1 unit is sold to agent 3. What is particularly unappealing about this solution is that it is Pareto dominated by the allocation where agent 2 buys the remaining two units at a price less than \$2. Moreover, the socially efficient solution is to sell all three units to agent 4. Attempting to determine an exchange set based on a single price fails to exploit all profitable trades.

In order to present the agents with the information necessary to determine their hypothetical allocations, the auction needs to discriminate based on quantity. Nonlinear pricing allows an auctioneer faced with the example in Table 3.2 to quote, for example, \$3.5 for three units, \$4.5 for two units, and \$5.5 for one unit. These prices support the efficient allocation and clearly indicate to the agents whether they are in the tentative exchange set.

The topic of separating quotes in payment space is reserved for Chapter 5 in the context of auctions for bundles.

Before leaving the topic of price quotes, I should mention price clocks, which are used in variations of the Dutch auction. A specification of a price clock must include start

and end prices, and a (usually linear) price adjustment schedule. Clocks can be used to generate the bid quote, the ask quote, or both. Although the implementation of online Dutch auctions has been investigated (Rodríguez et al., 1997), considerable infrastructure is required to ensure that all agents have equal access to the auction over the distributed and asynchronous network. We argue elsewhere (Wellman and Wurman, 1998) that little is gained by attempting to reconstitute real-time auctions online; our effort is better spent developing auctions that take advantage of the asynchrony and flexibility of the online environment.

Quote Schedule

Like clear events, price quotes can vary in number and frequency. Some of the most significant choices are:

No price quotes. Auctions that reveal no information are traditionally called sealed-bid auctions.

Scheduled. Quotes are generated according to a specified nominal schedule.

Random. Price quotes are generated according to some stochastic process.

Bidder activity. Price quotes are generated with each new bid admitted.

Bidder inactivity. Price quotes are generated when no bid has been admitted for a specified period.

An auction may generate many price quotes as it proceeds, depending upon the quote schedule and the clearing and closing policies of the auction.

Order Book

The term *order book* is commonly used in organized exchanges, like the NYSE, to refer to the current set of active bids. The auctioneer may make some or all of the information in the order book public. The most common choices are to keep the book closed, reveal only the current winning bids, or to open the book completely. Online auctions, such as eBay and Onsale, commonly identify tentative winners.

Bidding rules	
Buyer/Seller	{one:many}, {many:one}, {many:many}
Expressiveness	single-unit point, single-price point, price-quantity step function
Bid Refinements	monotone, divisible
Sell-side dominance	ascending, descending
Buy-side dominance	ascending, descending
Beat-the-quote	seller, buyer, both
Information revelation	
Price quotes	bid-ask, nonlinear
Quote Timing	none, scheduled, random, activity, inactivity
Order book	closed, winners, open
Transaction history	public, private
Clearing policy	
Clear Timing	scheduled, random, activity, inactivity
Closing conditions	fixed time, random, inactivity, designated bids, equilibrium signal
Matching function	$\gamma^k, \gamma^{BP}, \gamma^{SP}, \gamma^{ET}, \gamma^{LT} \dots$
Tie breaking	arbitrary, earliest, quantity
Auctioneer fees	entrance, bid fee, ad valorem, nonlinear

Table 3.3: Summary of the parameters and their possible values for divisible-bid, discrete-good auctions.

Transaction History

Auctions may publicize selected information about past exchanges. Such information may include the prices, quantities, or even the identities of the transacting agents. If an auction has several clears but does not reveal historical prices, then agents get from the auction price information only for exchanges in which they participated. Publicly revealing past transaction prices avoids such an information asymmetry.

3.4 Parametrization of Well-Known Auctions

Table 3.3 shows a summary of the parameters and the unique values they can take when bids are monotone and for discrete resources. The matching functions are discussed in Chapter 4.

Many classic and online auctions can be described by the parameters in this chapter. Appendix B presents the parameter values for several well-known auction types. In cases where several common variations exist, such as the CDA, we describe just one of them.

As mentioned previously, the Dutch auction is something of a special case. The

parametrization does not fully specify the standard Dutch auction protocol. In particular, price quotes in our formulation are tied directly to bids—there is no notion of a price clock. However, we can capture the Dutch auction in our formulation with an appropriate combination of auction rules with a particular strategy played by the seller—namely, the single seller places a high initial bid and continuously decreases it until its bid is matched.

3.5 Conclusion

The auction design space presented in this paper captures the essential similarities and differences of many auction mechanisms in a more descriptive and useful format than the traditional taxonomic perspective. This parametrization is not exhaustive, but it is much more extensive than any others we have seen. We have found this organization of auction policy characteristics very useful in our development of the Michigan Internet Auction-Bot, and the approach seems to have been adopted in other research-oriented auction servers (Sandholm, 1999b). In particular, the deconstruction of auctions into functional components with parametrized behaviors coincides nicely with the object-oriented programming approach.

In addition, the parametrization serves as an organizational framework in which to classify research in auction analysis, and uncovers many new, potentially useful, mechanisms. I have seen instances in which an agent behavior that was assumed as part of a protocol was later found enforceable by a non-obvious combination of rules.

Finally, the parametrization facilitates the communication of auction rules to software agents—a critical step in the development of electronic commerce agents. The AuctionBot agent programming interface includes a method for software agents to request the rules of the auction, which enables agents to participate in the full range of auction types supported by the server.

Chapter 4

One-Dimensional Auctions for Discrete Resources

4.1 Overview

Matching functions are the components of auctions that determine the resources and monies exchanged between agents as a function of the bids. The matching function is invoked when the auction *clears*.¹ If no bids match, then no exchanges are formed.

Recall from Section 3.3.2 that a *locally efficient* matching function maximizes the surplus expressed in the bids. This definition, however, does not constrain the manner in which that surplus is divided among the participants. Clearly, the division of surplus influences agents' strategies. One of the primary results of mechanism design is that there does not exist a bilateral bargaining mechanism that is individually rational, efficient, and Bayesian incentive compatible for both buyers and sellers, that does not require outside subsidies (Myerson and Satterthwaite, 1983). In other words, there is no way to divide the surplus that guarantees that both agents' best response is to bid truthfully, unless the mechanism itself puts in some money or we compromise efficiency.

The matching functions described in the literature are of interest precisely because they make different tradeoffs between these desirable properties. However, most of these functions were originally described in the context of one particular set of rules, and must be generalized in order to serve as matching functions in the parameterization framework presented in Chapter 3. In this chapter, I define generalized versions of these common matching functions.

The remainder of this section reviews one-dimensional discrete-resource matching functions used in practice or proposed in the literature. Section 4.2 describes generalized ver-

¹I call the event that invokes the matching function a clear even if the matching function does not fully clear the auction.

sions of both *uniform-price* matching functions, in which the price per unit is the same for all transactions generated during a clear event, and *discriminatory-price* matching functions, which permit prices to differ in each exchange. Section 4.3 describes a computationally efficient algorithm for (uniform or discriminatory) one-dimensional auctions with divisible bids for discrete resources. Section 4.4 discusses issues related to matching functions for auctions with indivisible bids.

4.1.1 Matching Functions in the Literature

Consider, first, the four classic auctions for a single object. In the first price sealed bid auction (FPSB), the English open outcry auction, and the Dutch auction, the highest bidder wins and pays the price that it bid. In the Vickrey auction, the agent with the highest offer wins, and pays the price of the next highest bidder.

The classic auction for multiple buyers and sellers is the continuous double auction (CDA). If a new buy offer matches a standing sell offer, the auction immediately generates an exchange at the price of the standing offer. Likewise, if the new offer is a sell and the standing offer is a buy, the exchange occurs at the buyer’s offered price. If the new bid did not match a standing bid, it is added to the standing bid queue.

Satterthwaite and Williams (1989, 1993) developed the k -double auction, and investigated its application to call markets.² In the k -double auction, a uniform exchange price is determined in the following straightforward manner. First, the upper and lower clearing prices are calculated (as described in Section 3.3.3). Then, the exchange price is computed according to $p = k\bar{p} + (1 - k)\underline{p}$, where $k \in [0, 1]$.

McCabe , et al. (1993) also examined uniform price double auctions (UPDAs), and compared the empirical performance of two variants. The first adds a beat-the-quote-by-delta to a k -double auction.³ The second variant also uses a beat-the-quote rule, but has a different policy for computing quotes and exchange prices (the policy is called “IS” by the authors). \underline{p} is determined by the second highest offer in the union of all the non-winning buy bids and the lowest winning buy bid. \bar{p} is determined in an analogous manner using the sell bids. \bar{p} and \underline{p} are used in the auctions beat-the-quote rule. The price charged when the auction clears is halfway between the bid of the highest accepted seller and the

²A call market is a {many:many} auction with a fixed clearing time. Call markets are used to start the trading day on many stock exchanges.

³The authors used $\delta = 1$ in their experimentation, but this could be a variable.

lowest accepted buyer. The 1S rule is intended to speed convergence, and forgoes locally efficiency and separating price quotes to do so.

The Dual-Price mechanism (McAfee, 1992) uses the k -auction pricing but excludes the lowest-priced buy offer in the exchange set, and the highest-priced sell offer in the exchange set. The Dual-Price mechanism maintains individual rationality, incentive compatibility, and budget balance, but sacrifices local efficiency by eschewing the lowest-surplus trade.

4.1.2 Matching Functions Online

A variant of the English auction is the dominant online auction format when a single object is for sale. In addition, several online sites offer a “reserve price” auction which allows the seller to specify a reserve price in addition to the minimum bid price. A buy offer is admitted if it beats the current price quote, but it will not actually win unless it is the highest bid and it beats the seller’s reserve price, which is kept hidden from the buyers. This format is supplied to allow a seller to set the initial price low in order to attract bidders, without risking selling below her actual reserve price. EBay, Onsale, Amazon.com, and many others provide both of these auctions.⁴

More variety exists for auctions designed to sell multiple units. Onsale provides the “Yankee” auction, an ascending auction in which each winning bidder pays its bid. Yahoo’s auction site seems to be powered by Onsale’s auction engine, and as a consequence, provides the same multi-unit auction.

EBay and Amazon.com (and probably others) offer the misnamed “Dutch” auction—an ascending auction for multiple items in which all of the winning buyers pay the price of the lowest accepted buy bid. The bids are, by default, divisible. In actuality, this is a M th-price auction.⁵

Ubid supplies what they refer to as a “traditional” auction, which permits indivisible bids on multiple items.⁶ The auction uses a greedy algorithm as its matching function. The algorithm prioritizes bids by price, then quantity, then initial bid time, and greedily makes assignments until the total supply is exhausted. If a bidder’s indivisible offer to

⁴The URLs for these, and all other websites mentioned in this thesis, are listed in Appendix A.

⁵The fact that this auction has come into widespread use under the name “Dutch”, despite the long tradition of the descending-price Dutch auction, is more evidence in support of developing a descriptive language for auction rules.

⁶Egghead.com also seems to provide this auction format, although the description on their website was lacking considerably in detail.

buy x units cannot be filled, then the offer is skipped. The algorithm is computationally efficient ($O(m)$, where m is the number of bids). However, the information provided by the auction makes it difficult for participants to determine bidding strategies for two reasons. First, rather than announce prices, Ubid's implementation announces all of the current winning bidders, a list which often contains a hundred or more different bidders. Second, even though the auction employs an ascending rule, a bid that presently is not winning could become a winning bid in the future. Consider the simple case where there are two units for sale. The first bidder is agent 1, who bids \$5 for one unit. Then agent 2 bids \$6 each for two units, displacing agent 1. Finally, agent 3 bids \$8 for one unit. The algorithm now decrees that agents 3 and 1 are the winners.

The now-defunct Z-auction provided another twist on this auction format. Bidders were allowed to specify a "minimum accepted" quantity in addition to the desired quantity. This feature allowed users to place indivisible bids with a range of acceptable quantities. Z-auction also used a greedy algorithm to determine the winning bids.

Priceline.com sells airline tickets and new cars through a "reverse auction" format. A buyer names the price it is willing to pay for an item, such as an airline ticket, and within a few minutes, Priceline responds with either a rejection or a contract. It is unclear how the back end negotiation actually occurs, but the system can be modeled as a single-buyer auction for each purchase request with a clearing price determined by the buyer's original bid.

In reality, the number of options provided in the consumer market is limited. Much greater variety is likely to occur in the business-to-business markets. The higher stakes in these markets and increased complexity stemming from integrating dynamic trade into ongoing business relationships will require more sophisticated mechanisms. In addition, many procurement situations involve the repeated purchase of commodities that are bought and sold by several companies. For example, FastParts.Com operates a two-sided market to facilitate the exchange of electronic components. Their mechanism is essentially a CDA that allows participants to specify minimum lot sizes (i.e. single-price, indivisible bids).

This sampling of online auction sites provides background to the task of specifying common matching functions for one-dimensional, discrete-resource auctions. It is also worth noting that the parametrization from Chapter 3 captures all of the rules used in these online auctions except the "reserve price" feature.

4.2 Generalized, One-Dimensional Matching Functions

The matching functions described in this section apply when bids are divisible and monotone.

4.2.1 Uniform-Price Matching Functions

Recall that a quote is separating if an agent can correctly infer its tentative allocation. When bids are discrete, monotone, and divisible, a separating price is guaranteed to exist. In fact, there is typically a range of prices that satisfy the feasibility condition

$$\sum_{i \in \mathcal{I}} w_i(p) = 0.$$

The set of bids in the exchange set can be identified in the following manner. Let m denote the number of unit sell offers at or below \underline{p} , and n the number of unit buy offers at or above \bar{p} . Let $l = \min(m, n)$. The set of winning buy offers, B_{in} , is the l highest unit buy offers. The set of winning sell offers, S_{in} , is the l lowest unit sell offers. A uniform price auction applying these rules can use any tie-breaking rule and any algorithm for pairing the winning buy and sell bids. I justify restricting attention to this method of generating the transaction set based on the local efficiency argument. This procedure for identifying B_{in} and S_{in} can be extended to monotone, divisible bids simply by breaking the offer up into component single-unit bids.

Having selected the winning bids, we now turn our attention to setting the transaction price. The matching function γ^k , used in the k -double auction, sets the transaction price according to $p = k\bar{p} + (1 - k)\underline{p}$, where $k \in [0, 1]$. γ^k covers the full range of transaction prices that are uniform, are consistent with separating prices, and support the locally efficient allocation.

Notice that the two extreme values of k produce the $(M + 1)$ st and M th prices, respectively. Although no uniform-price sealed bid auction is Bayes-Nash incentive compatible for multi-unit buyers or sellers (Wurman et al., 1998a), the $(M + 1)$ st-price, sealed bid auction is incentive compatible for single-unit buyers, and the M th-price, sealed bid auction is incentive compatible for single-unit sellers. These results are incremental extensions to Myerson and Satterthwaite's (1983) famous result.

The Dual-Price mechanism (McAfee, 1992), uses the k -auction pricing but excludes

Time	Agent	Offer
t_1	Agent 1	sell at \$5
t_2	Agent 2	buy at \$8
t_3	Agent 3	buy at \$7
t_4	Agent 4	sell at \$6
t_5	Agent 5	buy at \$9

Table 4.1: A sequence of five bids.

the lowest-priced offer in B_{in} , and the highest-priced offer in S_{in} . I designate this matching function γ^{DP} .

The other uniform price auction mentioned in Section 4.1.1 is the UPDA. The procedure described in this section cannot be used to implement the 1S version of the UPDA because the auction does not guarantee that inclusion in the exchange set is ordered by highest bid for buyers, and lowest bid for sellers. Instead, membership in B_{in} and S_{in} is dependent on the chronological order bids are received. I designate this matching function γ^{S1} .

4.2.2 Discriminatory-Price Matching Functions

When we relax the constraint that prices be uniform, a wider range of pricing options becomes available. Table 4.1 provides an example with which we can compare the matching functions in this section. Here $t_1 < t_2 < t_3 < t_4 < t_5 < t_c$, where t_c is the time the clear occurs.

Local efficiency demands that agents 1 and 4 be the winning sellers, and agents 2 and 5 be the winning buyers. We can determine B_{in} and S_{in} in the manner discussed in the previous section. There are two possible transactions sets that can be formed from this combination of buyers and sellers. Under uniform pricing, all transactions occur at the same price. Thus, agents are indifferent between the two possible transaction combinations. However, as the following discussion shows, when prices are discriminatory, it matters a great deal how agents in the transaction sets are matched.

Pay Buyer's/Seller's Bid

There are several ways in which we might consider generalizing the first and second price rules. One is the uniform M th- and $(M + 1)$ st-price rules already discussed. An alternate generalization is to require that transactions occur always at the seller's price or always at the buyer's price. Consider the application of the buyer's-price and seller's-price policies

to the example in Table 4.1. Suppose the auction matches the highest buy offer with the lowest sell offer until the exchanges are exhausted. The *seller's-price* function, γ^{SP} , would result in transactions

agent 1 sells to agent 5 for \$5,
agent 4 sells to agent 2 for \$6.

The *buyer's-price* function, γ^{BP} produces

agent 1 sells to agent 5 for \$9,
agent 4 sells to agent 2 for \$8.

We can form a convex combination of the two extremes. Let $\kappa \in [0, 1]$. The price of the transaction between buyer, i , and seller, h , is $\kappa w_i + (1 - \kappa)w_h$. Note that, unlike the k -price matching function, the κ -price is computed for each pair of agents. This function is used in a study by Hu and Wellman (1998).

Chronological Pricing

Chronological pricing uses the bids' submission times to determine the transaction price. Given a transaction pair, *earlier-bid* pricing uses the price of the bid that was placed earlier, while *later-bid* pricing uses the price of the later bid. The former, designated γ^{ET} , generalizes the method in which prices are determined in the CDA to permit it to be used with clearing schedules other than bidder activity. The later-bid pricing, designated γ^{LT} , is simply γ^{ET} 's natural complement.

Consider the application of the earlier-bid pricing to the exchanges determined above. Earliest pricing computes the transactions:

agent 1 sells to agent 5 for \$5,
agent 4 sells to agent 2 for \$8.

If, instead, the auction formed the same matches but set the prices according to the later bid prices, the following transactions would occur:

agent 1 sells to agent 5 for \$9,
agent 4 sells to agent 2 for \$6.

	Transactions	
	1 sells to 5 for	4 sells to 2 for
M th	\$8	\$8
$(M + 1)$ st	\$7	\$7
Dual Price ($k = 1$)	\$8	excluded
Seller's price	\$5	\$6
Buyer's price	\$9	\$8
Earlier price	\$5	\$8
Later price	\$9	\$6

Table 4.2: Transaction prices for the six matching functions.

	Transactions	
	4 sells to 2 for	4 sells to 5 for
Seller's price	\$5	\$6
Buyer's price	\$8	\$9
Earlier price	\$8	\$6
Later price	\$5	\$9

Table 4.3: In discriminatory auctions, agent 4's transaction price depends on its trading partner.

4.2.3 Comparisons

Table 4.2 summarizes the different transaction prices for the seven pricing functions described to this point. The example problem was constructed to illustrate differences between these seven, and does not elicit distinguishing behavior from the γ^{1S} , so that function is excluded from the comparison.

As mentioned, the surplus an agent captures depends on how the transaction pairs are determined. Table 4.3 shows, for the four discriminatory pricing functions, how the price, and therefore agent 4's surplus, depends on who agent 4 trades with.

4.3 The 4-Heap Algorithm

In the study by McCabe, et al. (1993), the authors note that bids can be organized into four sets: included buyers, included sellers, excluded buyers, and excluded sellers. In fact, all of the matching functions described in Sections 4.2 can benefit from this scheme. In this section, I present a computationally efficient algorithm that maintains the bids in data structures that support the iterative updating of these sets.

The algorithm is applicable to all of the previously mentioned matching functions, but I present it first in the context of the k -auction matching function.

To support the core activities discussed in Section 3.2.2, an auction algorithm must implement the following operations:

Insert/Remove: place a new bid into the data structure, or remove a current one.

Price Quote: calculate the bid and ask quotes given the current set of bids.

Clear and Match: calculate a clearing price and remove all of the bids that match.

Note that when an agent modifies its bid, the auction can implement this by removing the agent's current bid and inserting a new one.

Perhaps the most straightforward implementation of the rules of Section 4.2.1 would be to maintain a sorted list of all bids, and perform clears by traversing the list to determine the M th and $(M + 1)$ st prices. Assume that we have $L = M + N$ single-unit bids in a sorted, doubly-linked list. Inserting a new bid takes $O(L)$ time. Generating a price quote takes $O(M)$ time. Removing a bid can be accomplished in constant time if a secondary mechanism (such as a hash table) is used to associate bidders with their bids. Clearing and matching is an $O(L)$ operation because we have to trace through all of the bids to find the ones that matched. We could do somewhat better—clearing in $O(\min(M, N))$ —by employing two lists, one for buy bids and one for sells. In either case, when the number of bids is small, a sorted-list algorithm might be fine. We can, however, do better for large M and N .⁷

4.3.1 Description of the Algorithm

My algorithm uses four heap structures to organize the bids. I distinguish bids by whether they represent buy or sell offers, and whether or not they are in the current match set. The four heaps are:

B_{in} : Contains all of the buy bids that are in the current match set. The heap priority is minimal price, so that the lowest priced bid is on top. The size of this heap is $O(\min(M, N))$.

B_{out} : Contains all of the buy bids that are *not* in the current match set. The heap priority is maximal price. The size of this heap is $O(N)$.

⁷Note that M and N are more likely to be large when we permit bid schedules that contain many price-quantity points.

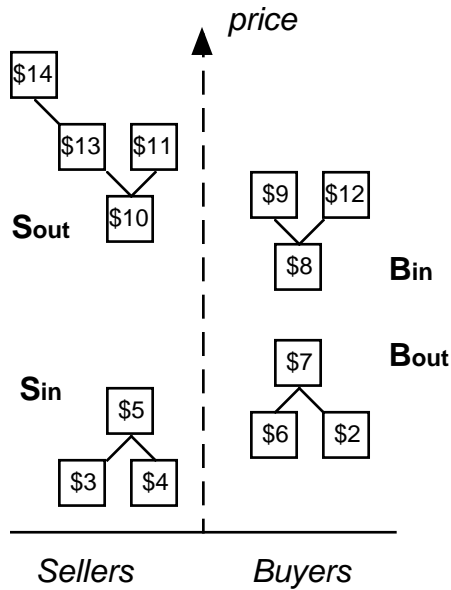


Figure 4.1: Schematic diagram of bids arranged in the four heaps.

S_{in} : Contains all of the sell bids in the current match set, prioritized by maximal price.

The size of this heap is $O(\min(M, N))$.

S_{out} : Contains all of the sell bids not in the current match set, prioritized by minimal price. The size of this heap is $O(M)$.

Figure 4.1 illustrates the relationships between the heaps.

Recall that a heap data structure is a complete binary tree, with the property that each node has a priority not exceeding its parent's (Cormen et al., 1990). Inserting a new node into the heap (`put`), or removing the top node (`get`) each take $O(\lg K)$ time, where K is the size of the heap.

The 4-HEAP algorithm ensures several constraints among the heaps. First, the number of units in B_{in} must equal the number in S_{in} . The rest of the constraints are characterized in terms of the top nodes of each heap. Let b_{in} , b_{out} , s_{in} , and s_{out} be the top nodes of B_{in} , B_{out} , S_{in} , and S_{out} , respectively. $Value(n)$ is the value of node n . The algorithm enforces the following constraints:

- $Value(b_{in}) \geq Value(b_{out})$,
- $Value(s_{out}) \geq Value(s_{in})$,
- $Value(s_{out}) > Value(b_{out})$,
- $Value(b_{in}) \geq Value(s_{in})$.

```

if ((Value( $s_{new}$ )  $\leq$  Value( $b_{out}$ )) and
(Value( $s_{in}$ )  $\leq$  Value( $b_{out}$ )))
    put( $s_{new}$ ,  $S_{in}$ )
     $b \leftarrow$  get( $B_{out}$ )
    put( $b$ ,  $B_{in}$ )
else if (Value( $s_{new}$ )  $<$  Value( $s_{in}$ ))
     $s \leftarrow$  get( $S_{in}$ )
    put( $s$ ,  $S_{out}$ )
    put( $s_{new}$ ,  $S_{in}$ )
else
    put( $s_{new}$ ,  $S_{out}$ )

```

Figure 4.2: Pseudocode for receiving a new sell bid.

4.3.2 Complexity Analysis

The complexity of bookkeeping in the 4-HEAP algorithm comes from the need to keep B_{in} and S_{in} the same size while performing inserts and removes. To simplify the description, I restrict attention to single-unit bids. The extension to multi-unit bids is relatively straightforward, and is discussed in Section 4.3.4.

Insert

When a new bid comes in, the algorithm may need to do more than place it in one of the heaps. Using the example depicted in Figure 4.1, if the auction receives a sell offer for \$6, not only would it be placed onto S_{in} , but the top bid in B_{out} must be transferred into B_{in} to equilibrate the *in* heaps. This requires one **get** from B_{out} , one **put** into S_{in} and one **put** into B_{in} . Receiving a buy bid also requires as many as three heap operations. Thus, the insert operation for an arbitrary bid is bounded by $O(\lg L)$.

In general, when a new sell bid, s_{new} , arrives, there are three possible actions. Either the new bid forms a new match with the top bid in B_{out} , or it displaces a bid in S_{in} , or it is placed into S_{out} . The pseudocode is shown in Figure 4.2.

The logic for new buy bids is similar, with all of the Bs and Ss switched, and the inequalities reversed.

Remove

To remove bids efficiently, we employ an external lookup mechanism, such as a hash table, to locate the bid within its containing heap in constant time. We can then delete this

node from its heap in logarithmic time. If the bid is in one of the *out* heaps, we are finished. However, if it is in S_{in} or B_{in} , we must also transfer the top bid from the other *in* heap to its corresponding *out* heap. Thus, in the worst case, removing a bid requires three heap operations, for a total time bounded by $O(\lg L)$.

Clears and Quotes

A price quote can be generated simply by inspecting the tops of the heaps. The bid quote, or $(M + 1)$ st price, is $\max(\text{Value}(s_{in}), \text{Value}(b_{out}))$. The ask quote, or M th price, is $\min(\text{Value}(s_{out}), \text{Value}(b_{in}))$. The constraints on the heaps ensure that the ask quote is above the bid quote. Calculating a price quote is a constant time operation.

Clearing prices for the M th-price rule are set to the ask price. The $(M + 1)$ st auction clears at the bid price. We match buyers and sellers by disassembling the *in* heaps, B_{in} and S_{in} . Matching necessarily takes time proportional to the number of bids matched, which is $O(\min(M, N))$.

4.3.3 Discriminatory Price Auctions

As stated in Section 4.2.2, the method for forming matches has a large impact on the distribution of surplus in discriminatory matching functions. Therefore, we may wish to put some structure on the way transaction pairs are formed. The default pairings that the 4-HEAP algorithm generates are artifacts of the heap maintenance.

Here are three systematic methods of forming trading pairs:

- *Chronological pairing* matches the earliest buy bids with the earliest sell bids in the transaction sets, until the sets are exhausted.
- *Reverse chronological pairing* matches the earliest buy bids with the latest sell bids, until the sets are exhausted.
- *High-low pairing* matches the highest stated buy offer with the lowest stated sell offer until the sets are exhausted.

The 4-HEAP algorithm must be enhanced in order to efficiently form transaction pairs according to one of these disciplines. For example, the high-low pairing can be implemented by repeated application of the `get` operator, in $O(L \lg L)$ time. In the worst case, we simply decompose the heaps and resort the bids according to the priorities implied by the discipline.

4.3.4 4-Heap and Multiple-Unit Auctions

The 4-HEAP algorithm can be extended to handle bids expressed as price-quantity step functions by permitting nodes to represent multiple units. However, the algorithm is complicated by the fact that bids transferred between *in* and *out* heaps may need to be split in order to maintain an exact equivalence in the number of units stored in the two *in* heaps. Similarly, insertion or removal of a multi-unit bid may entail several nodes be transferred across the complementary pair of *in* and *out* heaps.

A simple example illustrates the point. Agent 1 submits a bid to sell two units at \$3. The auction places this bid into S_{out} . Agent 2 now submits a bid to buy one unit at \$5. The auction puts the new bid into B_{in} , and splits the first bid, moving one unit into S_{in} and leaving one unit in S_{out} . When the auction receives a third bid, from agent 3 to buy two units at \$4, it moves the remaining unit of the first bid from S_{out} to S_{in} , and splits the third bid between B_{in} and B_{out} .

It is easy to see that any multi-unit bid may, over time, be split entirely into single-unit bids. Hence, the worst-case complexity of bid operations in the 4-HEAP algorithm must be characterized in terms of number of units, rather than number of bids.

4.3.5 Tradeoffs between Operation Costs

To summarize the above analysis, the 4-HEAP algorithm can process bids (insert and remove) in $O(\lg L)$ time, issue price quotes in $O(1)$ (constant) time, and, when uniform pricing is used, perform clears in $O(\min(M, N))$ time.

As a benchmark, I introduce the *simple sealed-bid* algorithm, which makes a different tradeoff between the computational costs of the respective operations. This algorithm works as follows. As each bid is received, append it to one of two unordered lists, representing the buy and sell bids. This is a constant time operation. To clear the auction, sort the buy offers in descending order by price, and the sell offers in ascending order. While the top buy offer is greater than or equal to the top sell offer, remove the front element of each list and place them in the transaction set. The computational complexity of this algorithm is in terms of R , the total number of price points in the bids (in contrast to the number of units). For instance, an agent who bids to sell two units at \$5 and two at \$10, has submitted one bid which has two price points and offers four units. The clear is an $O(R \lg R)$ operation. A price quote is the same as a clear, except that the original lists are not destroyed.

When bids are restricted to single units, $R = L$, and both the 4-HEAP and simple sealed-bid algorithms require $O(L \lg L)$ total time over the auction's life cycle, assuming a constant number of quotes and clears, and a constant number of bid revisions per unit bid. The simple sealed-bid algorithm is most appropriate for single-clear, sealed-bid auctions, since price quotes are relatively expensive. The 4-HEAP algorithm offers reduced clear-time latency at the expense of more work during the bid processing stage, and will be clearly superior in settings demanding frequent price quotes. I expect these observations hold for reasonably large sized bids and moderate bidding activity.

However, when there is a large variation in the sizes of bids, and a high frequency of withdraws and edits, it is possible that the simple sealed-bid algorithm will outperform the 4-HEAP algorithm even in auctions with frequent price quotes and multiple clears.

4.4 Matching Functions with Indivisible Bids

So far in this chapter we have consider single and multi-unit divisible bids. This section completes the one-dimensional picture with a brief discussion of the implications of permitting indivisible bids.

Suppose all of the offers are indivisible. The per-unit price that agent i would be willing to buy (sell) z units for is given by $w_i^{-1}(z)$. Note that a negative value of z expresses a sell offer, and a quantity that i has not expressed a value for has $w_i^{-1}(z) = 0$. The surplus maximization problem can be stated:

$$\begin{aligned}
 \max \quad & \sum_i \sum_z z w_i^{-1}(z) \delta^{iz} & (4.1) \\
 \text{s.t.} \quad & \sum_i \sum_z z \delta^{iz} \leq 0, \\
 & \sum_z \delta^{iz} \leq 1, \forall i, \\
 & \delta^{iz} \in \{0, 1\}.
 \end{aligned}$$

where $\delta^{iz} = 1$ means that i 's offer for z units is part of the solution. The first constraint states that we do not allocate more than we have (and can dispose of any extra supply at no cost). The second constraint ensures that each agent wins at only one quantity.

When there is only one sell offer, and it has an associated reserve price of zero, (4.1) reduces to a 0-1 knapsack problem (Garey and Johnson, 1979). Therefore, (4.1) is also

Agent	Offer	Divisible?
Agent 1	sell z units at \$1	Yes
Agent 2	buy 1 unit at p_2	Yes
Agent 3	buy z units at p_3	No

Table 4.4: A set of bids that can lead to arbitrarily bad outcomes when $p_2 > p_3$ and the greedy algorithm is used.

NP-complete.

In practice, several online auction services which allow indivisible bids in a single seller auction implement a greedy approximation algorithm, γ^{greedy} . Bids are sorted decreasing in price, with ties broken in favor of quantity and submission time. The algorithm traverses the list, accepting offers until supply is reached, and skipping offers which cannot be satisfied from the remaining supply.

In the worst case, this algorithm can lead to arbitrarily bad allocations. Consider the situation in Table 4.4. When $p_2 > p_3$, the greedy algorithm will select agent 2's offer first, and we will be unable to satisfy agent 3's offer. If $zp_3 > p_2$, then we will miss $zp_3 - p_2$ surplus. We can make this arbitrarily bad by raising z or p_3 .

4.5 Conclusion

In this chapter I have recapped the well known k -price matching function, and generalized versions of common discriminatory matching functions to fit into the parametrization framework. I have also presented the 4-HEAP algorithm, a computationally efficient algorithm for managing the sets of tentatively winning bids in an iterative one-dimensional auction with monotone, divisible bids for discrete resources.

Chapter 5

Multidimensional Auctions for Discrete Resources

5.1 Overview

The recent U.S. FCC auctions in electromagnetic spectrum have rekindled economists' interest in mechanisms that allocate heterogeneous, discrete resources when complementarities exist. The FCC and its consultants expected that the licensees would see significant synergies from operating the same spectrum bands in geographically contiguous regions of the country, or contiguous bands in the same region. McMillan (1994) reports that, at the time, the mechanism designers employed by the FCC did not feel that combinatorial auctions were well-developed enough to use in practice. Instead, they developed the Simultaneous Ascending Auction (discussed in more detail in Section 5.2.3), which uses eligibility rules and a multi-round format to (hopefully) extract enough information to facilitate coordination.

Considerable effort has been focused on the problem in the last few years. The work described in this chapter contributes to the understanding of auctions that *do* permit bundle bidding. Because price equilibrium are not guaranteed to exist, I consider the question of whether (less-restrictive) payment equilibria exist. This chapter shows that anonymous payment equilibria always exist that support the efficient allocation, even when both complementary and substitutable preferences are permitted. This result leads to an innovative family of multidimensional auctions for the single-seller, discrete-resource case.

Section 5.2 discusses the state of the art in the allocation of heterogeneous discrete resources. Section 5.3 shows how to construct a separating payment lattice that supports the efficient allocation when agents' valuations are monotone on the lattice. This construction serves as the basis for a family of auctions, including the Ascending $k = 1$ Bundle

	A	B	AB
Agent 1	0	0	3*
Agent 2	2	2	2

Table 5.1: An example without equilibrium prices for individual goods.

Auction (A1BA), described in Section 5.4. Finally, Section 5.5 reports initial simulation results with myopic agents in A1BA.

5.2 Existing Results

I categorize the research related to the allocation of heterogenous discrete resources into three subtopics. The first is focused on identifying conditions under which price equilibria exist. The second subtopic is the extension of Vickrey’s original results to an incentive compatible direct revelation mechanism called the Generalized Vickrey Auction (GVA). The third subtopic contains iterative auctions proposed as alternatives to the direct revelation mechanism.

5.2.1 Existence of Price Equilibria

Discreteness does not necessarily rule out price equilibria. For example, it has been shown that price equilibria exist if a *gross substitutability* condition holds (Kelso and Crawford, 1982), or if utility functions satisfy the *no-complementarities* condition (Gul and Stacchetti, forthcoming). Bikhchandani and Mamer (1997) demonstrated that price equilibria exist iff the total value of the solution to the discrete allocation problem is equal to the value of the corresponding relaxed linear optimization problem.

However, as discussed in Section 2.4.2, it is easy to construct examples in which price equilibria fail to exist. That example is reproduced in Table 5.1. The efficient allocation is indicated with an asterisk. There are no prices on the two objects individually which balance supply and demand.

5.2.2 Generalized Vickrey Auction

The GVA is an incentive compatible, efficient, and individually rational direct revelation mechanism (MacKie-Mason and Varian, 1994). Each agent submits its utility function, and the auction computes the optimal allocation and charges each agent a payment. The

agent's payment is the impact that its presence has on the welfare of the other agents,

$$\pi^i = V(\mathbf{f}^{*(\mathcal{I}\setminus i)}) - [V(\mathbf{f}^*) - v_i(\mathbf{f}_i^*)],$$

where $\mathbf{f}^{*(\mathcal{I}\setminus i)}$ is the optimal solution to the problem of allocating the resources when i is excluded.

The GVA extends the intuition gained from Vickrey's (1961) seminal work, and results by Clarke (1971) and Groves (1973) in general allocation problems.¹ The mechanism's incentive compatibility property follows from the fact that an agent's bid determines what it gets, but not how much it pays (or receives).

The GVA has several drawbacks that inhibit its use in practice. One is that the GVA is not guaranteed to always be budget balanced. In particular, it is sometimes necessary to subsidize the auction in order to get the desired properties. However, in some restricted problem domains, the GVA is guaranteed to not run a deficit. The distributed scheduling problem described in Section 2.5 is one such domain (Wellman et al., 1999).

A second drawback, highlighted by Banks et al. (1989), is that the GVA requires that each participant specify a complete utility function, which (potentially) requires that she report $2^n - 1$ values. If the preferences cannot be expressed in a compact form, this requirement may make the GVA impractical.

In addition, in many cases it may be costly for agents to determine their true valuation for every possible combination of items, the majority of which play no role in determining the solution. Parkes et al. (1998) argue that the cognitive costs of accurately determining one's true valuation is a major factor in the predominance of the English auction online—a participant needs an accurate appraisal of its value for a good only if it is still in the running near the end of the auction. This problem is exacerbated in a combinatorial auction.

A third objection to using the GVA in practice is that, even in the one-sided case, it computes discriminatory payments. This can be seen in the example in Table 5.2 in which the items A and B could be two different units of the same resource. The GVA payments are $\pi^1 = \$3$, and $\pi^2 = \$2$. In many situations, such discriminatory practices are illegal or unacceptable to the bidders.

One of the roles that markets play in society is that of *price discovery*. This is partic-

¹The general class of mechanisms derived from the independent work of these three authors are called Vickrey-Clarke-Groves mechanisms. The GVA is a specific instance of the class.

	A	B	AB
Agent 1	5*	5	7
Agent 2	5	5*	8

Table 5.2: An example in which the GVA payments are discriminatory.

ularly important in repeated interactions, or where the resulting prices are of interest to agents not in the immediate allocation problem (e.g. stockholders typically watch stock prices even though they aren't actively trading on a given day). The GVA produces payments that are associated with agents, not bundles. Thus, it does not facilitate price discovery.

Finally, because the GVA results in nonlinear, discriminatory prices, it can be manipulated by coalitions and bids submitted under false names. These concerns are especially relevant on the Internet (Sakurai et al., 1999), where communication is inexpensive, and identities are easily manufactured. Thus, the desirable incentive properties of the GVA rely on assumptions that are likely to not hold in practice. Without incentive compatibility, we also lose our guarantees of efficiency.

5.2.3 Combinatorial Auction Designs

Several alternatives to the GVA have been suggested for the allocation of heterogeneous discrete goods with complementarities. All of the auctions discussed in this section involve a single-seller, and all but one are iterative. None of them are incentive compatible or guaranteed to find efficient allocations.

The majority of the auctions use a bid dominance and/or beat-the-quote rule. Cramton (1998) provides some general arguments in favor of ascending auctions. I reiterate a few of the more salient arguments, enhanced to take advantage of the broad view of auction features developed earlier in this thesis. The primary advantage of ascending auctions is that they reveal information which can focus the attention of the bidders on bundles that are likely to become part of the final allocation. It seems likely that relatively efficient allocations can be found without requiring full specification of an agent's utility function. In addition, this information reduces uncertainty and allows an agent to adapt its bidding strategy accordingly. When used with anonymous prices (payments), ascending auctions facilitate price (payment) discovery, which is especially important when the valuations are uncertain (and possibly correlated), or when the environment involves repeated interactions.

We can use features defined in Chapter 3 to further classify these auctions. One major distinction among the candidates is whether or not they permit bids on bundles.² In general, when bundle bids are allowed, computing the locally efficient allocation is NP-hard. The auctions discussed in this section either undertake that calculation or implement some tractable approximation. In addition, the auction's quote can be either linear prices or payments associated with bundles. These price quotes may be anonymous or discriminatory.

The motivation for permitting bundle bidding is to combat the *exposure problem*: without bundle bids, agents who have superadditive valuations are forced to either bear risk or bid over-cautiously. Suppose, for example, agent i has a valuation for the bundle AB that exceeds the sum of its value for A and B alone. If agent i offers no more for either A or B than their individual values, the agent may fail to purchase the bundle even when the combined cost is under its valuation. On the other hand, if i is less conservative, it can offer more than its value for one of the goods. This strategy exposes the agent to potential losses if, in the end, it gets only one item and pays more than its value for the item.

Allowing bundle bids introduces the *free rider problem* (Rothkopf et al., 1998), in which bidders buying smaller bundles need to collaborate to displace a larger bundle bidder. Consider the situation where agent 1 values A at \$5, agent 2 values B at \$4, and agent 3 values AB at \$7. Suppose agent 3 has offered \$6 for AB. Neither agent 1 nor agent 2 can displace agent 3 without taking a loss. In order to displace agent 3, the two agents have to collaborate. However, the more that agent 2 bids, the less agent 1 will have to contribute to displace agent 3. The same logic holds for agent 2. Thus, the two agents have an incentive to free ride on each other, which could lead to a coordination failure and an inefficient outcome.

Rassenti, Smith, and Bulfin (1982)(Rassenti et al., 1982) investigated a single-sided, sealed-bid auction that allowed agents to bid on bundles. The auction computed a locally efficient allocation from these bids. A secondary market was provided that allowed participants to sell bundles to one another to recapture some of the lost efficiency. The authors were investigating the allocation of airport slots. Thus, they were interested in generating information from one allocation period that could be used to guide bidding in the next. After the initial allocation, the auction announced two price vectors derived from relaxed linear programs. One vector contained the lowest prices that excluded all non-winning

²Bundle bidding is the motivation for introducing bids in payment space in Chapter 3.

bids, and the other vector contained the highest prices that included all winning bids. This information could be used by the participants in the next period.

The game theorists hired by the FCC developed the Simultaneous Ascending Auction (SAA) (McMillan, 1994; McAfee and McMillan, 1996), which received very positive press when it generated billions of dollars of revenue for the U.S. government. The SAA allocates resources by operating one first-price auction for each resource. The auctions are synchronized, and the mechanism enforces an *eligibility* constraint that restricts agents from bidding on more items than they had “active” bids (i.e. winning or newly submitted) in the previous round.³

Banks et al. (1989) introduced an iterative mechanism, called the Adaptive User Selection Mechanism (AUSM), that permits bids on bundles. AUSM posts the current best allocation on a bulletin board visible to all of the participants. To become part of the current best allocation, a new bid has to offer more than the sum of all of the bids it displaces. Bids that are not part of the best allocation are posted in a standby queue designed to facilitate the coordination of two or more smaller bidders combining on a new bid large enough to displace a larger bidder. However, posting bids on a standby queue reveals information about the bidders that they may prefer not to reveal.

Both SAA and AUSM have complex strategy spaces and have so far proved intractable to game-theoretic analysis. Ledyard et al. (1997) executed a series of controlled experiments to investigate the performance of three mechanisms: sequential ascending bid auctions, the SAA, and AUSM. In problems the authors classified as “hard”—involving significant complementarities and a range of solutions—AUSM significantly outperformed the other two mechanisms. This led the authors to conclude that, when there are complementarities, mechanisms that allow package bidding will perform better. This seems to have been taken to heart by the FCC and its consultants, and the current recommendation is that the FCC should explore mechanisms that allow agents to bid on bundles (CRA, 1998).

³Recent events have cast a large shadow over the FCC auction results. Several of the original winners have since defaulted on their obligations, and the FCC has been forced to reauction large numbers of licenses, typically getting much less revenue. For example, 339 C-Block PCS licenses that originally sold for \$3.8 billion recently fetched a mere \$410 million (Culver, 1999). This evidence suggests that the task of estimating market value of a license is indeed difficult and prone to error. Moreover, companies’ valuations for these licenses were probably more accurately modeled as correlated rather than independent, and thus subject to the *winner’s curse* (Milgrom, 1989). The winner’s curse is a well known phenomenon when agents have correlated values. If the item has some actual value, and the agent’s expected values are distributed around the actual value, then the more bidders, the more likely that the winner will have offered more than the actual value of the item. Thus, the agent wins the auction, but loses money.

Recently, mechanism designers at Cal Tech proposed the Resource Allocation Design (RAD) (DeMartini et al., 1998), which combines features of AUSM and SAA. RAD allows agents to place bids on bundles, then computes the locally efficient allocation. The auction solves a linear program to generate approximate prices, and enforces a beat-the-quote rule. In addition, agents need to maintain eligibility by continuing to win items, or by submitting new bids.

In experimentation with Cal Tech students, RAD averaged more efficient solutions in fewer iterations than either AUSM or SAA. However, RAD has shortcomings of its own. One problem that plagued both SAA and RAD was that some bidders lost money. This turned out to be a side effect of the eligibility rule. In order to maintain eligibility in future rounds, participants often placed low bids on items they didn't really want, but which they ended up winning anyway. A second drawback of RAD is that the prices computed by the linear program are not guaranteed to be separating prices. This makes it difficult for agents to determine whether they are winning.

Another recently proposed ascending auction is *iBundle* (Parkes, 1999). *iBundle* allows bids on bundles and computes the locally efficient allocation. It associates payments with bundles, in effect, announcing a payment lattice as a price quote. There are three variations of *iBundle*, which differ based on the manner in which payments are computed: *iBundle*(2) announces anonymous payments to every agent, *iBundle*(3) announces discriminatory payments, and *iBundle*(d) announces discriminatory payments for some agents, and anonymous payments for the rest. In all three versions, the auction ends when no agents submit new bids, and the winners pay the price of their bid.

The combinatorial auctions mentioned in this section have very complex strategy spaces. We already know that price equilibria may not exist when agents have complementary preferences. A natural question is whether we can relax the equilibrium condition and always find a lattice of equilibrium payments that support the efficient allocation.

In both RAD and *iBundle*, the winners pay the exact amount of their bids. In general, charging agents their bids may not correspond to a payment equilibria—at the final payments, agents may wish to purchase a bundle other than their allocation.

Consider a simple example where agent 1 has bid \$5 for A, \$4 for B, and \$7 for the bundle AB. Agent 2 offers \$2 for A, \$3 for B, and \$6 for AB. The bids are diagrammed in Table 5.3. The optimal allocation assigns A to agent 1 and B to agent 2. Suppose we have reached the end of the auction and that these bids actually represent the agents' true valuations for the items. If we charge each agent its bid, then agent 1 will pay \$5

	A	B	AB
Agent 1	5*	4	7
Agent 2	2	3*	6

Table 5.3: An example in which charging bid payments is not an equilibrium.

for A, and agent 2 will pay \$3 for B. At these payments, agent 1 would rather buy B than A, because that would provide it with a surplus of \$1, whereas purchasing A for \$5 leaves it no surplus.⁴ Thus, the agent has an incentive to withhold information about its valuation for A—it would rather either pay less for A, or have the auction determine an allocation that has it winning B. In larger problems this disincentive to reveal information can further complicate the agent’s basic strategic problem.

In the rest of this chapter, I develop an alternative ascending auction that allows bids on bundles and has the property that the payments announced at the end of the auction correspond to a payment equilibrium (as defined in Section 2.3.2). Moreover, because these payments are anonymous, they represent a form of price discovery and establish a meaningful social value for bundles. The first task is to show that a payment equilibrium that supports the efficient allocation always exists.

5.3 Equilibrium Payments

5.3.1 Construction

Recall from Chapter 2 that \mathbf{f}^* is the efficient allocation, and $v_i(b)$ is agent i ’s valuation for bundle b . I consider the existence of payment equilibria that support the efficient allocation when agents have valuations that are monotone on the finite lattice \mathcal{B} . I show that such equilibrium payments always exist by constructing them in a three-step procedure. First, find the efficient allocation. Next, compute payments on the bundles that are allocated in \mathbf{f}^* . Finally, compute payments on the unallocated bundles.

Step 1. The socially efficient allocation satisfies the following maximization problem:

⁴It is unclear whether AUSM or RAD could arrive at this situation. The descriptions available are not precise about what happens to bids when they are displaced.

$$\begin{aligned}
\max \quad & \sum_i \sum_b v_i(b) \delta^{ib} & (5.1) \\
\text{s.t.} \quad & \sum_b \delta^{ib} \leq 1, \forall i, \\
& \sum_b \sum_i \delta^{ib} b_j \leq 1, \forall j, \\
& \delta^{ib} \in \{0, 1\},
\end{aligned}$$

where $\delta^{ib} = 1$ iff b is allocated to i , and $b_j = 1$ iff $j \in b$. The first constraint implies that each agent receives at most one bundle. The second constraint ensures that no item is allocated more than once.

Let \mathbf{f}^* be a solution to this integer program. The value of the solution is $V(\mathbf{f}^*)$.

Step 2. Recall that \mathcal{I}_Θ is the set of agents who receive items in \mathbf{f}^* , that is, $\mathcal{I}_\Theta = \{i \mid \mathbf{f}_i^* \neq \emptyset\}$. The set $\mathcal{I}_{-\Theta}$ denotes the agents who receive nothing, $\mathcal{I}_{-\Theta} = \mathcal{I} \setminus \mathcal{I}_\Theta$. Let \mathcal{B}_Θ be the subset of bundles which are assigned in \mathbf{f}^* , and \mathcal{B}_\emptyset be the unassigned bundles. For each agent $i \in \mathcal{I}_{-\Theta}$, introduce a null item ϕ_i which represents the agent's null allocation. Let $\Phi \equiv \{\phi_i \mid i \in \mathcal{I}_{-\Theta}\}$. Every agent has a valuation of zero for every element of Φ .

Let $G = \mathcal{B}_\Theta \cup \Phi$, and $g \in G$. G , along with the corresponding agent valuations, describes an assignment problem (Shapley and Shubik, 1972). I refer to it as the assignment subproblem because in the process of formulating it, we have discarded information about the original allocation problem. Note also, that we already have the solution, \mathbf{f}^* , from Step 1.

We now compute π_g , for all g , that support the solution to the assignment subproblem. To accomplish this, I employ the dual program used by Leonard (1983) to compute minimal payments for assignment problems. Let LP_{lower} be the following linear program:

$$\begin{aligned}
\min \quad & \sum_g \pi_g & (5.2) \\
\text{s.t.} \quad & s_i + \pi_g \geq v_i(g), \forall i, g, \\
& s_i, \pi_g \geq 0, \\
& \sum_i s_i + \sum_g \pi_g = V(\mathbf{f}^*).
\end{aligned}$$

The last constraint ensures that the first constraint is satisfied at equality for the optimal

assignment.

The s_i term represents the surplus achieved by agent i . LP_{lower} maximizes each agent's surplus within the range of equilibrium payments that support the optimal solution to the assignment subproblem. Note that the introduction of the items in Φ is simply a trick to include the agents in $\mathcal{I}_{-\Theta}$ in the assignment subproblem. In the solution, $s_i = 0$ and $\pi_{\phi_i} = 0$ for all $i \in \mathcal{I}_{-\Theta}$.

LP_{lower} has a complementary program, LP_{upper} , which computes upper bound payments:

$$\begin{aligned}
\min \quad & \sum_i s_i & (5.3) \\
\text{s.t.} \quad & s_i + \pi_g \geq v_i(g), \forall i, g, \\
& s_i, \pi_g \geq 0, \\
& \sum_i s_i + \sum_g \pi_g = V(\mathbf{f}^*).
\end{aligned}$$

Clearly the payment vector produced in finding a solution to either LP_{lower} or LP_{upper} is a payment equilibrium for the assignment subproblem—the first constraint requires that an agent cannot gain any more surplus from another assigned bundle than it receives from the one allocated to it.

Step 3. Given a solution to either LP_{lower} or LP_{upper} , the next step is to set payments on the bundles in \mathcal{B}_\emptyset . This can be done quite simply using the surpluses calculated in Step 2. For all $b \in \mathcal{B}_\emptyset$,

$$\pi_b = \max_i [v_i(b) - s_i]. \quad (5.4)$$

5.3.2 Properties

Theorem 5.1 *The bundle payments, $\underline{\pi}^*$, computed by LP_{lower} and (5.4), support the efficient allocation.*

Proof: We began the construction from the optimal allocation, \mathbf{f}^* . The question remains whether the constructed payment, $\underline{\pi}^*$, support \mathbf{f}^* . A solution to LP_{lower} has the property that \mathbf{f}_i^* maximizes i 's surplus among the bundles in G . Koopmans and Beckmann (1957) show that an integer solution to the assignment problem always exists. By the Duality Theorem of Linear programming, we can support this integer solution with payments.

For all other bundles, equation (5.4) sets the payment on the unallocated bundle b such that $\underline{\pi}_b^* \geq v_i(b) - s_i$. Therefore, no bundle provides more surplus to i than its assignment. \square

Theorem 5.2 *The bundle payments, $\overline{\pi}^*$, computed by LP_{upper} and (5.4), support the efficient allocation.*

Proof: The proof for Theorem 5.1 holds when LP_{upper} is used in place of LP_{lower} . \square

These payments satisfy the same monotonicity constraint imposed on valuations.

Theorem 5.3 *The payment lattice computed by LP_{lower} and (5.4), or by LP_{upper} and (5.4), is monotone when agent valuations are monotone on \mathcal{B} .*

Proof: Condition (5.4) implies that, for all $i, b \in \mathcal{B}_\emptyset$,

$$\pi_b \geq v_i(b) - s_i,$$

which can be written,

$$s_i + \pi_b \geq v_i(b). \tag{5.5}$$

The first constraint in LP_{lower} (LP_{upper}) imposes the same condition on all $b \in \mathcal{B}_\emptyset$.

Therefore, (5.5) holds for all i and b . Let h be the agent that maximizes π_b . That is, $h = \arg \max_i [v_i(b) - s_i]$. Thus, $s_h + \pi_b = v_h(b)$. When valuations are monotone, $c \supset b$ implies that $v_h(c) \geq v_h(b)$. Therefore

$$s_h + \pi_c \geq v_h(c) \geq v_h(b) = s_h + \pi_b,$$

which reduces to

$$\pi_c \geq \pi_b. \square$$

Let $\overline{\pi}^*$ be a lattice of payments calculated by LP_{upper} and (5.4). Similarly, $\underline{\pi}^*$ is the payment lattice resulting from solving LP_{lower} and applying (5.4). I now consider a range of equilibrium payments that support the optimal allocation.

Theorem 5.4 *For all $k \in [0, 1]$, $k\overline{\pi}^* + (1 - k)\underline{\pi}^*$ is a payment equilibrium.*

Proof: Let b be (one of) agent i 's most preferred bundles at $\bar{\pi}^*$, and c be some other bundle. Because $\bar{\pi}^*$ and $\underline{\pi}^*$ both support the same efficient allocation, i must not prefer c to b at $\underline{\pi}^*$. Formally,

$$v_i(b) - \bar{\pi}_b \geq v_i(c) - \bar{\pi}_c,$$

and

$$v_i(b) - \underline{\pi}_b \geq v_i(c) - \underline{\pi}_c.$$

In both cases, the relation is invariant to positive scalar transformations. Thus, for $k \in [0, 1]$,

$$k[v_i(b) - \bar{\pi}_b] \geq k[v_i(c) - \bar{\pi}_c],$$

and

$$(1 - k)[v_i(b) - \underline{\pi}_b] \geq (1 - k)[v_i(c) - \underline{\pi}_c].$$

Adding the two equations and simplifying gives

$$v_i(b) - [k\bar{\pi}_b + (1 - k)\underline{\pi}_b] \geq v_i(c) - [k\bar{\pi}_c + (1 - k)\underline{\pi}_c]. \square$$

The k parameter here is directly analogous to the parameter used in the k -double auction (Satterthwaite and Williams, 1989). $\bar{\pi}^*$ and $\underline{\pi}^*$ bound the range of payments for which supply equals demand. Reducing any payment in $\underline{\pi}^*$ would create excess demand for some of the items. There is a slightly weaker analogy for the upper bound side. We cannot raise the payment of any $b \in \mathcal{B}_\Theta$ without disequilibrating supply and demand. However, in some cases we can increase the payments of bundles in \mathcal{B}_\emptyset without adverse affects.

To my knowledge, the only other study of payment equilibrium in combinatorial auctions is a recent paper by Bikhchandani and Ostroy (1998). B&O thoroughly analyzed the competitive equilibrium properties of what they call the *package assignment* model. In this problem, buyers and sellers exchange packages of goods. Their formulation has the restriction that buyers are allowed to request only one package from each seller, and

	A	B	C	AB	BC	AC	ABC
Agent 1	4	4	4.25	7.5*	7	7	9
Agent 2	4	4.25	4*	7	7	7.5	9

Table 5.4: An example from Bikhchandani & Ostroy.

	A	B	C	AB	BC	AC	ABC
p^1	3.5	3.4	3.75	6.9	6.5	6.5	9
p^2	3.5	3.75	3.4	6.5	6.5	6.9	9

Table 5.5: Third-order equilibrium payments.

sellers are allowed to earmark only one package for each buyer. B&O present four different “order” assignments which allow decreasing levels of repackaging by the auctioneer. In a first-order assignment the auctioneer can arbitrarily repackage and relabel bundles, whereas in a fourth-order assignment the auctioneer can do neither. B&O present four bundle pricing functions (the first of which is linear and the rest are nonlinear), and show that a j th-order assignment along with a j th-order pricing function supports a Walrasian equilibrium (i.e. each agent is maximizing its utility with respect to the prices it sees) iff it is a solution to the associated linear programming problem.

B&O also show that for the single seller model, third-order Walrasian equilibrium exist, but second-order equilibrium may not. In their terminology, third-order equilibrium prices require a separate price vector for each buyer, whereas second-order pricing has one nonlinear vector presented to all parties and is equivalent to what I call anonymous payments. An example from their paper is shown in Table 5.4.

B&O’s equilibrium model requires that the seller designates one package for each buyer. In this example, in order for the seller to be maximizing its utility with respect to the prices, separate price vectors are needed for each buyer, as shown in Table 5.5.

B&O’s results seem at odds with my results, but the contradiction is explained by the difference in the allowable actions in the two models. In contrast to B&O’s model, my model does not allow the seller to earmark bundles for buyers. Instead, I assume the seller has consigned the objects to the auctioneer, who then redistributes it to the buyers. For the example in Table 5.4, $(3.4, 3.6, 3.4, 6.65, 6.4, 6.9, 8.4)$ is an anonymous payment lattice, computed using my method, that supports the assignment where agent 1 receives AB and agent 2 receives C. I have fixed the payment for C at \$3.4 to make the payments comparable to B&O’s. Notice that the seller receives less revenue (\$10.05 versus \$10.3) under my scheme than B&O’s. In some sense, this is the cost of producing anonymous prices.

	A	B	C	AB	AC	BC	ABC
Agent 1	6	6	5*	10	7	8	12
Agent 2	3	3	2	8*	5	6	11
Agent 3	4	2	1	7	6	5	10

Table 5.6: An example with three agents.

	AB	C	ϕ_3
Agent 1	10	5	0
Agent 2	8	2	0
Agent 3	7	1	0

Table 5.7: The assignment subproblem.

5.3.3 Example

Consider the example in Table 5.6. The efficient allocation is to assign AB to agent 2 and C to agent 1. Agent 3 gets nothing. This allocation has a social welfare of 13.

In order to construct equilibrium payments, we introduce the dummy good ϕ_3 . Now, solve the linear programs LP_{lower} and LP_{upper} for the assignment subproblem in Table 5.7.

The solution to LP_{lower} for this problem is $\pi_{AB} = 7$, $\pi_C = 1$, and of course $\pi_{\phi_3} = 0$. This leaves $s_1 = 4$, $s_2 = 1$, and $s_3 = 0$. The final upper and lower equilibrium payment vectors are given in Table 5.8.

Figure 5.1 shows the binding linear constraints for this example projected into the space $\pi_{AB} \times \pi_C$. The constraints are labeled with the agent whose valuations impose them (i.e. a_1 corresponds to agent 1). The white area is the space of equilibrium prices, and the dashed line between $\bar{\pi}^*$ and $\underline{\pi}^*$ are the k -bundle prices.

Notice that $\bar{\pi}_C^*$ is less than $v_1(C)$. In Table 5.6, agent 1 values AB at \$10, while agent 2 values it at \$8. However, in the efficient allocation, agent 2 receives AB. To make agent 1 satisfied with this outcome, we give it some surplus by reducing the price of its allocation. Ironically, this situation arises only when agents have subadditive preferences. Although complementarities motivate us to permit bundle bidding, it is subadditivity that provides a challenge when constructing equilibrium payments.

	A	B	C	AB	AC	BC	ABC
$\bar{\pi}^*$	4	4	3	8	6	6	11
$\underline{\pi}^*$	4	2	1	7	6	5	10

Table 5.8: Equilibrium payments.

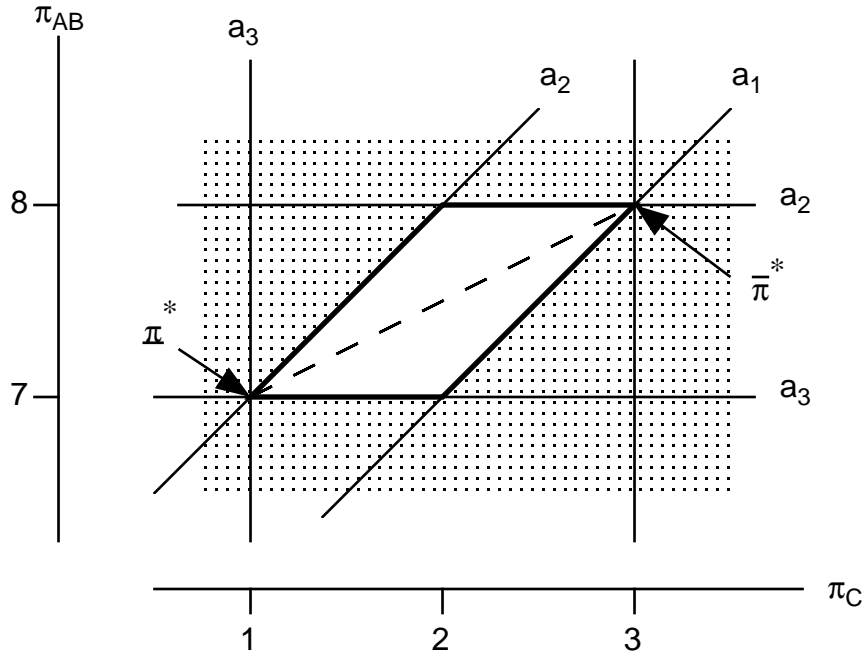


Figure 5.1: The space of possible equilibrium payments for the example in Table 5.6. The dashed line indicates the range of k -bundle prices.

5.4 The Family of k -Bundle Auctions

Section 5.3 presents the construction of equilibrium payments in the context of an omniscient mediator. The methodology can be used directly as a matching function.

Definition 5.1 *The matching function γ^{kB} assigns a payment to each element of the lattice \mathcal{B} according to the formula $k\bar{\pi}^* + (1 - k)\underline{\pi}^*$, where $k \in [0, 1]$ and $\underline{\pi}^*$ and $\bar{\pi}^*$ are computed by applying (5.1), LP_{lower} , LP_{upper} , and (5.4) to the agents' bids.*

We can combine the γ^{kB} with the full set of auction parameters described in Chapter 3 (restricted in that we have defined γ^{kB} for only the single-seller case). This gives potentially hundreds of new auction types to explore. The main topic of this section is the introduction of the Ascending k -Bundle Auction. However, to gain some insight into the incentives agents face under this pricing scheme, I first examine the sealed-bid variants.

5.4.1 Sealed-Bid k -Bundle Auctions

The sealed-bid k -bundle auction accepts bids that express a willingness-to-pay value for each bundle. At a prespecified time, the auction computes the payments and allocation using γ^{kB} .

	A	B	AB
Agent 1	0	0	3
Agent 2	2*	2	2
Agent 3	2	2*	2

Table 5.9: An example with multiple equilibrium bundle payment vectors.

First, consider the sealed-bid bundle auction where $k = 0$. Although LP_{lower} produces the GVA payments for the assignment subproblem (Leonard, 1983), it is not necessarily true that these payments are equivalent to the GVA payments for the original problem. Recall the example in Table 5.2. The GVA payments are \$3 for agent 1, and \$2 for agent 2. However, $\underline{\pi}^* = (0, 0, 3)$ for this problem.

Therefore, we would not expect the $k = 0$ bundle auction to be incentive compatible. The following example shows how the auction can be manipulated. Consider the valuations in Table 5.9. Assume that agents 2 and 3 reveal their true valuations. Can agent 1 be better off by reporting something other than its true valuation? The answer is yes. By reporting its true valuation, agent 1 is allocated nothing and receives a surplus of zero. Suppose, instead, that agent 1 reports $\xi_1^{-1}(AB) = 5$. The mechanism calculates the optimal allocation to be the one in which agent 1 receives AB, and a supporting payment lattice $\underline{\pi}^* = (2, 2, 2)$. Thus, by not reporting its true valuations, agent 1 receives a surplus of 1.⁵

Now we turn our attention to the sealed-bid bundle auction where $k = 1$. This pricing strategy would discourage manipulations like the one above, because agent 1 would be forced to pay 5. However, like the standard sealed-bid first-price auction, it suffers from potential efficiency losses due to strategic behavior. This can be seen in the example in Table 5.1 in which agent 1 has an incentive to bid $2 + \epsilon$. If the agent's information is imperfect, it risks underbidding and losing the good.

5.4.2 Ascending k -Bundle Auction (A1BA)

I now introduce a member of the Ascending k -Bundle Auction family, where $k = 1$ (abbreviated A1BA). A1BA accepts bids on bundles, and computes a tentative optimal allocation and a quote, in the form of the payment lattice, by using the k -bundle payment procedure. A1BA applies the ascending rule and the beat-the-quote rule. The combination of these rules requires that agent i 's new offer on bundle b satisfies $\hat{\xi}_i^{-1}(b) \geq \max(\pi_b, \xi_i^{-1}(b))$ for all

⁵In fact, this holds for any $\xi_1^{-1}(AB) \geq 4$.

b , and $\hat{\xi}_i^{-1}(b) \geq \max(\pi_b, \xi_i^{-1}(b)) + \delta$ for at least one b . The auction clears when a period of bidding inactivity transpires.

As a practical matter, rather than having an agent specify a valuation for every member of the lattice in each offer, the auction stores the agent’s previous bids (beginning at zero) and the agent’s message is treated as an update. I assume that the auctioneer interprets these updates in a manner consistent with the assumption of monotone valuations. If agent i increases its offer on b , the auction ensures that i ’s offer is raised on all $c \supset b$. Formally, for all $c \supset b$,

$$\hat{\xi}_i^{-1}(c) = \max(\xi_i^{-1}(c), \hat{\xi}_i^{-1}(b)).$$

For example, if the agent first bid $\xi_i^{-1}(ABC) = \$5$, and then bid $\xi_i^{-1}(AB) = \$10$, The auctioneer will consider the agent’s current bid on ABC to be $\xi_i^{-1}(ABC) = \$10$.

The auction begins with a quote in which each payment on the lattice is zero. Thereafter, it calculates a new payment lattice each time a bid is admitted. With $k = 1$, the quote is the payment lattice $\bar{\pi}^*$ computed from the current set of bids. Because of the manner in which the payments are determined, the agent knows it is winning the bundle that maximizes $\xi_i^{-1}(b) - \pi_b$. However, if more than one bundle satisfies this condition, the agent may not be able to tell from among these which one it is winning. To remedy this, the auction informs each agent which bundle, if any, it is tentatively winning.

The auction clears and closes after no new bids are received within a specified period (inactivity-based clear).

Because agents can bid on bundles in A1BA, they do not suffer from the exposure problem in which they are reluctant to express their true valuation for a bundle when they need to purchase the elements individually. However, there is still an incentive for an agent to free ride. It is unclear whether the threat of the auction closing while two agents play “chicken” is enough to avoid the free rider problem.

The auction design is directly applicable to situations where bids are restricted in order to admit polynomial time algorithms for computing \mathbf{f}^* (Rothkopf et al., 1998). Even without such restrictions, when bids are sparse, the computation of the efficient allocation can be cast as a straightforward search problem, and recent experimentation has found that problems with thousands of (superadditive) bundle bids can be solved in this manner (Fujishima et al., 1999; Sandholm, 1999a). The resulting allocation is locally efficient, and the $k = 1$ payments are separating with respect to the bids received. The

auction needs to announce prices on only those bundles that have received bids, since the agents can easily perform the inference to determine minimal payments for the rest of the bundles.

5.5 A1BA Simulations with Myopic Agents

A complete game-theoretic analysis of A1BA is difficult. To gain some insight into the auction’s potential performance, I analyzed a protocol in which agents followed a straightforward best-response bidding strategy.

5.5.1 The Myopic Strategy

Let \check{b}_i be agent i ’s tentative allocation, as asserted by the auction. A myopic agent behaves as if it can win any bundle that it is not currently winning by bidding $\pi_b + \delta$. The agent’s real surplus from purchasing its tentative allocation at the announced payment is $v_i(\check{b}) - \pi_{\check{b}}$. The myopic agent’s strategy is to bid on bundle, b' , that maximizes its real surplus at the given prices,

$$b' = \arg \max_b \begin{cases} v_i(\check{b}) - \pi_{\check{b}} & \text{if } b = \check{b}, \\ v_i(b) - (\pi_b + \delta) & \text{otherwise.} \end{cases} \quad (5.6)$$

Then, if the solution to (5.6) provides strictly more surplus than the agent’s tentative allocation, the agent will increase its offer on b' to $\pi_b + \delta$.⁶

5.5.2 Design of the Simulation

Agents were assigned valuations on all possible bundles according to the following algorithm, parametrized by ℓ and $\beta > 0$.

1. Assign values to individual items from the uniform distribution of integers between one and ℓ . That is, $v_i(j) \in [0, \ell]$.
2. Starting with bundles of size 2, and progressively increasing the bundle size,

⁶Note that b' is not necessarily unique. When \check{b} is one of the bundles that maximizes the agent’s real surplus, the bidding policy ensures that the agent won’t change its bid. However, in other case where more than one bundle maximizes (5.6), agents must still choose among the potential candidates. In the experimentation, such ties were broken in favor of bundles that come earlier in a particular ordering based on the bundles binary representations.

Solution Type	Percent of Experiments
{ABCDE}	1.5%
{ABCD} {E}	11.8%
{ABC} {DE}	6.5%
{ABC} {D} {E}	26.4%
{AB} {CD} {E}	16.3%
{AB} {C} {D} {E}	33.1%
{A} {B} {C} {D} {E}	4.4%

Table 5.10: Distribution of (normalized) solutions when $\beta = 1.5$.

Let $\underline{v}_b = \max_{c \subset b} v_i(c)$.

Let $\bar{v}_b = \max_{c \subset b} v_i(c) + v_i(b \setminus c)$.

Assign a value to $v_i(b)$ selected from a uniform distribution whose range is $[\underline{v}_b, \underline{v}_b + \beta(\bar{v}_b - \underline{v}_b)]$.

The parameter ℓ determines the range of valuations for individual items. The parameter β controls the potential supermodularity of agent preferences. If $\beta = 0$, then an agent's valuations are extremely subadditive—an agent's valuation for the bundle is its maximal valuation for any element of the bundle. When $\beta > 1$, an agent has the potential for superadditive valuations. When $\beta = 2$, any given valuation assignment has a 50% chance of being superadditive.

This method of constructing random valuation functions was designed to produce problems which were likely to have both subadditive and superadditive components. The simulation consisted of 1000 randomly generated problems, each with five agents and five resources, with $\ell = 10$ and $\beta = 1.5$. The distribution of optimal solutions among the 1000 random problems (with resources relabeled to normalize the solutions) is shown in Table 5.10.

5.5.3 Results of the Simulation

The A1BA was run for each of these 1000 random problems with agents using the myopic best-response bidding policy described above with $\delta = .5$. In 918 of the trials, the allocation reached by the protocol was optimal.⁷ The average efficiency of solutions found by the protocol was 99.8%. On average, the seller (who I assumed had zero reserve prices)

⁷In 222 of the trials with efficient outcomes, the auction reached a different allocation than the A* search algorithm I used to compute the optimal solution. This indicates that many of the problems had at least two efficient solutions.

	A	B	AB
Agent 1	5	3*	7
Agent 2	5*	2	5

Table 5.11: An example in which it is in agent 1’s interest to not bid on A.

captured 80% of the social welfare generated. The lowest revenue percentage was 44%.

Although these results are encouraging, the numbers themselves should be viewed with some skepticism. As has been pointed out in similar studies, efficiency percentages can be inflated by adding a large constant to everyone’s valuations—the ratio $\frac{1}{2}$ does not look as good as $\frac{1001}{1002}$. Also, the minimum bid increment, δ , affects both the convergence speed and the quality of the solution: large δ s converge faster, but can prevent an agent from placing a winning bid that triggers the efficient allocation.

In addition, the assumption that agents would follow the myopic best-response strategy in a realistic setting should be suspect. An examination of the trials in which the auction terminated at inefficient allocations reveals something about possible agent strategies. In several of these trials, an agent won a large bundle, say b , without ever bidding on an element, $c \subset b$, that it should receive as part of the optimal solution. Because the agent did not bid on c , the auction lacked the information necessary to find the efficient allocation.

Table 5.11 illustrates the type of problem in which this can occur. Suppose agent 1 bids on only the bundle AB. Eventually, it will win this bundle at $\pi_{AB} = \$5 \pm \delta$. Thus, it will get a surplus of $\$2 \pm \delta$. If, instead, it bids on all of the bundles, a likely outcome is that it will pay $\$2 \pm \delta$ and win B (as it should in the efficient allocation). However, agent 1’s surplus is less in the second scenario than in the first, and the agent therefore has an incentive to bid strategically.

Recent experimentation by Parkes (1999) has produced similar positive results for myopic agents using *iBundle*. It is an open question whether myopic behavior in ascending bundle auctions (such as *iBundle*) always terminates in equilibrium payments ($\pm\delta$) whether or not we use γ^{kB} to set the intermediate prices.

Parkes achieves full efficiency with myopic agents by introducing discriminatory bundle payments. The motivation he gives for switching from anonymous to discriminatory payments is in the context of jump bids, and results in exactly the situation that makes the calculation of equilibrium payments interesting in the first place—an agent has a higher bid for a bundle than the agent who is actually receiving it. In his example, agent 1 offers to buy bundle AB at 200 and bundle C at 150. Agent 2 offers to buy bundle

AB for 100. The efficient allocation is to give AB to agent 2 and C to agent 1. However, charging each agent its bid is not consistent with a payment equilibrium. The A1BA solution is to calculate a payment for C that is consistent with a payment equilibrium, namely $\pi_C = 50$. The solution proposed by Parkes is to announce a different π_C to each agent.

5.6 Conclusion

In this chapter I discuss the problems that occur when agents have complementary preferences for discrete goods. There is a great deal of interest in ascending auctions that allocate resources under these conditions. Ascending auctions are attractive because they reveal information that reduces uncertainty about the value of bundles, and focuses the bidders efforts on bundles that are likely to be part of the final allocation.

I established that, when nonlinear payments on bundles are allowed, the efficient solution can always be supported by a payment equilibrium. In fact, any convex combination of the upper and lower equilibrium payments supports the efficient allocation. Moreover, this can be done with anonymous payments, which simplifies the auction's computational task and facilitates price discovery.

The method used to construct a payment equilibrium can be used as a matching function, which enables us to combine it with the rules defined in Chapter 3. I studied one auction in particular, A1BA, which performs quite well in simulations with myopic best-response agents. In addition, A1BA is the only iterative bundle auction that guarantees even a weak form of equilibrium.

Chapter 6

Multidimensional Auctions for Continuous Resources

This chapter addresses the topic of matching functions in multidimensional auctions for continuous goods. I show how standard fixed-point algorithms can be applied to the subproblem of computing partial equilibrium prices for the resources in the scope of the auction. The particular challenge of adapting fixed-point algorithms to our context is that we violate a principal assumption, namely, that only relative prices matter. In our case, an agent’s “demand” for the resources in α_J is dependent upon the prices of the resources in J relative to the prices of resources outside the scope of the auction.

Section 6.1 presents the motivation for the original application of fixed-point algorithms to economic analysis and reviews the basic algorithm. Section 6.2 presents a straightforward adaptation of the fixed-point algorithm to the computation of partial equilibrium.

6.1 Fixed-Point Algorithms

Over a century ago, Léon Walras conjectured the existence of equilibrium prices that balance supply and demand. The existence of equilibrium, for a broad class of problems, was formally proved by Arrow and Debreu (1954). However, the convergence of the Walrasian tatonnement price adjustment process is guaranteed only under significantly stronger assumptions. In 1960, Scarf demonstrated that iterative price adjustment processes will fail to converge in a relatively simple class of problems if their initial price vector is anything other than the equilibrium price vector.

One sufficient condition for the convergence of the tatonnement process is that aggregate demand functions exhibit *gross substitutability*—when the price of one resource goes

up, the demand for other resources cannot go down. Another sufficient condition for tatonnement convergence is that utility functions be quasilinear and the underlying preferences be convex. These conditions are discussed in detail in standard texts (e.g. (Mas-Colell et al., 1995, Chapter 17)).

The classic tatonnement process is a hill climbing algorithm—the auctioneer announces a price vector, agents state their excess demand, and the auction adjusts the prices of the resources that is most over- or under-demanded. We would expect iterative price adjustment algorithms to have difficulty navigating such topological features are ridges, saddle points, and local optima.

Economists have investigated two basic classes of algorithms for the centralized computation of Walrasian equilibrium. One approach is to apply Newton-like methods (Smale, 1976; Whalley, 1973). While these algorithms tend to work well in practice, they are not guaranteed to converge to the equilibrium. The second approach involves the application of *fixed-point algorithms*. Although they are computationally more demanding than Newton methods, fixed-point algorithms are guaranteed to converge.

Work on the application of fixed-point algorithms to the computation of economic equilibria was initiated by Scarf (1967). The approach is based on the fixed-point theorems of Brouwer—which shows the existence of a fixed point for a continuous, point-to-point mapping of a function onto itself—and Kakutani—which shows the existence of fixed points in point-to-set mappings.

Fixed-point equilibrium analysis is typically conducted in the context of a closed economic model, and the calculation is based on complete information. We assume that prices induce a single-valued *demand correspondence*, that is, at each price vector \mathbf{p} , agent i 's demand set has just one element. The concept of an offer that we have used thus far is more general than the traditional notion of a Walrasian demand correspondence. For the purposes of discussing this algorithm, I assume that the demand correspondence serves as a bound on the agent's offer, and, abusing notation somewhat, let $w_i(\mathbf{p})$ represent this bound.

When a correspondence is single-valued, the bid can be separated into resource-specific functions:

$$w_i(\mathbf{p}) = \begin{bmatrix} w_{i,1}(\mathbf{p}) \\ \vdots \\ w_{i,n}(\mathbf{p}) \end{bmatrix}.$$

Each $w_{i,j}(\mathbf{p})$ is referred to as an *offer curve*.

Generally, demand functions are assumed to be homogeneous of degree zero, that is, they are invariant with respect to uniform, positive scalar transformations of prices. Thus, only relative prices matter. For convenience, we normalize prices to a unit simplex,

$$\sum_j p_j = 1.$$

Let the total excess demand for resource j at price vector \mathbf{p} be

$$d_j(\mathbf{p}) = \sum_i w_{i,j}(\mathbf{p}),$$

where, in this case, $w_i(\mathbf{p})$ is the agent's truthful offer curve.

Scarf uses a simple function to map the excess demand back into the unit simplex. The mapping

$$h_j(\mathbf{p}) = \frac{p_j + \max(0, d_j(\mathbf{p}))}{1 + \sum_k \max(0, d_k(\mathbf{p}))}.$$

ensures that at every point, \mathbf{p} , on the simplex, $h(\mathbf{p})$ is also on the simplex.

Scarf's algorithm triangulates the simplex and systematically examines the the triangulated regions to find the equilibrium. By refining the mesh on the simplex, finer approximations to equilibrium can be computed. Scarf's original algorithm was later enhanced by Merrill (1972), Van der Laan and Talman (1979), and others to permit dynamic refinement of the mesh. Scarf's 1973 monograph provides a very accessible discussion of the basic algorithm, although it naturally does not include more recent improvements. Shoven and Whalley (1995) provide an excellent overview of the theory, implementation, and application of fixed-point algorithms in the computation of economic equilibria.

6.2 The Partial-Equilibrium Fixed-Point Algorithm for Multidimensional Auctions

The standard fixed-point algorithm is applied in a mechanism structured $\mathcal{M} = \{\alpha_{\{1, \dots, n\}}\}$. The market structure presented in this thesis provides a framework within which we can question whether the fixed-point algorithm can be used in an auction with more limited scope. If so, then we can investigate whether we can guarantee convergence with less than complete aggregation of resources, and a corresponding reduction in the size of messages.

In particular, if such gains exist, I would expect they come from clustering resources that exhibit complementarity.

6.2.1 The Partial-Equilibrium Fixed-Point Procedure

Consider the auction for the set of continuous resources J . Assuming competitive behavior, an agent bidding in α_J will place a bid that assumes the prices of resources $\mathcal{J} \setminus J$ are constant.

Even when the agent assumes that the prices of resources $\mathcal{J} \setminus J$ are fixed, the agent's demand for those resources is a function of the prices of J . For every price vector \mathbf{p}_J , and fixed prices $\mathbf{p}_{\mathcal{J} \setminus J}$, there is unique demand vector. Let $\mu_{\mathcal{J} \setminus J}^i$ be the total expenditures for resources in $\mathcal{J} \setminus J$ that i demands at \mathbf{p}_J with fixed $\mathbf{p}_{\mathcal{J} \setminus J}$. That is,

$$\mu_{\mathcal{J} \setminus J}^i(\mathbf{p}_J) = \sum_{j \in \mathcal{J} \setminus J} p_j w_{i,j}(\mathbf{p}).$$

In other words, $\mu_{\mathcal{J} \setminus J}^i$ is the amount of income that i is planning to spend on resources outside of J .

Let us treat $\mu_{\mathcal{J} \setminus J}^i$ as a resource whose true price, relative to resources in $\mathcal{J} \setminus J$, is one. I now define the auction $\hat{\alpha}_J$ as the auction for J enhanced with the resource $\mu_{\mathcal{J} \setminus J}^i$. An agent's bid in $\hat{\alpha}_J$ must express its demand for the resources $j \in J$ and $\mu_{\mathcal{J} \setminus J}$, as a function of \mathbf{p}_J :

$$w_i^J(\mathbf{p}_J) = \begin{bmatrix} w_{i, \mu_{\mathcal{J} \setminus J}}(\mathbf{p}) \\ w_{i,1}(\mathbf{p}) \\ \vdots \\ w_{i,|J|}(\mathbf{p}) \end{bmatrix}.$$

If the agent's demand function is homogeneous of degree zero, then its demand function in $\hat{\alpha}_J$ is also homogeneous of degree zero. The argument is as follows. A scalar transformation of the prices in $\hat{\alpha}_J$ (which includes $p_{\mu_{\mathcal{J} \setminus J}}$) implicitly transforms the prices of resources outside the auction. For instance, suppose we multiply the prices of all resources by $\beta > 0$. Homogeneity of degree zero implies that the agent's demands will be the same. Thus, its expenditure at $\mathbf{p}_{\beta J}$ will be exactly $\beta \mu_{\mathcal{J} \setminus J}^i$.

Thus, given $\mathbf{p}_{\mathcal{J} \setminus J}$, the agents' demand functions are nonnegative, continuous, and homogeneous of degree zero in \mathbf{p}_J . This allows us to (temporarily) normalize the prices

such that

$$p_{\mu_{\mathcal{J}\setminus J}} + \sum_{j \in J} p_j = 1.$$

and apply the fixed-point algorithm to compute equilibrium prices.

The final step is to return the computed prices to the original scale. Recall that $\mu_{\mathcal{J}\setminus J}^i$ represents the expenditures that the agents plan to make on resources outside the auction. In fact, $\mu_{\mathcal{J}\setminus J}^i$ is the numeraire, and its “price” relative to all other resources in the mechanism is one. Thus, after using the fixed-point algorithm to compute the relative prices of the partial equilibrium, we restore \mathbf{p}_J to values consistent with the prices on other resources by dividing them by $p_{\mu_{\mathcal{J}\setminus J}^i}$.

Note that the equilibrium computed is a *partial equilibrium* of the entire mechanism, because it is computed assuming the prices in other auctions are fixed. The prices in $\hat{\alpha}_J$ will equal *general equilibrium prices* only if the prices of $\mathcal{J} \setminus J$ are also general equilibrium prices. For this reason, I refer to this approach as the *partial-equilibrium fixed-point* (PEFP) algorithm. In general, a mechanism using this matching function will iterate through successive equilibration of each market, possibly converging to the general equilibrium.

6.2.2 Relationship to Tatonnement and Centralized Computation

It is worth noting that the two extremes of the partial-equilibrium fixed-point algorithm are standard approaches to the computation of equilibrium. Tatonnement-like processes, like WALRAS (Cheng and Wellman, 1998), have a separate auction in each resource. That is, they have structure $\mathcal{M}^{\text{WALRAS}} \equiv \{\alpha_{\{1\}}, \dots, \alpha_{\{n\}}\}$. We could map the demand curves in one-dimensional auctions onto a simplex using the above trick.

The standard fixed-point algorithm has a market structure $\mathcal{M}^{\text{FP}} \equiv \{\alpha_{\{1, \dots, n\}}\}$. This also is a special case of the PEFP algorithm—the expenditures dimension collapses because there are no resources outside the scope of the auction, and therefore, no use for the numeraire.

The tatonnement version requires agents to send one-dimensional messages (quantity as a function of price), but iterates (potentially) many times. The traditional fixed-point mechanism, on the other hand, requires only one message from the agents, but that message contains n n -dimensional offer curves. The PEFP auction gives us a method to explore the middle ground between these two extremes, and to more explicitly make

tradeoffs between computational and informational complexity.

The first question we should investigate is whether convergence guarantees can be established for the market structure in which complementary resources are aggregated into a single auction cleared by a PEPF algorithm.

6.3 Conclusion

This chapter briefly introduced the Partial-Equilibrium Fixed-Point algorithm. PEPF provides a way to guarantee convergence to partial equilibrium prices in subsets of the market. This gives us a tool with which to explore various market structures in the domain of continuous resources, and explore tradeoffs between message size, distribution of computation, and convergence properties. Cheng and Wellman (1998), for example, provide some motivation for distributed computation of economic equilibria.

Moreover, this algorithm can be used in auctions whether or not agents bid truthfully, and combined with other auction rules. Naturally, the convergence and existence properties of the competitive protocol may not hold, but the mechanism itself remains well defined as long as we require that bids be expressed as continuous offer curves.

Thus, in addition to structural variations, we can explore auctions that combine different rules from Chapter 3.

Chapter 7

Summary and Future Work

7.1 Summary of Contributions

Since its establishment as a field over thirty years ago, mechanism design has developed a solid theoretical foundation. However, the field has, until recently, had little impact on practice—the majority of auctions studied in the literature have been around for hundreds of years. The most prominent exception is the development of the Simultaneous Ascending Auction used by the FCC to allocate spectrum licenses in the early part of this decade.

Electronic commerce presents an unprecedented opportunity for auction theorists to influence the evolution of mainstream economic institutions. The rate at which online auctions are being implemented is nothing short of astounding. The entrepreneurs behind this wave of market-building are asking the academy for a cookbook in which they can lookup their problem description and design goals, and find the appropriate auction type.

Unfortunately, no such cookbook exists. Many issues that arise in practice, and which influence the auction design, have (appropriately) been assumed away in theoretical analysis. Moreover, for many problem domains, satisfactory allocation mechanisms have yet to be identified. The solution that I advocate is to build flexible auction infrastructure that can be quickly configured for different applications and easily extended as new auction rules are developed.

To support the development of configurable auction servers, this thesis defines a very general semantic description of the ingredients from which auctions can be built. This effectively decomposes auctions into orthogonal rules, which can be easily implemented in an object-oriented auction server. Indeed, our development of the Michigan Internet AuctionBot benefited greatly from an earlier version of the auction parametrization.

I define these rules for multidimensional auctions that accept bids in the form of

correspondences between net allocations and monetary transfers. This parametrization is far more general and inclusive than previous classification schemes, and captures the vast majority of auctions described in the literature and used in practice. Most of the rules are fully orthogonal, which allows them to be combined in novel ways. This converts the auction designer’s decision from a question of “which auction type should I use?” to the question “which rules should I enable?”

In addition, the parametrization provides a convenient way to describe auctions to electronic agents. It is the first step in the development of an auction description language that will facilitate the deployment of software agents capable of negotiating across auction sites that may have different rules.

Finally, the parametrization is of theoretical interest because it exposes the underlying structure of the space of auction designs, and highlights areas in need of further research.

One such area is the allocation of heterogeneous resources when agents have complementary preferences. This setting is particularly challenging for mechanism designers. In the space of market-based solutions, the difficulty is evidenced by the fact that price equilibria are not guaranteed to exist. There is great interest in developing an ascending auction in this space because the iterative revelation of information helps guide agents toward the efficient allocation with (hopefully) less revelation of private information than direct revelation mechanisms. Moreover, it is the general consensus of researchers in this area that such auctions must allow bids on bundles. In fact, the FCC is considering whether to permit bundle bidding in future spectrum license auctions.

This thesis proposes an innovative auction for this problem domain. The auction, called A1BA, is an ascending, single-sided auction that permits bids on combinations of objects. It generates price quotes that associate payments with bundles. Unlike many competing combinatorial auctions, A1BA’s quotes are separating: an agent can exactly determine its tentative allocation. Moreover, these payments are anonymous—every agent receives the same information—which facilitates price discovery and simplifies the auctioneer’s task.

Most importantly, A1BA is guaranteed to result in payments that support a payment equilibrium—a slightly weaker version of the standard price equilibrium. In a payment equilibrium, no agent prefers some other agent’s allocation at the posted payments. This is a feature unique to A1BA.

Like other combinatorial auctions, A1BA sacrifices efficiency and incentive compatibility for reduced message size. The auction is well defined even when bids are sparse, or

when the bundles that can be bid on are restricted to admit polynomial time computation of the locally efficient allocation.

A1BA achieved very high efficiency in simulations with myopic agents. The simulation is perhaps most useful in highlighting strategic behaviors that can improve an agent's utility. In realistic settings, we would expect agents to attempt to play such strategies.

In addition to the two main contributions, this dissertation presents two computational algorithms that can be used in computerized auctions. The first is the 4-HEAP algorithm which can be used in one-dimensional, discrete resource auctions with divisible bids. The second is the PEFP algorithm, an adaptation of standard fixed-point algorithms for use in multidimensional auctions for continuous resources. These two algorithms are part of a recent trend we might call *computational auction theory*: the design of computational algorithms for automated auctions.

7.2 Future Work

Delineating the design space is just the first step in serving the needs of computer scientists applying market-based techniques or building electronic commerce infrastructure. To effectively implement market-oriented solutions to distributed resource allocation problems, we need to understand the properties of possible market designs. Many questions developed in this thesis remain to be answered.

For example, the primary motivation for the structural organization of markets is to investigate whether we can improve convergence to, or existence of, equilibrium at intermediate levels of resource clustering. I expect that, at least in some cases, partial aggregation will strike the right balance among desirable economic properties and computational effort. This dissertation provides the machinery to address that question in future research.

Although the parametrization of auction rules captures a large number of auction types, it could be broader still. There are a number of auctions that cannot be fully described by my parameter set. More importantly, the presentation is a semantic description of the auction parameters. For it to be widely used, we need to take the next step and develop a convenient syntax, perhaps in a standard interchange language.

The PEFP algorithm is a promising tool to support market decomposition. However, a more rigorous analysis of its properties needs to be conducted, and the algorithm needs to be implemented and compared with other candidates. Indeed, the research commu-

nity has just begun to investigate computational algorithms in the context of auction implementation.

The investigation of the A1BA auction suggests that it performs well in the context of the allocation of heterogenous, discrete resources. This analysis is preliminary, and much more work needs to be done to understand the A1BA's incentives, and to compare it with other leading combinatorial auctions. These auctions are complex, and the research is currently directed toward experimentation with human subjects or simulations with software agents. Eventually, we would like either an analytic solution to the game, or to know that no solution exists.

Finally, the majority of research in combinatorial mechanisms is focused on one-sided auctions. It is very likely that two-sided combinatorial mechanisms will be required in some e-commerce or multiagent systems applications. As the one-sided mechanisms mature, the attention of researchers will shift to the double-sided combinatorial auctions.

Appendices

Appendix A

URLs for Online Auction Sites

Amazon.com	http://www.amazon.com
Amazon's "Shop the Web"	http://shoptheweb.amazon.com
BargainFinder	http://bf.cstar.ac.com
eBay	http://www.ebay.com
Egghead.com	http://www.egghead.com
FastParts.com	http://www.fastparts.com
Jango	http://www.jango.com
Michigan Internet AuctionBot	http://auction.eecs.umich.edu
Onsale	http://www.onsale.com
Priceline	http://www.priceline.com
Ubid	http://www.ubid.com

Appendix B

Parametrized Descriptions of Familiar Auctions

This appendix demonstrates how many auction types can be described by the parametrization. In cases where several common variations exist, such as the CDA, I describe just one of them.

	eBay Standard	eBay “Dutch”	UBid “Traditional”
Bidding rules			
Participation	{many:1}	{many:1}	{many:1}
Bid Semantics	single-unit point	single-price point	single-price point
Bid Refinements	NA	divisible	indivisible
Seller bid dominance	none	none	none
Buyer bid dominance	none	none	none
Beat-the-quote	buyer	buyer	buyer
Eligibility	NA	NA	NA
Information revelation			
Price quotes	ask	ask	nonlinear
Quote schedule	activity	activity	activity
Order book	winners	winners	winners
Transaction history	NA	NA	NA
Clearing policy			
Clearing schedule	inactivity	inactivity	inactivity
Closing conditions	inactivity	inactivity	inactivity
Matching function	$\gamma^{k=1}$	γ^{BP}	γ^{greedy}
Tie breaking	earliest	quantity, earliest	quantity, earliest
Transaction fees	entrance, nonlinear	entrance, nonlinear	entrance, nonlinear

Table B.1: Parameter choices that describe three online auctions.

	English Outcry	Vickrey	CDA
Bidding rules			
Participation	{many:1}	{many:1}	{many:many}
Bid Semantics	single-unit point	single-unit point	single-price point
Bid Refinements	NA	NA	indivisible
Seller bid dominance	none	none	none
Buyer bid dominance	none	none	none
Beat-the-quote	buyer	NA	NA
Eligibility	NA	NA	NA
Information revelation			
Price quotes	ask	none	bid-ask
Quote schedule	activity	none	activity
Order book	open	closed	closed
Transaction history	NA	NA	prices
Clearing policy			
Clearing schedule	inactivity	fixed time	activity
Closing conditions	inactivity	fixed time	never
Matching function	$\gamma^{k=1}$	$\gamma^{k=0}$	γ^{ET}
Tie breaking	earliest	arbitrary	earliest
Transaction fees	none	none	none

Table B.2: Parameter choices that describe three classic auctions.

	WALRAS	SAA	A1BA
Bidding rules			
Participation	{many:many}	{many:1}	{many:1}
Bid Semantics	continuous bids	single-unit point	combinatorial
Bid Refinements	NA	NA	indivisible
Seller bid dominance	none	none	none
Buyer bid dominance	none	none	none
Beat-the-quote	NA	buyer	buyer
Eligibility	NA	buyer	NA
Information revelation			
Price quotes	p^*	ask prices	ask payments
Quote schedule	activity	periodic	activity
Order book	closed	open/closed	closed
Transaction history	NA	NA	NA
Clearing policy			
Clearing schedule	Equilibrium Signal	fixed time	inactivity
Closing conditions	Equilibrium Signal	fixed time	inactivity
Matching function	γ^{eq}	γ^{BP}	$\gamma^{kB}, k = 1$
Tie breaking	arbitrary	earliest	earliest
Transaction fees	none	none	none

Table B.3: Parameter choices that describe some other interesting auctions.

Appendix C

Summary of Notation

b A bundle of resource, $b \equiv \{j \mid b(j) = 1\}$.

\mathcal{B} The set of all possible bs . $\mathcal{B} \equiv \mathcal{P}(\mathcal{J})$. \mathcal{B} is a lattice.

\mathcal{B}_Θ The set of bundles that are allocated.

\mathcal{B}_\emptyset The set of bundles that are unallocated.

b_j Indicator function that determines if j is in b .

c Secondary label for bundles.

d_j Aggregate demand for j .

$e_{i,j}$ Agent i 's endowment of j , a number

e_i Agent i 's endowment vector.

e The vector of endowment vectors for all agents.

$f_{i,j}$ The amount of resource j that i gets in f .

f_i The allocation to agent i in the solution f .

f A solution. f specifies an allocation for every agent, however f is constrained to be feasible.

f^I A solution to the subproblem of reallocation of the endowments of the agents in I .

F The space of all *feasible* solutions.

g An object (item or bundle) that is assigned.

g_i Agent i 's assignment.

G The set of object used in the assignment problem.

γ A matching function.

h An agent. Used as a secondary label.

i An agent.

\mathcal{I} The set of agents.

I A subset of the agents, $I \subseteq \mathcal{I}$.

\mathcal{I}_Θ The agents who are allocated nonempty sets.

$\mathcal{I}_{-\Theta}$ The agents who are allocated empty sets.

j A resource type.

\mathcal{J} The set of resource types.

J A subset of resource types, $J \subseteq \mathcal{J}$.

$j = 1 \dots n$ the labels of resources.

k A secondary label for a resource type.

m The number of agents in \mathcal{I} .

\mathcal{M} An allocation mechanism.

n The number of resource types in \mathcal{J} .

p_j The price of good j , a number

\mathbf{p} The price vector $[p_1, \dots, p_n]$.

\mathbf{p}^* An equilibrium price vector.

q A signal (message) put out by the mechanism.

$q(t)$ The signal put out by the mechanism at time t .

q^i The (discriminatory) signal sent to agent i .

ρ is the function that an auction uses to generate q from signals. $q = \rho(\mathbf{w})$.

s_i The surplus achieved by agent i .

s The total surplus achieved.

u_i an agent's utility function

$u_i(\mathbf{x}_i, \mu) = v_i(\mathbf{x}_i) + \mu$ a quasilinear utility function.

$v_i(\mathbf{x}_i)$ Agent i 's valuation, under quasilinear preferences, for \mathbf{x}_i .

$V(\mathbf{f})$ The total social value of \mathbf{f} under quasilinear preferences.

w_i A bid message from agent i in price space.

$w_i^{-1}(j)$ An offer from i for resource j in price space.

\mathbf{w} The matrix of all offers (in price space).

W The space of possible offer messages.

$x_{i,j}$ The amount of j allocated to i .

\mathbf{x}_i The vector of resources allocated to i .

$\mathbf{x} = [\mathbf{x}_1, \dots, \mathbf{x}_m]$ a vector of agent allocations.

X the domain of resource vectors, $\mathbf{x} \in X$.

$z_{i,j}$ The net change in i 's allocation of j .

\mathbf{z}_i Agent i 's net allocation vector.

\mathbf{z} The vector of net allocations, one for each agent.

α an auction, the fundamental mediator in a market.

α_J an auction that mediates the allocation of resources in the subset J . J is called the *scope* of α .

μ Money.

μ_J The expenditures to be reserved for purchasing resources in J .

π_b The payment for bundle b , a number.

π^i The payment made by i .
 π The payments for all bundles, a lattice.
 $\pi_\Theta, \pi_\emptyset$ Payments on allocated/unallocated bundles.
 $\sigma(\alpha)$ denotes the scope of α . $J = \sigma(\alpha_J)$.
 θ An exchange, which moves resources (including money) between two agents.
 Θ The set of exchanges calculated at a clear.
 $\chi_i(q, w_i)$ The tentative allocation implied by the mechanism's signal.
 ψ^i Agent i 's strategy.
 ψ The vector of agent strategies.
 Ψ The space of possible agent strategies.
 ξ_i An offer from i expressed in payment space.
 $\xi_i^{-1}(j)$ An offer from i for resource j , expressed in payment space.
 Ξ The space of possible offers expressed as payments.
 ϕ_i an empty bundle labeled for agent i .
 $\Phi \equiv \{ \phi_i \mid i \in \mathcal{I}_{-\Theta} \}$.

Bibliography

- Anderson, A., Birgean, I., and MacKie-Mason, J. K. (1999). Bilateral negotiation with fees. In *IBM/IAC Workshop on Internet-Based Negotiation Technologies*.
- Arrow, K. J. and Debreu, G. (1954). Existence of an equilibrium for a competitive economy. *Econometrica*, 22:265–290.
- Banks, J. S., Ledyard, J. O., and Porter, D. P. (1989). Allocating uncertain and unresponsive resources: An experimental approach. *Rand Journal of Economics*, 20:1–22.
- Bikhchandani, S. and Mamer, J. W. (1997). Competitive equilibrium in an economy with indivisibilities. *Journal of Economic Theory*, 74:385–413.
- Bikhchandani, S. and Ostroy, J. M. (1998). The package assignment model. Technical report, University of California at Los Angeles.
- Campbell, D. E. (1987). *Resource Allocation Mechanisms*. Cambridge University Press, London.
- Chavez, A. and Maes, P. (1996). Kasbah: An agent marketplace for buying and selling goods. In *First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology*, London.
- Cheng, J. Q. and Wellman, M. P. (1998). The WALRAS algorithm: A convergent distributed implementation of general equilibrium outcomes. *Computational Economics*, 12:1–24.
- Clarke, E. H. (1971). Multipart pricing of public goods. *Public Choice*, 11:17–33.
- Clearwater, S. (1995). *Market-Based Control: A Paradigm for Distributed Resource Allocation*. World Scientific.

- Cormen, T. H., Leiserson, C. E., and Rivest, R. L. (1990). *Introduction to Algorithms*. MIT Press and McGraw Hill.
- CRA (1998). Report 2: Simultaneous ascending auctions with package bidding. Technical Report CRA No. 1351-00, Charles River Associates Incorporated and Market Design, Inc.
- Cramton, P. (1998). Ascending auctions. *European Economic Review*, 42(3-5):745–756.
- Culver, D. (1999). Spectrum licenses: After the gold rush. *Interactive Week Online*.
- Dalton, G. (1999). Bazaar advantages. *InformationWeek*, pages 42–48.
- Davis, R. and Smith, R. G. (1983). Negotiation as a metaphor for distributed problem solving. *Artificial Intelligence*, 20:63–109.
- Demange, G., Gale, D., and Sotomayor, M. (1986). Multi-item auctions. *Journal of Political Economy*, 94:863–72.
- DeMartini, C., Kwasnica, A. M., Ledyard, J. O., and Porter, D. (1998). A new and improved design for multi-object iterative auctions. Technical report, California Institute of Technology.
- Doorenbos, R. B., Etzioni, O., and Weld, D. S. (1997). A scalable comparison-shopping agent for the world-wide web. In *First International Conference on Autonomous Agents*, pages 39–48.
- Engelbrecht-Wiggans, R. (1980). Auctions and bidding models: A survey. *Management Science*, 26:119–142.
- Friedman, D. (1993). The double auction market institution: A survey. In (Friedman and Rust, 1993), pages 3–26.
- Friedman, D. and Rust, J., editors (1993). *The Double Auction Market: Institutions, Theories, and Evidence*. Addison-Wesley Publishing, Reading, MA.
- Fudenberg, D. and Tirole, J. (1996). *Game Theory*. MIT Press.
- Fujishima, Y., Leyton-Brown, K., and Shoham, Y. (1999). Taming the computational complexity of combinatorial auctions: Optimal and approximate approaches. In *Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*.

- Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco.
- Gibbard, A. (1973). Manipulation of voting schemes: A general result. *Econometrica*, 41:587–601.
- Groves, T. (1973). Incentives in teams. *Econometrica*, 41:617–631.
- Gul, F. and Stacchetti, E. (forthcoming). Walrasian equilibrium without complementarities.
- Guttman, R. H., Moukas, A. G., and Maes, P. (1998). Agent-mediated electronic commerce: A survey. *Knowledge Engineering Review*, 13(2):147–159.
- Hu, J. and Wellman, M. P. (1998). Online learning about other agents in a dynamic multi-agent system. In *Second International Conference on Autonomous Agents (Agents 98)*, pages 239–246, Minneapolis.
- Jordan, J. S. (1982). The competitive allocation process is informationally efficient uniquely. *Journal of Economic Theory*, 28:1–18.
- Kelso, A. S. and Crawford, V. P. (1982). Job matching, coalition formation, and gross substitutes. *Econometrica*, 50:1483–1504.
- Koopmans, T. and Beckmann, M. (1957). Assignment problems and the location of economic activities. *Econometrica*, 25:53–76.
- Kumar, M. and Feldman, S. I. (1998). Internet auctions. In *3rd USENIX Workshop on Electronic Commerce*, pages 49–60.
- Ledyard, J. O., Porter, D., and Rangel, A. (1997). Experiments testing multiobject allocation mechanisms. *Journal of Economics and Management Strategy*, 6(3):639–675.
- Leonard, H. B. (1983). Elicitation of honest preferences for the assignment of individuals to positions. *Journal of Political Economy*, 91(3):461–479.
- MacKie-Mason, J. K. and Varian, H. R. (1994). Generalized Vickrey Auctions. Technical report, University of Michigan.
- Mas-Colell, A., Whinston, M. D., and Green, J. R. (1995). *Microeconomic Theory*. Oxford University Press, Oxford.

- McAfee, R. P. (1992). A dominant strategy double auction. *Journal of Economic Theory*, 56:434–450.
- McAfee, R. P. and McMillan, J. (1987). Auctions and bidding. *Journal of Economic Literature*, 25:699–738.
- McAfee, R. P. and McMillan, J. (1996). Analyzing the airwaves auction. *Journal of Economic Perspectives*, 10(1):159–175.
- McCabe, K. A. and Smith, V. L. (1993). Designing a uniform-price double auction: An experimental evaluation. In (Friedman and Rust, 1993), pages 307–332.
- McMillan, J. (1994). Selling spectrum rights. *Journal of Economic Perspectives*, 8(3):145–162.
- Merrill, O. H. (1972). *Applications and Extensions of an Algorithm that Computes Fixed Points of Certain Upper Semi-continuous Point to Set Mappings*. PhD thesis, IOE, University of Michigan.
- Milgrom, P. (1987). Auction theory. In Dewley, T. F., editor, *Advances in Economic Theory: Fifth World Congress*. Cambridge University Press.
- Milgrom, P. (1989). Auctions and bidding: A primer. *Journal of Economic Perspectives*, 3(3):3–22.
- Milgrom, P. R. and Weber, R. J. (1982). A theory of auctions and competitive bidding. *Econometrica*, 50:1089–1122.
- Mount, K. and Reiter, S. (1974). The informational size of message spaces. *Journal of Economic Theory*, 8:161–191.
- Mullen, T. and Wellman, M. P. (1995). A simple computational market for network information services. In *First International Conference on Multiagent Systems*, pages 283–289, San Francisco, CA.
- Mullen, T. and Wellman, M. P. (1996). Market-based negotiation for digital library services. In *Second USENIX Workshop on Electronic Commerce*, pages 259–269, Oakland, CA.
- Myerson, R. B. and Satterthwaite, M. A. (1983). Efficient mechanisms for bilateral trade. *Journal of Economic Theory*, 29:265–281.

- Nash, J. (1950). Two-person cooperative games. *Proceedings of the National Academy of Sciences*, 21:128–140.
- Parkes, D. C. (1999). *iBundle*: An efficient ascending price bundle auction. Technical report, University of Pennsylvania.
- Parkes, D. C., Ungar, L. H., and Foster, D. P. (1998). Accounting for cognitive costs in on-line auction design. Submitted for publication.
- Rassenti, S. J., Smith, V. L., and Bulfin, R. L. (1982). A combinatorial auction mechanism for airport time slot allocation. *Bell Journal of Economics*, 13:402–417.
- Rodríguez, J., Noriega, P., Sierra, C., and Padget, J. (1997). FM96.5 a java-based electronic auction house. In *Second International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM '97)*.
- Rodríguez-Aguilar, J. A., Martín, F. J., Noriega, P., Garcia, P., and Sierra, C. (1998). Competitive scenarios for heterogeneous trading agents. In *Second International Conference on Autonomous Agents*, pages 293–300.
- Rosenschein, J. S. and Zlotkin, G. (1994). *Rules of Encounter*. The MIT Press, Cambridge.
- Rothkopf, M. H., Pekeč, A., and Harstad, R. M. (1998). Computationally manageable combinatorial auctions. *Management Science*, 44(8):1131–1147.
- Rust, J., Miller, J. H., and Palmer, R. (1994). Characterizing effective trading strategies: Insights from a computerized double auction tournament. *Journal of Economic Dynamics and Control*, 18:61–96.
- Sakurai, Y., Yokoo, M., and Matsubara, S. (1999). A limitation of the generalized vickrey auction in electronic commerce: Robustness against false-name bids. In *Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, pages 86–92.
- Sandholm, T. (1993). An implementation of the contract net protocol based on marginal-cost calculations. In *Proceedings of 11th National Conference on Artificial Intelligence (AAAI-93)*, pages 256–262.
- Sandholm, T. (1999a). An algorithm for optimal winner determination in combinatorial auctions. In *Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*.

- Sandholm, T. (1999b). mediator: a next generation electronic commerce server. In *AAAI-99 Workshop on Artificial Intelligence for Electronic Commerce*.
- Satterthwaite, M. A. (1975). Strategy-proofness and Arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions. *Journal of Economic Theory*, 10:187–217.
- Satterthwaite, M. A. and Williams, S. R. (1989). Bilateral trade with the sealed bid k -double auction: Existence and efficiency. *Journal of Economic Theory*, 48:107–133.
- Satterthwaite, M. A. and Williams, S. R. (1993). The Bayesian theory of the k -double auction. In (Friedman and Rust, 1993), pages 99–124.
- Scarf, H. E. (1960). Some examples of global instability of the competitive equilibrium. *International Economic Review*, 1(3):157–172.
- Scarf, H. E. (1967). The approximation of fixed points of a continuous mapping. *SIAM Journal of Applied Mathematics*, 15:1328–1343.
- Scarf, H. E. and Hansen, T. (1973). *The Computation of Economic Equilibria*. Yale University Press.
- Shapley, L. and Shubik, M. (1972). The assignment game I: The core. *International Journal of Game Theory*, 1:111–130.
- Shoven, J. B. and Whalley, J. (1995). *Applying General Equilibrium*. Cambridge University Press, London.
- Smale, S. (1976). A convergent process of price adjustment and global newton methods. *Journal of Mathematical Economics*, 3:107–120.
- Tesler, L. G. (1994). The usefulness of core theory in economics. *Journal of Economic Perspectives*, 8(2):151–164.
- Topkis, D. M. (1998). *Supermodularity and Complementarity*. Princeton University Press, Princeton, NJ.
- van der Laan, G. and Talman, A. J. J. (1979). A restart algorithm for computing fixed points without an extra dimension. *Mathematical Programming*, 17:74–84.
- Vickrey, W. (1961). Counterspeculation, auctions, and competitive sealed tenders. *Journal of Finance*, 16:8–37.

- Walsh, W. E. and Wellman, M. P. (1998). A market protocol for distributed task allocation. In *Third International Conference on Multiagent Systems*, pages 325–332, Paris.
- Walsh, W. E., Wellman, M. P., Wurman, P. R., and MacKie-Mason, J. K. (1998). Some economics of market-based distributed scheduling. In *18th International Conference on Distributed Computing Systems*, pages 612–621, Amsterdam.
- Wellman, M. P. (1993). A market-oriented programming environment and its application to distributed multicommodity flow problems. *Journal of Artificial Intelligence Research*, 1:1–23.
- Wellman, M. P. and Walsh, W. E. (1999). Distributed quiescence detection in multiagent negotiation. In *AAAI-99 Workshop on Negotiation: Settling Conflicts and Identifying Opportunities*.
- Wellman, M. P., Walsh, W. E., Wurman, P. R., and MacKie-Mason, J. K. (1999). Auction protocols for decentralized scheduling. Submitted for publication.
- Wellman, M. P. and Wurman, P. R. (1998). Real time issues for Internet auctions. In *IEEE Workshop on Dependable and Real-Time E-Commerce Systems*, Denver, CO.
- Whalley, J. (1973). A numerical assessment of the April 1973 United Kingdom tax reform. *Econometrica*, 42:139–161.
- Wurman, P. R., Walsh, W. E., and Wellman, M. P. (1998a). Flexible double auctions for electronic commerce: Theory and implementation. *Decision Support Systems*, 24:17–27.
- Wurman, P. R., Wellman, M. P., and Walsh, W. E. (1998b). The Michigan Internet AuctionBot: A configurable auction server for human and software agents. In *Second International Conference on Autonomous Agents*, pages 301–308, Minneapolis.
- Ygge, F. (1998). *Market-Oriented Programming and its Application to Power Load Management*. PhD thesis, Lund University.