

Chapter 10

Conclusions

Auctions offer great promise as mechanisms for optimal resource allocation in complex distributed systems with self-interested agents. However, limited and costly computation necessitates a rethinking of traditional auction theory because direct extensions of auctions that work well in small problems can fail in complex distributed systems. My thesis is that it is necessary to take an explicitly computational approach to auction design. Indeed, the value of auctions in e-commerce systems will depend on the ability to maintain the desirable properties of auctions, for example economic efficiency, robustness, and simplicity, as methods are introduced to allow tractable computation. Once computational issues are successfully addressed, auctions may provide simple, stable, and robust solutions to many important distributed optimization problems.

Agents often demand bundles of items, and have values for bundles that are not equal to the sum of the values of the items in the bundle; e.g., for task allocation, procurement problems, and dynamic bandwidth allocation. Combinatorial auctions allow agents to represent these preferences explicitly by submitting bids on bundles of items, for example stating “I only want A if I also get B”. Desirable properties of combinatorial auctions include strategy-proofness, such that truthful bidding is optimal whatever the strategies of other agents, and allocative-efficiency, such that the auction implements the value maximizing allocation. However there is a tension between the classic game-theoretic solution to this problem and computational efficiency.

The Groves family of mechanisms, and the Generalized Vickrey Auction (GVA) in particular, are strategy-proof and efficient. But the GVA is a sealed-bid combinatorial auction. Every agent must first report its value for every possible bundle, before the auctioneer

computes the allocation and agent payments. Sealed-bid auctions are undesirable computationally because of this complete revelation requirement, which soon becomes intractable in large complex domains for agents with hard valuation problems. Iterative mechanisms are absolutely critical in such domains because they can solve problems without complete information revelation from agents, and even without individual agents computing their exact values for all outcomes. The revelation principle of mechanism design, coupled with the uniqueness of Groves mechanisms provides useful guidance. Any strategy-proof and efficient iterative combinatorial auction must compute Groves payments at the end of the auction.

My dissertation develops an efficient iterative combinatorial auction, which avoids the centralization and complete revelation problems of the GVA. The auction, *iBundle*, and its extension *Extend&Adjust*, is an ascending-price combinatorial auction that allows agents to adjust their bids in response to bids from other agents. *iBundle* solves realistic problems without complete information revelation from agents, and with *Extend&Adjust* inherits much of the strategy-proofness of Groves mechanisms by computing Vickrey-Groves payments at the end of the auction. Instead of terminating as soon as the efficient allocation has been determined, *Extend&Adjust* remains open just long enough to collect additional information from agents (via best-response bids) to compute Vickrey payments.

In summary, the main contributions to computational mechanism design are:

- *iBundle Extend&Adjust*, an iterative combinatorial auction that computes minimal competitive equilibrium prices in the combinatorial allocation problem, with myopic best-response agent strategies.
- *iBundle Extend&Adjust*, an iterative combinatorial auction that computes the efficient allocation and Vickrey payments in all problems in which Vickrey payments can be priced in (non-linear and perhaps non-anonymous) competitive equilibrium, with myopic best-response agent strategies.

Another significant contribution is:

- *VICKAUCTION*, a primal-dual algorithm that computes Vickrey payments with myopic best-response information from agents.

iBundle Extend&Adjust interprets the primal-dual algorithm *VICKAUCTION* as an

ascending-price auction. The difference introduced in *i*Bundle Extend&Adjust is an explicit method to adjust prices in the second “extend” phase of the auction, which aims to be more “natural” to participants than the method in VICKAUCTION. An agent’s bids in the rounds that follow the normal termination point of *i*Bundle affect neither the final allocation nor the final price that the agent pays. The only effect is to reduce the price that other agents pay. Therefore it is important that an agent should not be able to detect that the auction is in this phase, or else it would simply drop out and wait for the auction to finally terminate.

The outstanding challenge is to complete a proof of the following conjecture.

- (conjecture) *i*Bundle Extend&Adjust is the first iterative combinatorial auction to compute the efficient allocation and Vickrey payments in all combinatorial allocation problems.

We provably compute Vickrey payments and the efficient allocation whenever the extended auction terminates, but still need a proof that the current dummy agent rules are sufficient for termination. There is very strong experimental evidence that this is indeed the case.

The combined system, *i*Bundle Extend&Adjust, with a proxy bidding agent interface, is a promising dynamic mechanism to solve combinatorial allocation problems with distributed self-interested agents. The proxy bidding agents constrain agents to follow myopic best-response strategies, boosting the robustness-to-manipulation that is achieved through terminating with Vickrey payments.

10.1 A Brief Review

Chapter 1 provides an introduction to the computational problems inherent in classic game-theoretic mechanisms, such as the Groves mechanisms. The Groves mechanisms are very centralized solutions to decentralized optimization problems. The mechanism makes truth-revelation a dominant strategy for agents, but requires them to provide complete information about their preferences, to the auctioneer which then computes an optimal system-wide solution.

An important challenge in computational mechanism design is to develop a strategy-proof and dynamic mechanism, in which it is an agent’s dominant strategy to provide

truthful information incrementally to the mechanism, until just enough is known about agents' problems to solve the system-wide problem. The scheme proposed in this dissertation, comprising of *iBundle Extend&Adjust*, makes significant progress in this direction for the combinatorial allocation problem. Transformation techniques between the exclusive-or bidding language provided in *iBundle* and other more expressive bidding languages also promise to lead to domain-specific implementations that can exploit structure in agent preferences without incurring the potentially exponential cost of a translation into explicit values on bundles.

Chapter 2 introduces important results from classic mechanism design, including the Groves family of mechanisms. Groves mechanisms are the *only* strategy-proof and efficient mechanisms among direct-revelation mechanisms. Taken along with the *revelation principle*, which states that any mechanism can be implemented as an equivalent direct-revelation mechanism (with agents reporting complete information about their preferences), this uniqueness provides useful focus to the design of iterative mechanisms. In particular, any efficient and strategy-proof iterative mechanism must compute the outcome of a Groves mechanism for the underlying preferences of agents.

The revelation principle does not mean that we need to only consider direct-revelation mechanisms when constructing mechanisms for combinatorial problems. The revelation principle assumes unlimited computational resources, both for agents in submitting valuation functions, and for the auctioneer, in converting a mechanism into a single-shot direct-revelation solution. In fact, the work in my dissertation clearly demonstrates that iterative auctions *expand* the implementation space when computation is an issue.

The tensions between classic game-theoretic solutions and tractable computational solutions soon become evident as one considers the application of mechanisms to difficult distributed optimization problems, such as supply-chain procurement or bandwidth allocation. Chapter 3 considered the computational demands within a mechanism, addressing computation at both the infrastructure (e.g. auctioneer) and agent level.

The first important cost is the *computational complexity* on the mechanism infrastructure. I surveyed a number of methods to address this cost, including approximations, special-cases and decentralized methods. Naive attempts to introduce approximations into mechanisms can break useful game-theoretic properties, such as strategy-proofness.

The second important cost is the *strategic complexity* on agents, which is closely linked

to its game-theoretic properties. In particular, a mechanism in which every agent has a *dominant strategy*, i.e. an optimal strategy whatever the strategies and preferences of other agents, is useful game-theoretically *and* computationally. From a game-theoretic perspective, a dominant strategy equilibrium is a very robust solution concept, stable for example against “irrational” agents. From a computational perspective, with a dominant strategy an agent can avoid costly modeling and game-theoretic reasoning about other agents.

The third important cost is the *valuation complexity* on agents; valuation problems are often hard, agents have limited and/or costly computation, and there are many different outcomes in combinatorial domains. I considered two approaches: high-level bidding languages and dynamic methods.

High-level bidding languages, such as *bidding programs*, can help to address valuation complexity in situations in which it is easier to specify a method to compute values for outcomes than to compute values for every possible outcome. This might be the case if an agent has a well-formulated optimization problem to compute its value for different resource bundles. However, it is quite likely that each outcome has a different structure, and requires additional information and a new specification. Another problem with this approach is that it shifts computation to the center and is problematic from a privacy and informational perspective.

My dissertation suggests *dynamic mechanisms* as an alternative approach to address valuation complexity, in which agents reveal incremental information about their preferences until the mechanism can compute and verify an optimal solution. *iBundle* is an iterative combinatorial auction that solves realistic problems without complete information revelation from agents. The idea is to “open up” the algorithm for computing the outcome of the GVA, and involve agents dynamically in the computational process. It is easy to construct examples in which it is not necessary to have complete information about agents’ valuation problems to compute and verify the outcome of the auction. Some simple examples were described towards the end of Chapter 3. A well structured dynamic method will ask agents for just enough information to enable the mechanism to compute and verify the outcome.

Table 4.7 summarized the progress in iterative auction design over the past two decades.

Each contribution relaxes assumptions on agent preferences and/or strengthens the equilibrium analysis of the auction. All auctions terminate with efficient allocations and prices that are in competitive equilibrium, or equal to Vickrey payments, or both. Prior to *iBundle* there was no method that terminated in competitive equilibrium in the general problem, let alone in the minimal competitive equilibrium solution, or with Vickrey payments.

My approach to iterative combinatorial auction design integrates principles from linear programming and game-theoretic mechanism design.

First, I assumed that agents will follow a myopic best-response bidding strategy in every round of the auction. This allowed the direct application of linear programming theory, and primal-dual algorithms in particular, to auction design. With myopic best-response there is an easy mapping from a suitable primal-dual algorithm to an ascending-price combinatorial auction. The corresponding auction, *iBundle*, is the first efficient and iterative combinatorial auction for any reasonable agent bidding strategy, in this case myopic best-response.

Second, I extended the approach to compute Vickrey-Groves payments at the end of the auction, in addition to the efficient allocation. When successful, this makes myopic best-response an optimal *sequential* strategy for an agent in the auction, in equilibrium with best-response from every other agent. I derived a new primal-dual algorithm, VICKAUCTION, from a linear program formulation of the Groves mechanism for the combinatorial allocation problem. VICKAUCTION provably computes the Vickrey payments with only best-response information from agents. VICKAUCTION corresponds to *iBundle* with a second phase, Extend&Adjust. The purpose of the second phase is to elicit enough additional preference information from agents to compute Vickrey payments. It turns out that Vickrey payments require *more information* than that which is required to compute the efficient allocation. Vickrey-Groves payments are computed by an adjustment procedure from final prices, based on primal-dual information collected from best-response bids from agents.

Chapters 4 and 5 presented the *iBundle* auction, which computes efficient allocations with agents that follow myopic best-response bidding strategies. Chapter 4 reviews linear program models for the combinatorial allocation problem, and introduces a primal-dual algorithm, COMBAUCTION, that dynamically computes prices with enough structure to support the efficient allocation in competitive equilibrium. In some problems it is necessary

to use non-anonymous prices, with a different price for the same bundle to different agents, and non-linear prices, with the price on a bundle different from the total price over the items in a bundle.

COMBAUCTION has a natural interpretation as an ascending-price auction. The primal solution corresponds to a provisional allocation and the dual solution corresponds to bundle prices. Complementary slackness conditions demonstrate that a provisional allocation is efficient whenever: (1) every agent receives a bundle in its myopic best-response bid set, and (2) the allocation maximizes revenue for the auctioneer. This connection to linear programming theory proves the allocative-efficiency of *iBundle*.

Experimental results confirm that *iBundle* computes efficient allocations across a suite of problem instances with myopic best-response agent strategies, and with less information revelation than in the GVA. In addition, the effect of price discrimination on allocative efficiency is small, and I expect *iBundle* to perform well with anonymous prices in most problems. Approximations are possible within *iBundle*. For example, increasing the minimal bid increment ϵ in *iBundle* decreases the number of rounds and can provide an order-of-magnitude speed-up for small losses in allocative efficiency. Methods to leverage the sequential nature of winner-determination within *iBundle* were also studied, via caching and hot-start techniques.

Chapters 6 and 7 introduced *Extend&Adjust*, a method to extend *iBundle* and compute Vickrey payments at the end of the auction, in addition to the efficient allocation. The method justifies myopic best-response, making myopic best-response an optimal sequential strategy for an agent in equilibrium with best-response strategies from other agents. The design leaves *iBundle* basically unchanged, simply keeping the auction open for a few more rounds and then computing discounts to agents at the end of the second phase. In the end, agents are charged discounted prices based on discounts computed during the second phase.

First, I derived a linear program to compute *minimal* competitive equilibrium (CE) prices from a set of suitable competitive equilibrium prices and the efficient allocation. Minimal CE prices provide some protection against manipulation, and support the Vickrey payments in special cases. The linear program formulation leads to the procedure ADJUST, which computes minimal CE prices from the price information at the end of

COMBAUCTION, and similarly at the end of *i*Bundle. I characterized necessary and sufficient conditions under which ADJUST will compute minimal CE prices, and proved that *i*Bundle(3) (the variation with non-anonymous prices in all rounds) with ADJUST terminates with minimal CE prices. The discounted prices are equal to Vickrey payments in all problems for which minimal CE prices support Vickrey payments.

THEOREM 10.1 *iBundle and ADJUST is an iterative Vickrey auction when minimal CE prices support Vickrey payments.*

A fast but accurate method ADJPIVOT computes approximate discounts to agents after *i*Bundle terminates, checking complementary-slackness conditions only with respect to the “pivotal” allocations that were previously computed as provisional allocations during the auction.

Second, I derived a linear program formulation to compute the Vickrey payment of any one agent in all combinatorial allocation problem instances. The Vickrey payment to every agent can then be solved as a *sequence* of independent linear programs. This leads to a linear program to compute the Vickrey payment to an agent from a set of suitable competitive equilibrium prices and knowledge of the optimal allocation (but without additional information about agent valuation functions or the value of the optimal allocation). Finally, I proposed procedure ADJUST*, a slight variation on ADJUST, as a method to compute Vickrey payments from the price information at the end of an iterative auction. Considering necessary and sufficient conditions on CE prices for ADJUST* to compute minimal Vickrey payments, the most important non-obvious condition can be stated as follows:

... if an agent in the optimal allocation is not in one or more second-best allocations (without one of the agents in the allocation) then the price on the bundle it receives in the optimal allocation must equal its value.

This condition will not necessarily hold at the end of COMBAUCTION, or at the end of *i*Bundle. However, I was able to derive a primal-dual algorithm, VICKAUCTION, which computes Vickrey payments with ADJUST*, by proposing a second phase to COMBAUCTION, called PHASEII, which continues to adjust prices until the conditions for Vickrey payments are met. VICKAUCTION implements the allocation computed at the end of

COMBAUCTION, with adjusted prices based on discounts computed during PHASEII.

I prove optimality for VICKAUCTION:

THEOREM 10.2 *VICKAUCTION is a primal-dual algorithm to compute the Vickrey payments and efficient allocation in the combinatorial allocation problem, with only best-response information from agents.*

Significantly, VICKAUCTION, computes the Vickrey outcome with only best-response information from agents and without direct information about agent valuation functions.

Chapter 7 introduces an experimental auction method, *iBundle Extend&Adjust*, to implement VICKAUCTION in a decentralized system. The auction introduces a second phase to compute Vickrey payments, collecting just enough additional information from best-response agent strategies. The main difficulty in implementing the primal-dual method, VICKAUCTION, as an auction arises because it is important that agents can not detect the transition from Phase I to Phase II. The experimental method used to drive price competition in the extended phase of *iBundle* introduces *dummy agents* into the auction as real agents drop out, to: (a) provide enough competition to drive the prices to agents in the efficient allocation high enough to compute Vickrey discounts to every agent; and (b) provide a “competitive effect” that is hard to distinguish from the bids of the real agents they replace. The valuation functions of the dummy agents are configured by the auctioneer dynamically to mimic continued bidding from the real agents as they drop out.

I proved that the adjusted prices in *iBundle Extend&Adjust* are Vickrey payments when the auction terminates. The outstanding open question is whether the current rules for quiescence detection and dummy agents are sufficient to generate termination conditions in all problems.

Chapter 7 presented encouraging experimental results for *iBundle Extend&Adjust*. The extended auction is indeed able to compute the Vickrey outcome with myopic best-response agent strategies across a suite of problems. I make the following conjecture:

CONJECTURE 10.1 *iBundle Extend&Adjust is an iterative Generalized Vickrey Auction.*

Vickrey payments make myopic best-response becomes a Bayesian-Nash equilibrium of the auction.

THEOREM 10.3 *Myopic best-response is a Bayesian-Nash equilibrium of an iterative auction that myopically implements the outcome of the Generalized Vickrey Auction.*

In other words, myopic best-response is a sequentially rational strategy for an agent in such an auction, in equilibrium with myopic best-response strategies from every other agent.

Finally, proxy bidding agents are suggested as a method to boost the strategy-proofness provided by Vickrey payments. The proxy agents act as an interface between the auction and the agents, receiving incremental information from agents about their preferences over allocations, and following a myopic best-response strategy with that information. The proxy agents check that the information provided in each round is mutually-consistent, and only bid when there is enough information to determine the best-response.

Given proxy agents we have the following proposition:

PROPOSITION 10.1 *Dynamic truth-revelation is a dominant strategy to the proxy agents if: (a) the proxy agents can constrain agents to providing information consistent with a single ex ante fixed (but perhaps untruthful) valuation function; and (b) the auction implements an iterative GVA with myopic best-response.*

The proxy agents cannot actually limit agents to a single valuation function, but checking consistency across rounds should be quite effective at constraining any possible manipulation. The only gap that remains in the auction's strategy-proofness is the extent to which consistency cannot be completely enforced without falling back on complete information.

Chapter 8 proposed a new auction property, *bounded-rational compatibility*, to characterize auctions in which agents can compute equilibrium strategies with approximate information about their preferences. For example, an iterative auction is *myopic* bounded-rational compatible if an agent can compute its myopic best-response strategy with an approximate valuation function, for example bounds on its value, in some problems. *iBundle* is (myopic) bounded-rational compatible, while the GVA is not (dominant-strategy) bounded-rational compatible. The extended *iBundle* auction is certainly Bayesian-Nash bounded-rational compatible in problems in which it computes Vickrey payments with myopic best-response strategies, because myopic best-response is the equilibrium strategy

in this case.

To begin to understand the advantage of myopic best-response over complete revelation I compared the efficiency and computation in iterative and sealed-bid auctions for a simple model of a bounded-rational agent. I modeled the deliberation decision of an agent explicitly, and computed myopically-rational metadeliberation strategies for agents in iterative and sealed-bid auctions. The experimental results showed that: (a) iterative auctions compute more efficient solutions than sealed-bid auctions, with agents allocating limited computational resources to more “important” bundles; (b) iterative auctions allow agents to avoid value computation.

Finally, Chapter 9 presented an application of auction-based methods to a distributed train scheduling problem. Auction methods are well suited to the natural information and control structure of modern railroads. In the model, trains bid for times to enter and exit the territories of each dispatcher along their route, while each dispatcher operates an ascending-price auction, for the right to enter and exit its territory at a particular time. *i*Bundle style price-update rules are applied to adjust the price on pairs of entry and exit times across rounds. One innovation is that train agents can bid with an expressive constraint-based bidding language to represent indifference across times, e.g. “any arrival time before 12 pm is fine”, and without an imposed discretization on times. Computational results on a simple linear network, for train agents with myopic best-response bidding strategies demonstrated that the auction-based method computed better solutions than a centralized method and in less time and suggest that the auction method has useful scaling properties.

10.2 Future Work

Let me outline some areas for future work.

10.2.1 Iterative Combinatorial Auction Extensions

Expressive Bidding Languages

*i*Bundle provides agents with a complete, but not very expressive, exclusive-or (XOR) bidding language, which allows agents to bid for multiple bundles of items and specify they want at most one bundle. Expressive languages, for example constraint-based and

functional-approximation languages, can reduce the informational and computational demands on agents and lead to faster winner-determination algorithms.

One interesting approach to develop a mechanism for a new representation language is to understand how to adjust prices via a transformation of the representation into the *iBundle XOR/bundles* representation. It may be possible to derive tractable rules for an allocatively-efficient dynamic mechanism in the new representation. First convert bids in the new language into the *iBundle XOR/bundle* representation, then determine price-updates and the provisional allocation in the XOR/bundle representation, and then understand the rules for price-updates and rules for winner-determination in the new representation. Ideally we would like to exploit structure in agents' bids within the winner-determination and price-update problems in the auction, and *compile* this transformation procedure so that it is not necessary to work in the XOR/bundle representation when that is not useful computationally.

Consider a transformation from exclusive-or (XOR) to additive-or (OR) bids, as a simple example of this technique. Additive-or bids can be simulated in *iBundle* as bids from separate agents. This illustrates that the *iBundle* auction rules with additive-or agent bids (and without constructing explicit bid prices on bundles from agents' bids) are:

- increase prices on any single bundle in an OR bid for which an agent is unsuccessful in the current round
- never introduce price-discrimination
- terminate when every agent receives *all* bundles in its bid in the provisional allocation

Automated compilation methods could also be introduced, to allow a user to define a language semantics from which optimal auction methods are generated on-the-fly.

Application Study

One outstanding and important piece of experimental work is to perform a computational comparison of agent valuation complexity in *iBundle* and the GVA in a combinatorial allocation problem domain, with concrete models for local agent problems. Here are some problems with desirable properties (hard agent valuation problems, well-formed central optimization problems, natural decentralization, existing problem sets, etc.).

- *Distributed traveling salesperson problem.* Andersson & Sandholm [AS98, AS99, AS00] define a multiagent TSP in which agents have locations and jobs have locations. The goal is to allocate jobs to agents to maximize the social welfare, which is measured as the total distance traveled by all agents. The authors report results for a distributed task allocation method on 8x8 simulations, with random agent and job locations selected on a simple grid.
- *Multiple project resource management.* Projects may involve dozens of firms and hundreds of people who need to be managed and coordinated. Examples include large construction projects, opening a new store, performing major maintenance, starting up a new manufacturing facility [SBG94]. One well-formulated multi-agent problem is known as multiple-project resource-constrained optimization, in which individual projects compete for the same constrained resources and must solve local scheduling problems to minimize costs given allocations. The local problem is hard, and as Shutb *et al.* [SBG94] note, it is not realistic to solve to optimality with several hundred activities.
- *Multi-agent scheduling problems.* A classic example is the airport takeoff and landing time-slot problem [RSB82, GIP89]. Airlines compete for takeoff and landing slots at airports. For a particular allocation of slots the airline must solve its local crew and airplane scheduling problem to compute the cost for using the slots.

Sequential Winner Determination

The winner-determination problem in iterative auctions presents an interesting dynamic computational problem, because agents' bids change only gradually during the auction as prices increase. In addition to simple caching across rounds, and hot-start techniques, one might also look to compute and re-use solutions to subproblems. As an example, with branch-and-cut optimization, in which new constraints are introduced during linear-program based branch-and-bound search, one can re-use cuts to prune search in later rounds. Another interesting approach is that of "continual computation", in which probabilistic and game-theoretic methods are used to predict future winner-determination problems and to precompute solutions in the down time between rounds.

The Agent Metadeliberation Problem

Given that an agent has a hard valuation problem, limited computational resources, and many possible bundles of items to value, how should an agent schedule deliberation across different bundles? This is an interesting domain for metadeliberation techniques, such as those that have found application in other time critical domains, such as in medical [Hor87] and robot path planning domains [BD89]. An iterative auction provides a dynamic and time-critical environment, with prices increasing across rounds, and a finite time delay between rounds. Values for bundles are nested within an agent's algorithm to compute its best response, and the expected utility of submitting a correct best-response in the current round depends on beliefs about strategies of other agents and future prices in the auction.

An Asynchronous *i*Bundle Auction

It would be useful to prove properties about *i*Bundle when some agents do not submit a bid in each round, or do not follow their complete myopic best-response in each round. For example: can we characterize the problem sets in which *i*Bundle remains allocatively-efficient despite the existence of some “slow” agents in addition to some “fast” agents; can we describe an auction in which agents do not provide their complete exclusive-or bid set in each round, but can provide bids on additional bundles as the auction progresses?

10.2.2 Electronic Commerce Foundations

Multiattribute auctions and combinatorial exchanges present just two new emerging areas of computational mechanism design in electronic commerce.

Multiattribute Auctions

A multiattribute auction allows agents to negotiate over the attributes (size, terms-of-payment, delivery schedule, speed, etc.) of an item in addition to the price. Multiattribute auctions can lead to more efficient outcomes than fixed attribute auctions, in which sellers are restricted to price competition in an *ex ante* fixed space of attributes. Multiattribute auctions promise to provide robust methods for efficient automated negotiation between multiple agents.

There are three central challenges in the design of multiattribute auctions: (1) tractable

winner-determination, which depends on an agent's preferences over the attributes of a good in addition to price; (2) minimize the amount of information revelation required by agents; (3) handle issues of strategic misrepresentation of preferences.

Informationally, it will be important to design iterative mechanisms that allow agents to reveal incremental information about their preferences. Another method to reduce the informational load on users is to position a semi-autonomous proxy bidding agent between a user and the auctioneer that will accept many different types of information, including ordinal, cardinal, hard constraints, functional approximations, and then submit optimal bids to the auctioneer based on this information.

It appears possible to reduce simple multiattribute allocation problems (for example with one seller and multiple buyers) to a combinatorial allocation problem, at least if attributes are discrete, and leverage the *i*Bundle methodology. A bundle of attributes becomes a bundle of items. The preferences of the seller can be introduced via an agent that competes with buyers not to sell goods with particular attributes if the price is too low, or if another set of attributes are more desirable at the current prices. The Myerson-Satterthwaite impossibility theorem limits what can be achieved in a game-theoretic sense. We cannot expect to implement an efficient multiattribute auction that is incentive-compatible for the buyer and the seller *and* budget-balanced, and we must sacrifice one of these conditions. New approaches will be required when the attributes are continuous instead of discrete, and when the solution should allow aggregation across sellers. Both of these changes take the problem further away from the combinatorial auction problem.

Reverse auctions, such as procurement auctions, provide a similar opportunity for *i*Bundle-based methods. In a reverse combinatorial auction there is a single buyer and multiple sellers, each able to provide bundles of items. The efficient allocation will depend on the value of the buyer for different bundles of items and on the costs of each seller to provide bundles. The optimal solution selects bundles of items from multiple sellers to maximize the difference between the value of the buyer and the total cost over the sellers. Again, I believe that it is possible to use a transformation approach, to a regular "forward" combinatorial auction, and derive *i*Bundle price update rules for the reverse auction.

Combinatorial Exchanges

A combinatorial exchange allows multiple buyers to trade with multiple sellers, with all agents able to express logical conditions across bundles of items. Combinatorial exchanges address an important weakness in combinatorial auctions, namely the assumption that there a single seller is able to offer bundles composed of many different types of items. On the contrary, we might expect that a more natural model would allow multiple buyers to engage in combinatorial trades with multiple sellers. Interesting applications of such exchanges include: a bandwidth exchange that aggregates supply from geographically disparate sellers to match bids for “virtual networks”, or a travel exchange, that aggregates the supply of excess seats and hotel rooms, to match bids for bundles of rooms and flights. The winner determination problem in a combinatorial exchange, to select bids to maximize surplus, is a classic combinatorial optimization problem.

The pricing problem is interesting, in particular because of the Myerson-Satterthwaite impossibility theorem that shows that it is not possible to achieve efficiency, budget-balance, and incentive-compatibility. We must sacrifice one of these desirable properties. One approach would fix strategy-proofness, and perhaps sacrifice efficiency in favor of budget-balance. This is similar to an approach of McAfee [McA92] for double auctions on homogeneous items. Another approach would fix budget-balance, and try to achieve as much strategy-proofness and efficiency as possible. Parkes, Kalagananam and Eso [PKE01] take this approach, allocating surplus when an exchange is cleared to minimize the distance between agent payments and Vickrey payments. The choice of distance metric has a distributional effect on surplus allocation and changes the incentive properties of the exchange. A simple *threshold rule* appears to perform well, providing partial discounts to agents that would receive a large discount in a Vickrey-Groves scheme.

It remains an open problem to identify special cases in which budget-balance is not a problem in combinatorial exchanges, for example with restricted models of aggregation and bidding languages.

10.2.3 Approximations, Intractability, and Bounded-Rationality

Recent algorithmic approaches to mechanism design consider the effect of approximation and intractability on the economic properties of mechanisms. The goal is to understand both what is possible and what is impossible when tractable computation is introduced as

an additional constraint.

In one sense, bounded-rationality and intractability can help; one might use NP-hardness results to design “bounded strategy-proof” mechanisms that cannot be manipulated without an agent solving a hard problem in polynomial time.

In another sense, bounded-rationality and intractability can be a problem; optimal game-theoretic mechanisms can require that the network infrastructure solves multiple intractable problems, and approximate solutions can quickly break incentive-compatibility properties. In addition to proving worst-case manipulation results for approximation algorithms, one might also use randomized mechanisms to expand the implementation space.

Another thread in this interface between intractability and mechanism design is the effect of agent bounded-rationality on preference revelation. It is often impossible for an agent to compute its complete preference set, i.e. its value for all possible outcomes. The direction started on in this dissertation is to design mechanisms that can solve distributed problems with approximate information revelation from agents, providing dynamic feedback to agents about information collected from other agents and about progress towards a system-wide solution. My current research results suggest that primal-dual optimization theory may provide a suitable set of mathematical tools to develop iterative strategy-proof mechanisms, when used in combination with classic results from mechanism design.