Winner Determination in Combinatorial Auction Generalizations*

Tuomas Sandholm
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213
sandholm@cs.cmu.edu

Subhash Suri
Computer Science Department
University of California
Santa Barbara, CA 93106
suri@cs.ucsb.edu

Andrew Gilpin, David Levine
CombineNet, Inc.
311 S. Craig St.
Pittsburgh, PA 15213
{agilpin,dlevine}@CombineNet.com

ABSTRACT

Combinatorial markets where bids can be submitted on bundles of items can be economically desirable coordination mechanisms in multiagent systems where the items exhibit complementarity and substitutability. There has been a surge of research on winner determination in combinatorial auctions. In this paper we study a wider range of combinatorial market designs: auctions, reverse auctions, and exchanges, with one or multiple units of each item, with and without free disposal. We first theoretically characterize the complexity of finding a feasible, approximate, or optimal solution. Reverse auctions with free disposal *can be approximated* (even in the multi-unit case), although auctions cannot. When XOR-constraints between bids are allowed (to express substitutability), the hardness turns the other way around: even finding a feasible solution for a reverse auction or exchanges is $\mathcal{NP}$-complete, while in auctions that is trivial. Finally, in all of the cases without free disposal, even finding a feasible solution is $\mathcal{NP}$-complete.

We then ran experiments on known benchmarks as well as ones which we introduced, to study the complexity of the market variants in practice. Cases with free disposal tended to be easier than ones without. On many distributions, reverse auctions with free disposal were easier than auctions with free disposal—as the approximability would suggest—but interestingly, on one of the most realistic distributions they were harder. Single-unit exchanges were easy, but multi-unit exchanges were extremely hard.

Categories and Subject Descriptors

I.2 [**Computing methodologies**]: Artificial intelligence; I.2.11 [**Artificial intelligence**]: Distributed artificial intelligence—*Multiagent systems, Coherence and coordination*; I.2.8 [**Computing methodologies**]: Artificial intelligence—*Problem solving, control methods, search*; F.2 [**Theory of computation**]: Analysis of algorithms and problem complexity; J.4 [**Computer applications**]: Social and behavioral sciences—*Economics*

*This work was funded by, and conducted at, CombineNet, Inc., 311 S. Craig St., Pittsburgh, PA 15213. An early version of this paper appeared in the AGENTS-01 Workshop on Agents for B2B, pp. 35–41, Montreal, Canada, May, 2001.

1. INTRODUCTION

Combinatorial markets can be used to reach economically efficient allocations of goods, services, tasks, resources, etc., in multiagent systems even when the agents' valuations for bundles of items are not additive. Some items can be complementary, and others can be substitutes.

While combinatorial markets have major economic advantages, they can be computationally complex to clear. There has been a recent surge of interest in developing combinatorial clearing algorithms [14, 16, 3, 10, 17, 1, 8, 2, 13, 19]. However, the bulk of this work has focused on single-unit combinatorial auctions with free disposal, with some work on multi-unit combinatorial auctions with free disposal [15, 17, 12, 5]. Certain other generalizations have also been discussed, but their complexity has not been analyzed theoretically or experimentally [15, 17].

In this paper we study the complexity of the other main variants of combinatorial markets. We study auctions, reverse auctions, and exchanges. In each setting we study the single-unit as well as the multi-unit case. We analyze each of these variations with and without free disposal.[1] This leads to $3 \times 2 \times 2 = 12$ important settings, of which only 2 have received significant attention so far. We also study the theoretical impacts of XOR-constraints among bids, in terms of complexity of finding a feasible, approximate, or optimal solution.

We first define the different market types. For each market type, we theoretically determine the complexity of finding a feasible, approximate, or optimal solution. We then compare the types experimentally.

2. CLASSES OF COMBINATORIAL MARKETS

In this section we introduce different combinatorial market types, and discuss the complexity of winner determination from a theoretical perspective.

[1]We use a strong version of the *no free disposal* case. If there is no free disposal, the sellers have to sell everything and the buyers cannot accept anything extra beyond what they bid on. In the future, we plan to also study the case where disposal is neither free nor impossible, but rather between these two ends of the spectrum. For example, disposal could have a predetermined cost.

2.1 Single-Unit Auctions

The most basic combinatorial auction, and the type that has received most of the attention in previous work [16, 3, 17], is a single-unit combinatorial auction with free disposal.

Definition 1. *The auctioneer has a set of items,* $M = \{1, 2, \ldots, m\}$*, to sell, and the buyers submit a set of bids,* $\mathcal{B} = \{B_1, B_2, \ldots, B_n\}$*. A bid is a tuple* $B_j = \langle S_j, p_j \rangle$*, where* $S_j \subseteq M$ *is a set of items and* $p_j \geq 0$ *is a price. The* binary combinatorial auction winner determination problem (BCAWDP) *is to label the bids as winning or losing so as to maximize the auctioneer's revenue under the constraint that each item can be allocated to at most one bidder:*

$$\max \sum_{j=1}^{n} p_j x_j \quad s.t. \quad \sum_{j | i \in S_j} x_j \leq 1, \; i = 1, 2, \ldots, m$$
$$x_j \in \{0, 1\}$$

If there is no free disposal (auctioneer is not willing to keep any of the items, and bidders are not willing to take extra items), an equality is used in place of the inequality.

By now it is well known that (the decision version of) BCAWDP with free disposal (even with integer prices) is $\mathcal{NP}$-complete [14]. It cannot even be approximated to a ratio of $n^{1-\epsilon}$ in polytime (unless $\mathcal{P} = \mathcal{ZPP}$) — as shown in [16] via an approximation-preserving reduction from MAX CLIQUE which is inapproximable [6]. However, finding a feasible solution is trivial (if there is free disposal): any bid alone constitutes a feasible solution. Another trivial feasible solution is that where no bids are accepted.

2.2 Multi-Unit Auctions

When there are multiple indistinguishable goods for sale, it is usually desirable (from a bid compactness and winner determination complexity perspective) to represent these goods as multiple *units* of a single item, rather than as multiple items. Different items can have multiple units each, where units of one item are indistinguishable but units of different items are distinguishable. This representation allows a bidder to place a single bid requesting the amount of each item that he wants, instead of placing separate bids on the potentially enormous number of combinations that would amount to those numbers of units of those items. An auction that allows this type of bidding is called a *multi-unit combinatorial auction*. They have been used, for example, in the *eMediator* ecommerce server prototype [15], and recent research has studied winner determination in this context [17, 12, 5]. Multi-unit auctions have many potential real-world applications including bandwidth allocation and electric power markets. The winner determination problem for multi-unit auctions follows.

Definition 2. *The auctioneer has a set of items,* $M = \{1, 2, \ldots, m\}$*, to sell. The auctioneer has some number of units of each item available:* $U = \{u_1, u_2, \ldots, u_m\}, u_i \in \Re^+$*. The buyers submit a set of bids,* $\mathcal{B} = \{B_1, B_2, \ldots, B_n\}$*. A bid is a tuple* $B_j = \langle (\lambda_j^1, \lambda_j^2, \ldots, \lambda_j^m), p_j \rangle$*, where* $\lambda_j^k \geq 0$ *is the number of units of item k that the bid requests, and* $p_j \geq 0$ *is the price. The* binary multi-unit combinatorial auction winner determination problem (BMUCAWDP) *is to label the bids as winning or losing so as to maximize the auctioneer's revenue under the constraint that each unit of an item can be allocated to at most one bidder:*

$$\max \sum_{j=1}^{n} p_j x_j \quad s.t. \quad \sum_{j=1}^{n} \lambda_j^i x_j \leq u_i, \; i = 1, 2, \ldots, m$$
$$x_j \in \{0, 1\}$$

If there is no free disposal (auctioneer is not willing to keep any units, and bidders are not willing to take extra units), an equality is used in place of the inequality.

Proposition 2.1. *Consider BMUCAWDP with free disposal. The decision problem is* $\mathcal{NP}$*-complete. The optimization problem cannot be approximated to a ratio* $n^{1-\epsilon}$ *in polynomial time unless* $\mathcal{P} = \mathcal{ZPP}$*. Both claims hold even with integer prices and integer units.*

Proof. Immediate from the $\mathcal{NP}$-completeness and inapproximability of BCAWDP since that is a special case ($u_i = 1$ for all $i \in \{1, 2, \ldots, m\}$). □

Again, finding a feasible solution is trivial: even any bid alone would constitute a feasible solution.

2.3 Reverse Auctions

In many market scenarios, for example in procurement, there is a buyer who wants to obtain some goods at the lowest possible cost, and a set of sellers who can provide the goods. The buyer can hold a *reverse auction* to try to obtain the goods. Again, if there is complementarity or substitutability between the goods, a *combinatorial* reverse auction can be beneficial. Each seller submits "asks" that say how much the seller asks for each bundle of goods she can provide. A single-unit combinatorial reverse auction is a special case of a multi-unit combinatorial reverse auction, so we only present the latter formally.

Definition 3. *The auctioneer (buyer) has a set of items,* $M = \{1, 2, \ldots, m\}$ *that she wishes to obtain. She specifies how many units of each item she wants:* $U = \{u_1, u_2, \ldots, u_m\}$*,* $u_i \in \Re^+$*. The sellers submit a set of asks,* $\mathcal{A} = \{A_1, A_2, \ldots, A_n\}$*. An ask is a tuple* $A_j = \langle (\lambda_j^1, \lambda_j^2, \ldots, \lambda_j^m), p_j \rangle$*, where* $\lambda_j^k \geq 0$ *is the number of units of item k offered by the ask. The ask price is* $p_j \geq 0$*. The* binary multi-unit combinatorial reverse auction winner determination problem (BMUCRAWDP) *is to label the asks as winning or losing so as to minimize the auctioneer's cost under the constraint that the auctioneer receives all of the units of items that she is asking:*

$$\min \sum_{j=1}^{n} p_j x_j \quad s.t. \quad \sum_{j=1}^{n} \lambda_j^i x_j \geq u_i, \; i = 1, 2, \ldots, m$$
$$x_j \in \{0, 1\}$$

If there is no free disposal (sellers are not willing to keep any units of their winning asks, and the buyer is not willing to take extra units), an equality is used in place of the inequality.

Proposition 2.2. *With free disposal, (the decision version of) BMUCRAWDP is* $\mathcal{NP}$*-complete both in the single-unit and the multi-unit case. This holds even for integer prices and integer units.*

PROOF. The decision version of BMUCRAWDP (even in the multi-unit case) is in $\mathcal{NP}$ because the solution can easily be checked in polynomial time. To prove the theorem, we then only need to show that the single-unit case is $\mathcal{NP}$-hard. We observe that the single-unit case is exactly the same problem as WEIGHTED SET *COVERING*.[2] Since WEIGHTED SET COVERING is $\mathcal{NP}$-complete, the single-unit combinatorial reverse auction is $\mathcal{NP}$-complete as well. □

Interestingly, *while winner determination is inapproximable in a combinatorial auction (even in the single-unit case), in a combinatorial reverse auction the winners can be approximated quite well (even in the multi-unit case)!* We present a greedy approximation algorithm for *multi-unit* combinatorial reverse auctions.[3] As before, let there be m items to be bought. Let u_i be the number of units of item i to be purchased. Let size(A_j) be the total number of units in ask A_j, that is, size$(A_j) = \sum_{i=1}^{m} \lambda_j^i$.

In the algorithm, F_i is the number of units remaining to be covered for item i. Similarly, rsize(A_j) is the total number of units in ask A_j, *not* counting units that are already covered. That is, rsize$(A_j) = \sum_{i=1}^{m} \min\{\lambda_j^i, F_i\}$.

ALGORITHM 2.1.

$ACCEPTED = \emptyset$

For all i, $F_i = u_i$

For all j, rsize$(A_j) = size(A_j)$

While some $F_i > 0$

  If every ask is in $ACCEPTED$, *return* $INFEASIBLE$

  $j^* \leftarrow \min_{j | A_j \notin ACCEPTED} \frac{p_j}{rsize(A_j)}$

  Add A_{j^*} *to* $ACCEPTED$

  For all i, $F_i \leftarrow F_i - \lambda_{j^*}^i$

  For all j, rsize$(A_j) \leftarrow \sum_{i=1}^{m} \min\{\lambda_{j^*}^i, F_i\}$

Return $ACCEPTED$, *that is, the set of accepted asks*

PROPOSITION 2.3. *If a solution exists, Algorithm 2.1 finds a solution that is within a factor* $(1+\ln N)$ *of optimal, where* N *is the total number of* units *being bought in the reverse auction.*[4] *Otherwise the algorithm returns* $INFEASIBLE$.

In order to prove the claim, we use the following auxiliary fact. When the algorithm picks ask A_j, define $\gamma = \frac{p_j}{rsize(A_j)}$. Assign $w(e) = \gamma$ for each unit e newly covered by A_j. Note that the total sum of these weights assigned when A_j is picked is p_j, because A_j covered $rsize(A_j)$ new units.

Number the units in the order they are covered by our algorithm, breaking ties arbitrarily. Let this order be $e_1, e_2, \ldots, e_N$.

LEMMA 2.4. $w(e_k) \leq \frac{OPT}{(N-k+1)}$, *where* OPT *is the total cost of the optimal cover.*

PROOF. Suppose e_k was first covered when ask A_j was picked. At this point, the number of remaining (uncovered) units is at least $N - k + 1$. Now, consider any optimal solution. Since that solution can obviously cover all these $N-k+1$ units at cost OPT, there must be at least one ask whose *average* cost of covering is at most $\frac{OPT}{(N-k+1)}$. Since our greedy algorithm chooses the set that covers most cost-effectively, we must have that $\gamma = \frac{p_j}{rsize(A_j)} \leq \frac{OPT}{(N-k+1)}$. Since the weight assigned to e_k is γ, the claim follows. □

PROOF. (of Proposition 2.3). Let $A_{j_1}, A_{j_2}, \ldots, A_{j_q}$ be the asks picked by Algorithm 2.1. The cost of this solution is $\sum_r p_{j_r}$. Because p_{j_r} is evenly distributed among the units that are newly covered by A_{j_r}, we also have that $\sum_r p_{j_r} = \sum_{j=1}^{N} w(e_j)$. Using the previous lemma, we get:

$$\begin{aligned} \sum_{j=1}^{N} w(e_j) &\leq OPT \cdot (1 + 1/2 + 1/3 + ... + 1/N) \\ &\leq OPT \cdot H_N, \end{aligned}$$

where H_N is the harmonic number. Finally, the claim follows from the fact that $H_N \leq 1 + \ln N$. □

With free disposal, finding a *feasible* solution (if one exists) to a combinatorial reverse auction is trivial, even in the multi-unit case. For example, one can simply accept all the asks. If this solution is not feasible, then no other solution is either.

2.4 Exchanges

In markets with many buyers and many sellers, exchanges are a natural choice for a market mechanism. In a *combinatorial exchange* [15, 17], the trades that the market determines to occur can involve multiple buyers and multiple sellers each. Unlike auctions and reverse auctions, there is no auctioneer in a combinatorial exchange. Rather the participants in the exchange are allowed to both buy and sell items, or just buy or just sell. Both auctions and reverse auctions are special cases of exchanges. Also, the single-unit exchange is a special case of the multi-unit exchange (where each demand $\lambda_j^k \in \{-1, 0, 1\}$) so we only present the multi-unit exchange formally.

DEFINITION 4. *The administrator of an exchange determines which items will be available in the exchange,* $M = \{1, 2, \ldots, m\}$. *Only these items may be included in the bids and asks in the exchange. A bid* [5] *in this setting is* $B_j = \langle(\lambda_j^1, \lambda_j^2, \ldots \lambda_j^m), p_j\rangle$, *where* $\lambda_j^k \in \Re$ *is the requested number of units of item k, and* $p_j \in \Re$ *is the price. A positive*

[2] This is a different problem than WEIGHTED SET PACKING, which is analogous to BCAWDP with free disposal.

[3] The special case of this algorithm for *single-unit* combinatorial reverse auctions is the same as the classic greedy algorithm for WEIGHTED SET COVERING. It is known that it achieves a $(1+\ln m')$-approximation for the problem, where m' is the largest number of items that any one ask contains [7].

[4] With a more complex analysis of the same algorithm, we can tighten the approximation guarantee to $(1+\ln N')$ where N' is the largest number of units that *any one ask contains.* We omit that complex proof due to limited space. Furthermore, an $O(\ln m)$ approximation ratio can be achieved (using scaling and reduction [20]), where m is the number of *items* in the auction.

[5] We will often simply refer to "bids" rather than "bids" and "asks" when the distinction between the two is unnecessary.

λ_j^k *represents buying and a negative* λ_j^k *means selling. A positive* p_j *represents bidding while a negative* p_j *means asking. The* binary multi-unit combinatorial exchange winner determination problem (BMUCEWDP) *is to label the bids as winning or losing so as to maximize surplus under the constraint that demand does not exceed supply:*

$$\max \sum_{j=1}^{n} p_j x_j \quad s.t. \quad \sum_{j=1}^{n} \lambda_j^i x_j \leq 0 \qquad i = 1, 2, \ldots, m$$

If there is no free disposal (buyers are not willing to take extra units, and sellers are not willing to keep any units of their winning bids), an equality is used in place of the inequality.

PROPOSITION 2.5. *Consider BMUCEWDP with free disposal (in the single- or multi-unit case). The decision problem is* $\mathcal{NP}$*-complete. The optimization problem cannot be approximated to a ratio* $n^{1-\epsilon}$ *in polynomial time unless* $\mathcal{P} = \mathcal{ZPP}$*. Both claims hold even with integer prices and integer units.*

PROOF. Immediate from the $\mathcal{NP}$-completeness and inapproximability of BCAWDP since that is a special case. □

2.5 XOR-Constraints

Basic combinatorial auctions allow bidders to express complementarity between items (value of a bundle being greater than the values of its parts). However, consider a bidder that has submitted a bid of \$4 for {1}, a bid \$5 for {2}, and a bid \$7 for {1,2}. Now the auctioneer could allocate 1 and 2 to the bidder for \$9. So, the basic bidding language does not allow the bidder to express substitutability (value of a bundle being less than the sum of its parts). Full expressivity of the bidding language is obtainable by allowing *XOR-constraints* between bids [16, 3, 15, 13].

XOR-constraints can also be used in combinatorial reverse auctions and combinatorial exchanges to achieve full expressiveness, but they have interesting implications on the complexity of winner determination.

Remark: In our model, a bidder[6] can insert XOR-constraints between any pairs of his bids he wants, but does not have to insert XOR-constraints between *all* pairs of his bids.

In combinatorial auctions, XOR-constraints do not change the theoretical complexity of clearing. This is understandable since the winner determination problem with XOR-constraints can be encoded in polynomial time and space into a winner determination problem without XOR-constraints (simply add a dummy item for each XOR-constraint, and have the bids that are XOR'ed include that dummy item). So, it is trivial to find a feasible solution (accept no bids), hard to find an approximation better than $n^{1-\epsilon}$, and $\mathcal{NP}$-complete to solve the problem optimally.

Interestingly, although basic combinatorial reverse auctions are approximable as we showed above, even finding a feasible solution is $\mathcal{NP}$*-complete in a combinatorial reverse auctions with XOR-constraints. Put in another way, without XOR-constraints, combinatorial reverse auctions are easier to clear than combinatorial auctions, but with XOR-constraints, combinatorial reverse auctions are harder to clear than combinatorial auctions! Note that with XOR-constraints, even* single-unit *combinatorial reverse auctions are harder to clear than* multi-unit *combinatorial auctions.*

THEOREM 2.6. *Finding a feasible solution in a combinatorial reverse auctions with XOR-constraints is* $\mathcal{NP}$*-complete (even in the single-unit case, even with integer prices).*

PROOF. Clearly this problem is in $\mathcal{NP}$ because feasibility of a given solution can easily be checked in polynomial time. So, the beef is to prove that it is $\mathcal{NP}$-hard. We show this using a reduction from EXACT COVER BY 3-SETS problem [4], which is $\mathcal{NP}$-complete. In that problem, there is a ground set, and a set of subsets of the ground set, where each subset includes 3 items from the ground set. The question is whether all the items of the ground set can be covered by non-overlapping subsets. Given an instance of EXACT COVER BY 3-SETS, we construct an instance of a combinatorial reverse auctions with XOR-constraints as follows. For every one of the subsets given in the EXACT COVER BY 3-SETS we construct a bid that includes the corresponding 3 items. If two subsets share items, we insert an XOR-constraint between them. Now, the reverse auction instance has a feasible solution if and only if the instance of EXACT COVER BY 3-SETS has a solution. □

Since combinatorial exchanges encompass combinatorial reverse auctions as a special case, finding a feasible solution in a combinatorial exchange with XOR-constraints is $\mathcal{NP}$-complete. Since finding a feasible solution in combinatorial reverse auctions and exchanges with XOR-constraints is $\mathcal{NP}$-complete, it follows immediately that finding any approximation to those problems is $\mathcal{NP}$-complete.

Combinatorial auctions, reverse auctions and exchanges are optimally clearable in polynomial time using linear programming, *if bids can be accepted partially.* However, a recent paper shows that XOR-constraints make the clearing problem $\mathcal{NP}$-complete even in that case [9]. The implications of other side constraints on market clearing complexity have also recently been studied [18].

2.6 Lack of Free Disposal

Free disposal refers to the property that each party prefers (possibly not strictly) more to less. In other words, for each item, there is at least one party in the market who can dispose of any number of units of that item for free. Each winner determination problem discussed so far can be changed to reflect the case where items do not exhibit free disposal by simply changing the inequalities in the integer programming formulations to equalities. Despite the apparent similarities in the integer programming formulations for markets with and without free disposal, the problems are quite different.

In general, an auction cannot be formulated as a reverse auction (e.g., by simply changing signs) with the expectation that the solution for the reverse auction will be the same as for the auction. This is because in the reverse auction we are looking for lower priced bids, while in the auction we are looking for higher priced bids. The winning bid sets differ even if prices were negated. In the case of no free disposal (even without negating prices), the set of feasible solutions in

[6] The auctioneer could also insert XOR-constraints. From the perspective of winner determination, it does not matter where the constraints come from.

an auction is the same set as in a reverse auction, but the set of optimal solutions is generally different. As we will show in the experiments, the time required to solve auctions and reverse auctions without free disposal can be very different.

We now characterize the complexity of the winner determination problem without free disposal.

THEOREM 2.7. *Consider the winner determination problem in a combinatorial auction (single-unit or multi-unit), combinatorial reverse auction (single-unit or multi-unit), or a combinatorial exchange (single-unit or multi-unit). Without free disposal, even finding a feasible solution is $\mathcal{NP}$-complete (even with integer prices and integer units).*

PROOF. Clearly these problems are in $\mathcal{NP}$ because feasibility can easily be checked in polynomial time. So, the beef is to prove that they are $\mathcal{NP}$-hard. We do this by showing that the following special case is already $\mathcal{NP}$-hard. Let every bid have exactly three items, and price 1. Let the number of items be a multiple of 3. Now, if we had a polynomial time algorithm to find a feasible solution for this problem, we could use that algorithm directly to solve the EXACT COVER BY 3-SETS problem [4], which is $\mathcal{NP}$-complete. □

Now, let us go through an example to see how likely it is that a randomly chosen problem instance is feasible. Consider an auction (or a reverse auction) where each bid is randomly assigned σ items without replacement (and no duplicate bids are allowed). Modulo pricing, there are $\binom{m}{\sigma}$ possible bids. Thus there are $\binom{\binom{m}{\sigma}}{n}$ problem instances. Now let us compute the number of feasible instances. In a feasible solution, each item is allocated to one bid. Consider a set of winning bids, in some particular order. The first bid's first item could be any of m items, the second item could be any one of the remaining $m-1$, etc. The first bid's last item could be any one of $m-\sigma+1$. The second bid's first item could be any one of the remaining $m-\sigma$, etc. So, together there are $m!$ feasible instances. (Note that this is independent of n and σ.) So, the fraction of instances that are feasible is $m!/\binom{\binom{m}{\sigma}}{n}$.

COROLLARY 2.1. *Without free disposal, the winner determination problem in a combinatorial auction (single-unit or multi-unit), combinatorial reverse auction (single-unit or multi-unit), or a combinatorial exchange (single-unit or multi-unit) cannot even be approximated in polynomial time (unless $\mathcal{P} = \mathcal{NP}$), even with integer prices and integer units.*

PROOF. Immediate from Theorem 2.7 □

In the rest of the paper we present experiments to see how hard these variants of the winner determination problem are in practice.

3. EXPERIMENTS

We designed the experiments so that each one would help illustrate the computational differences between a feature of an auction or exchange. We compared auctions and reverse auctions to see whether the fundamental difference in approximability shows up in practice. We compared free disposal and no free disposal. We also showed the hardness of exchanges. In our experiments, the units are integers, but the prices are reals.

All of the tests were run on a Pentium III 933 MHz processor, with 512 MB RAM. The test machine was running Linux 2.2. The algorithm that was used to solve the problems was CPLEX 7.0, a general-purpose mixed integer programming package.[7] CPLEX has recently been used to benchmark winner determination in the context of combinatorial auctions [1]. CPLEX uses a variant of A* search as its search algorithm (other search strategies are also available in CPLEX, but the default strategy turned out to be faster than any of the other variants that we experimented with). At every search node, a linear programming (LP) relaxation of the remaining subproblem is used to construct a heuristic upper bound. If the LP happens to return an integer solution, that is the optimal solution to that subproblem, so the subtree rooted at that node need not be searched. Quite frequently this occurs already at the root, in which case no search is conducted [19].

In all of our experiments, for any given parameter setting, it took CPLEX significantly longer to find an optimal solution than it took to prove infeasibility. In the cases with free disposal, the problem was never infeasible (infeasibility could only happen if in a reverse auction there are not enough units of some item in all of the bids combined). In the cases without free disposal, the constraints are all equalities, and CPLEX is quite effective at using them algebraically to reduce search. On distributions where CPLEX tended to find an integer solution with LP directly (and search was therefore not needed), CPLEX also was able to prove infeasibility without search. On the other hand, on distributions where CPLEX conducted search to find an optimal solution, it tended to also require search to prove infeasibility. To keep the times comparable, in all of the experiments, we only report execution times for feasible instances.

We ran experiments on several benchmark distributions. All of the values reported in the graphs are means over 50 instances.

3.1 Single-Unit Auctions and Reverse Auctions

We used the following common benchmark distributions for single-unit auctions [16]:

- **Random:** For each bid, pick the number of items randomly from $1, 2, ..., m$. Randomly choose that many items without replacement. Pick the price randomly from $[0, 1]$.
- **Weighted random:** As above, but pick the price between 0 and the number of items in the bid.
- **Uniform:** Draw the same number of randomly chosen items for each bid. Pick the prices from $[0, 1]$.
- **Decay:** Give the bid one random item. Then repeatedly add a new random item with probability α until an item is not added or the bid includes all m items. Pick the price between 0 and the number of items in

[7]Recently ILOG released CPLEX version 7.1. To make sure that the problems that we claimed to be hard are hard for the newest version also, we reran all the hard cases (multi-unit exchanges), and some of the easy cases, using CPLEX 7.1. Figure 3 top left, Figure 7, and Figure 8 are for CPLEX 7.1.

the bid. In the tests we used $\alpha = .75$ since the graphs in [16] show that this setting leads to the hardest (at least for their algorithm) instances on average.

Previously these distributions have only been used for single-unit auctions with free disposal. We use these distributions to benchmark reverse auctions as well. When using the **uniform** distribution with no free disposal, we show experiments where the bid size is a factor of the total number of items—otherwise there is no feasible solution.

It is clear from Figure 1 that there is a complexity difference between auctions with and without free disposal. In fact, CPLEX takes two orders of magnitude longer to solve no free disposal auctions and reverse auctions on the **random** distribution. Although the difference is less dramatic on the **weighted random** distribution, it is still present. On both **random** and **weighted random**, reverse auction with free disposal rarely require search, and if they do, only a few nodes. Auctions with free disposal require search a bit more often, and use a somewhat larger number of nodes when search does occur. In free disposal settings, auctions and reverse auctions lead to search almost every time, and the search trees are large.

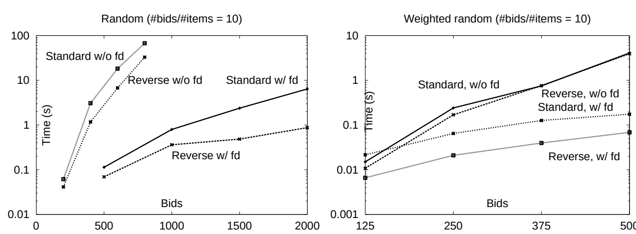

Figure 1: *Run times on the* **random** *and* **weighted random** *distributions.*

Another interesting thing to note is that auctions without free disposal almost consistently take longer to solve than reverse auctions without free disposal for these two distributions. The easiest market type to solve was the reverse auction with free disposal. This is not surprising in light of its approximability.

In Figure 2 we again see the clear difference in execution time with and without free disposal. What is surprising here is that in the **decay** distribution, reverse auctions take much longer than standard auctions, even in the case with free disposal. This is the exact opposite of what we saw in Figure 1 and what we see on the **uniform** distribution. That shows that the theoretical approximability does not always translate to shorter solution times when going for an optimal solution.

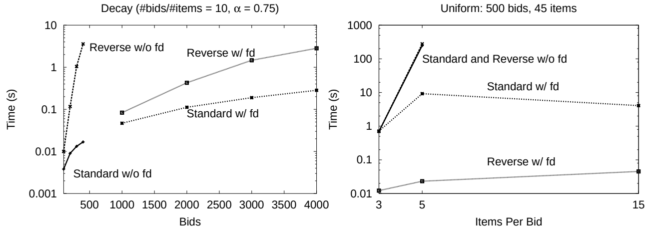

Figure 2: *Run times on the* **decay** *and* **uniform** *distributions.*

On the **uniform** distribution, for auctions and reverse auctions with no free disposal, there are no instances represented for bid size 15. That is because all of those instances were infeasible.

3.2 Multi-Unit Auctions and Reverse Auctions

We ran multi-unit auction experiments on two distributions. We used the same distributions for reverse auctions (by negating the prices of the bids)—a setting on which we have not seen any benchmark results before.

- **Decay-decay:** First assign the number of units for each item i: let item i have 1 unit. Repeatedly add another unit with probability α_0. Then, give the bid one random item. Then repeatedly add a new random item (without repetition) with probability α_1. Finally, for each item i in the bid, give that item quantity 1, then repeatedly add 1 to the quantity with probability α_2. If the quantity is greater than u_i, then set the quantity equal to u_i. The price is computed by taking a random number between and 0 and 1 and multiplying by the total number of units in the bid. (A similar distribution appeared in [12].) We used $\alpha_0 = .99$, and varied α_1 and α_2.

- **CATS multipaths:** This distribution models what might happen in an auction such as network bandwidth allocation [11]. As far as we know there have not been any performance results published for this distribution.

Figure 3 shows that optimal clearing scales well on multi-unit auctions and reverse auctions on the **decay-decay** distribution. In each of the graphs we see that reverse auctions

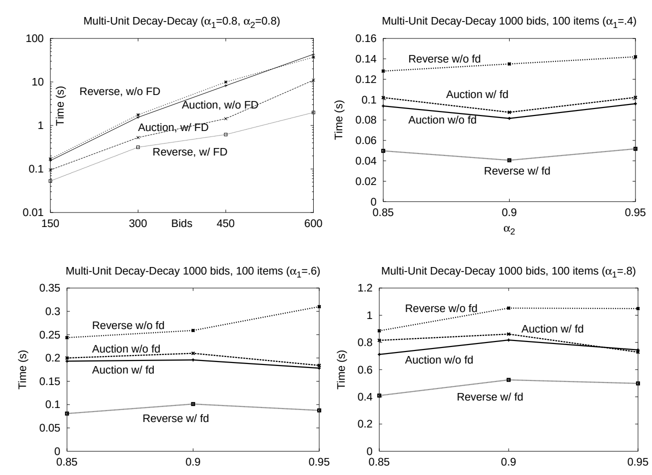

Figure 3: *Top left: scaling of optimal clearing on the* **decay-decay** *distribution. We used $\alpha_1 = \alpha_2 = 0.8$ (this is the hardest setting according to our experiments). Other figures: run-time of optimal clearing on the* **decay-decay** *distribution with varying $\alpha_1 \in \{.4, .6, .8\}$.*

with free disposal are routinely solved the fastest. Reverse auctions without free disposal are the slowest. In auctions, free disposal is only slightly faster than no free disposal. Before we ran these experiments we thought that auctions and reverse auctions without free disposal would have very similar run times since the set of feasible solutions is identical in these problems (the optimal solution is generally a different solution within that set). We conjecture that a specialized algorithm could mitigate the difference between the two. In any case, all of these instances were easy. For example at $\alpha_1 = .6,\ \alpha_2 = .9$, the LP solver of CPLEX returned integer solutions up front (and therefore, no search was needed) on 74% of the reverse auctions with free disposal, on 52% of the auctions with free disposal, on 50% of the auctions without free disposal, and on 22% of the reverse auctions without free disposal.

Without free disposal (and in reverse auctions even with free disposal), the instances from **CATS multipaths** were almost all infeasible. On auctions with free disposal, Figure 4 shows that CPLEX's execution time grows rapidly with the number of bids. We observed that with as few as 2,000 bids, the main memory of our test machine could get exhausted, resulting in very poor performance due to paging. Clearly there is room for improvement in scalability on this distribution.

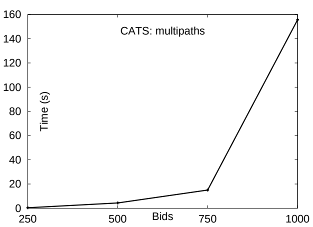

Figure 4: *Run times on multi-unit combinatorial auctions with free disposal from the CATS multipaths distribution.*

3.3 Single- and Multi-Unit Exchanges

We have not seen any benchmarks on combinatorial exchanges before. Therefore, we introduce a new benchmark distribution which is similar to the **decay-decay** distribution for multi-unit auctions.

- **Exchange decay-decay:** For each bid, assign it one item. Repeatedly add an additional item with probability α_1. For each item i in the bid, assign one initial unit and repeatedly add an additional unit of that item with probability α_2. With probability .5, negate the quantity of the item to indicate selling the item. The price is a random number between 0 and 1, multiplied by the net number of units in the bid (which is negative as often as it is positive). This distribution yields a single-unit exchange when $\alpha_2 = 0$.

Figure 5 (left) shows that free disposal makes almost no difference in single-unit exchanges (for any value of α_1). The graph on the right shows that CPLEX scales very well on *single-unit* exchanges.

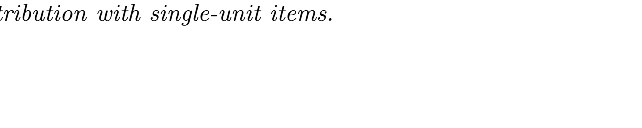

Figure 5: *Run times on the* **exchange decay-decay** *distribution with single-unit items.*

However, CPLEX scales extremely poorly on *multi-unit* exchanges. Even with just 10 items and 100 bids, it takes a long time. As the number of bids increases further, the run time increases extremely rapidly. CPLEX quickly becomes unusable for the harder parameter settings in the distribution (the hardest setting is $\alpha_1 = \alpha_2 = .8$). Figure 6 shows that the complexity increases drastically in both α_1 and α_2, i.e., when each bid specifies supply and demand on a large number of items and units. Both with and without free disposal, the run time increases *super-exponentially* in those parameters. The parameter α_1 is especially critical, as shown by the run time difference between Figure 6 Left and Figure 6 Right.

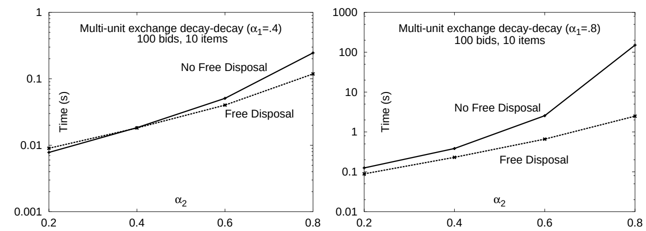

Figure 6: *Run times on the* **exchange decay-decay** *distribution with multi-unit items.*

As Figure 7 shows, optimal clearing scales extremely poorly on exchanges. As the number of bids increases further, the run time increases extremely rapidly. Even with only 12 items and 120 bids, winner determination takes a long time, even with free disposal.

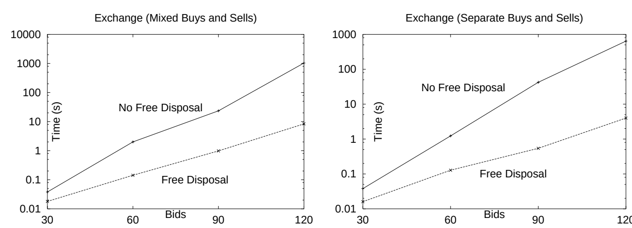

Figure 7: *Using CPLEX, optimal clearing of exchanges scales very poorly. We used a bids to items ratio of 10, and the values of the constants in the bid generation were $\alpha_1 = \alpha_2 = 0.8$. Left: each bid can buy some items while selling other items as the bid generation method, described above, does. Right: each bid either sells items or buys items, but does not do both. In both settings, exchanges with free disposal were easier to clear, but scaling was very poor even in that case.*

The clearing algorithms also keep track of the best solution found so far. This means that they implement the *anytime property*: the solution improves over time. In combinatorial auctions, the anytime property tends to work well in practice (although theoretically the inapproximability shows that no algorithm can guarantee a solution close to optimal in polynomial time). A good solution (within 99% of optimum) is often reached quickly [19]. However, as Figure 8 shows, the anytime property works very poorly in exchanges—even if there is free disposal. Solution quality tends to stagnate at around 85% of optimal. Wasting 15% of the economic value is clearly unacceptable in practice. Furthermore, without free disposal, it takes impractically long to find any feasible solution, not to mention a good one.

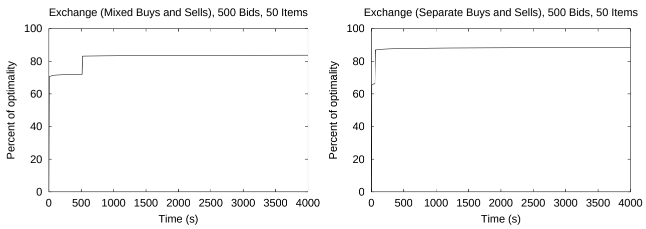

Figure 8: *Anytime performance of clearing combinatorial exchanges using CPLEX (50 items, 500 bids, $\alpha_1 = \alpha_2 = 0.8$). Left: each bid can buy some items while selling other items (the bids were generated using the method described above). Right: each bid either sells items or buys items, but does not do both. Each curve is for one problem instance. In both cases, CPLEX failed to find an optimal solution in 48 hours, at which time we terminated the runs.*

To summarize, the experiments show that combinatorial auctions and reverse auctions have reached the scalability that is needed in practice for many applications, but on combinatorial exchanges, even the fastest general-purpose commercial optimization package scales so poorly that it is far from being able to run combinatorial exchanges in practice. This suggests developing special-purpose algorithms for combinatorial exchanges.

4. CONCLUSIONS

We showed how different features of a combinatorial market affect the complexity of determining the winners. We studied auctions, reverse auction, and exchanges, with one or multiple units of each item, with and without free disposal. We theoretically analyzed the complexity of finding a feasible, approximate, or optimal solution.

The most interesting results were the following. Reverse auctions with free disposal can be approximated (even in the multi-unit case), although auctions cannot. When XOR-constraints between bids are allowed (to express substitutability), the hardness turns the other way around: even finding a feasible solution for a reverse auction or exchanges is $\mathcal{NP}$-complete, while in auctions that is trivial. Finally, in all of the cases without free disposal, even finding a feasible solution is $\mathcal{NP}$-complete.

We then studied the practical clearing time experimentally using a general-purpose mixed integer program solver on a variety of known benchmarks as well as ones which we introduced. As expected, cases with free disposal tended to be easier than ones without. On many distributions, reverse auctions with free disposal were easier than auctions with free disposal—as the approximability result would suggest—but interestingly, on one of the most realistic distributions they were harder. Single-unit exchanges were easy, but multi-unit exchanges were extremely hard. This suggests that faster, more specialized, algorithms are called for to scale winner determination to exchanges in practice.

5. REFERENCES

[1] A. Andersson, M. Tenhunen, and F. Ygge. Integer programming for combinatorial auction winner determination. *ICMAS*, p. 39–46, 2000.

[2] S. de Vries and R. Vohra. Combinatorial auctions: A survey. Draft, August 28th, 2000.

[3] Y. Fujishima, K. Leyton-Brown, and Y. Shoham. Taming the computational complexity of combinatorial auctions: Optimal and approximate approaches. *IJCAI*, p. 548–553, 1999.

[4] M. R. Garey and D. S. Johnson. *Computers and Intractability*. W. H. Freeman and Company, 1979.

[5] R. Gonen and D. Lehmann. Optimal solutions for multi-unit combinatorial auctions: Branch and bound heuristics. *ACM Ecommerce*, p. 13–20, 2000.

[6] J. Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. *Acta Mathematica*, 182:105–142, 1999.

[7] D. S. Hochbaum. *Approximation algorithms for NP-hard problems*. PWS Publishing Company, 1997.

[8] H. Hoos and C. Boutilier. Solving combinatorial auctions using stochastic local search. *AAAI-00*, 22-29.

[9] A. Kothari, T. Sandholm, and S. Suri. Solving combinatorial exchanges: Optimality via few partial bids. *AAAI workshop on AI for Business*, 2002.

[10] D. Lehmann, L. O'Callaghan, and Y. Shoham. Truth revelation in rapid, approximately efficient combinatorial auctions. *ACM Ecommerce*, p. 96-102, 1999.

[11] K. Leyton-Brown, M. Pearson, and Y. Shoham. Towards a universal test suite for combinatorial auction algorithms. *ACM Ecommerce*, p. 66–76, 2000.

[12] K. Leyton-Brown, M. Tennenholtz, and Y. Shoham. An algorithm for multi-unit combinatorial auctions. *AAAI*, 2000.

[13] N. Nisan. Bidding and allocation in combinatorial auctions. *ACM Ecommerce*, p. 1–12, 2000.

[14] M. H. Rothkopf, A. Pekeč, and R. M. Harstad. Computationally manageable combinatorial auctions. *Management Science*, 44(8):1131–1147, 1998.

[15] T. Sandholm. eMediator: A next generation electronic commerce server. *Computational Intelligence*, Special issue on Agent Technology for Electronic Commerce. To appear. Early versions: AGENTS-00, p. 73–96; AAAI-99 Workshop on AI in Electronic Commerce, p. 46–55; Washington Univ., WU-CS-99-02, Jan. 1999.

[16] T. Sandholm. Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence*, 135:1–54, Jan. 2002. Appeared as an invited talk at the Internat. Conf. on Information and Computation Economies, Charleston, SC, Oct. 25–28, 1998; Washington Univ., WUCS-99-01, Jan. 28th, 1999; IJCAI-99, p. 542–547.

[17] T. Sandholm and S. Suri. Improved algorithms for optimal winner determination in combinatorial auctions and generalizations. *AAAI-00*, p. 90–97.

[18] T. Sandholm and S. Suri. Side constraints and non-price attributes in markets. *IJCAI Workshop on Distributed Constraint Reasoning*, p. 55–61, 2001.

[19] T. Sandholm, S. Suri, A. Gilpin, and D. Levine. CABOB: A fast optimal algorithm for combinatorial auctions. *IJCAI*, p. 1102–1108, 2001.

[20] V. Vazirani. *Approximation Algorithms*. Springer, 01.