A Framework for Computing the Outcome of Proxied Combinatorial Auctions

Jie Zhong & Peter R. Wurman
North Carolina State University
Department of Computer Science
Raleigh, North Carolina
{jzhong, wurman}@ncsu.edu

Abstract

Proxy bidding has been proposed for combinatorial auctions as a means to speed up the auctions, to simplify the user interface, and to limit strategic behavior. The only previously known solution method for proxy bidding in combinatorial auctions requires the auctioneer to run the auction with myopic bidders to determine the outcome. In this paper we present a radically different approach that computes the bidders' allocation of their attention across the bundles only at the points at which they change their bidding patterns. This algorithm has several advantages over alternatives, including that it computes exact solutions and is invariant to the magnitude of the bids. We present a general framework and apply it to Ausubel and Milgrom's APA mechanism and our own Simple Combinatorial Proxy Auction. We present an example in which the approach is applied to a multi-stage proxy auction, and report on some preliminary computational results.

1 Introduction

Despite the recent advances in the theory of combinatorial auctions, we are only just beginning to see commercial applications. The vast majority of these applications are in the area of business-to-business procurement. Although held up as the gold standard by academics, the Generalized Vickrey Auction has been applied to very few real problems. Instead, practitioners tend to favor auctions which limit bid expressiveness or which engender price discovery through iteration.

Iterative versions of combinatorial auctions are attractive for several reasons, but particularly because they allow bidders to make bidding decisions with inexact value information and postpone the computation of exact values until it becomes clear which items are relevant to the final allocation. In cases where value determination is costly, this can save effort on the part of the bidders [2, 9, 10]. Consider, for example, the cost in time and money that a telecom company must expend in order to estimate the value of a broadcast license in a particular geographic market. Although attractive from a practical standpoint, iterative combinatorial auctions are difficult to analyze, in large part due to the tremendous size of the strategy space. The results that have been derived either assume a limited strategy space [2], or construct an iterative auction that arrives at the GVA prices through straightforward bidding, and by analogy, is incentive compatible [9].

Recently, mechanism designers at the FCC[1] and elsewhere have proposed incorporating *proxy bidding* into combinatorial auction designs. With proxy bidding, a bidder communicates a bid to a software agent that then follows a prescribed, incremental bidding policy until it either wins or exhausts the authority granted it by the bidder. In a combinatorial auction, the bidder sends a message to the proxy that expresses value for some or all of the bundles, where the values are typically more than are strictly necessary given the current state of the auction.

Intuitively, proxy bidding has several advantages. First, it speeds up the auction by allowing bidders to place larger bids which will be executed only to the extent necessary to outbid competitors [2, 9, 10]. Second, it may reduce the need for the bidders to accurately estimate the valuations of the other participants in the auction. For example, an equilibrium strategy in a first-price sealed-bid auction requires estimating the value of the second highest bidder. However, when a first-price, sealed-bid auction is enhanced with proxy bidding, as on eBay when snipers bid at the end of the auction, it reduces to a Vickrey auction and each bidder's equilibrium strategy is to submit her true value [11]. Third, proxies give mechanism designers a tool by which they can restrict the strategic flexibility of the bidders and thereby design successful auctions and predict their outcomes. Indeed, Ausubel and Milgrom's main result [2], in which they

[1] See the recent FCC Combinatorial Auction Conference at http:wireless.fcc.gov/auctions/default.htm?job=conference_papers&y=2003

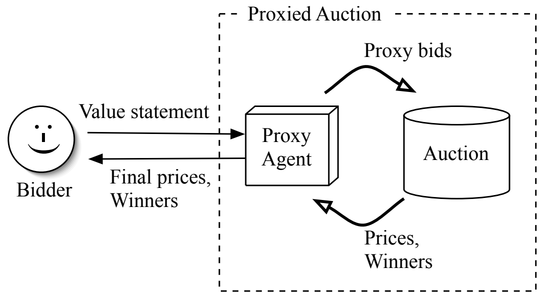

Figure 1. The interaction between the user and the proxy agent.

show that a semi-sincere equilibrium always exists in their Ascending Package Auction (APA), is in the context of a proxy agent following a simple bidding policy.

Figure 1 illustrates a proxy-enabled auction. The bidder gives a value statement to her proxy agent. The proxy agent places bids in the auction, which computes (possibly) prices and announces the current winners. Based on this feedback, and consistent with the auction rules and the agents' prescribed bidding policy, the proxy agents increment their bids. This cycle continues until the auction's termination conditions are met, at which point the system communicates the prices and winners to the bidders. Note that this entire process of bidders informing proxy agents and eventually hearing prices and winners can itself be iterative. That is, the humans, once they have seen the "final" prices and winners from the proxy round, can adjust their bids and begin another proxy round. In Section 5 we present an example with three proxy rounds.

Our focus in this paper is the problem circumscribed by the dotted box in Figure 1. Once the proxy-enabled auction has collected the value statements, it must compute the final prices and allocation entailed by the statements and the proxy bidding policy. We refer to this task as solving the *Proxy Auction Problem* (PAP). In contrast, the more commonly studied Winner Determination Problem (WDP) [1, 4, 12] is the problem faced by the auctioneer within the auction cylinder. It is important to note that, in general, the solution to a proxied combinatorial auction is not guaranteed to be the same as a sealed bid auction with the same initial bids. Nor, to our knowledge, is there a known optimization formulation that will always compute the result of straightforward bidding in a combinatorial auction. The closest result we know of is the demonstration by Bikhchandani [3] that APA is a primal-dual implementation that is guaranteed to find the optimal solution and the GVA prices when buyers are substitutes. For the reasons outlined above, we believe auctioneers will add proxy bidding to their combinatorial auctions, and will need an efficient method to solve PAP.

A natural method of solving PAP, which we refer to as *solving by simulation*, is to instantiate the proxy bidders as algorithms that, using the bidders' messages, incrementally bid until the winners and prices are determined. In contrast, our recent work [14], is based on an approach that more directly computes the outcome of the auction by computing the bidding patterns and the subsequent price trajectories engendered by the value statements. In our previous papers, we have outlined the framework and presented an application to the Simple Combinatorial Proxy Auction (SCPA). In this paper, we extend that work by presenting a solution to Ausubel and Milgrom's APA mechanism and simplifying the formulation for SCPA. In addition, we demonstrate how the algorithm can be used in multi-stage proxy auctions, and provide computational results comparing our approach to others.

2 The Framework

We refer to our approach as COMPROX, an acronym for "Combinatorial Proxy Framework". We present the formulism and the math programs needed to solve PAP with COMPROX, but we leave out the justifications due to a lack of space. The interested reader is referred to our earlier work [14] for a more detailed description.

Let $\mathcal{I}$ designate the set of m agents participating in the auction, and $\mathcal{J}$ the set of n objects being sold. Let $\mathcal{B}$ be the set of $2^n - 1$ combinations of n items, excluding the empty set. Denote agent i's valuation of bundle $b \in \mathcal{B}$ as $v_i(b)$. The vector v_i is the value statement supplied by the bidder to the agent and may not include a value for every possible bundle. Let $r_i^t(b)$ be agent i's bid on bundle b at time t, where $r_i^t(b) \in \mathcal{R}_+$, and we make the standard assumption that $r_i^t(\emptyset) = 0$. Let π_b^t denote the price associated with bundle b. Different auctions may compute π_b^t in different ways, or not at all. For APA, which does not announce prices, the price seen by i for b is $\pi_{b,i}^t = r_i^t(b)$.

We consider iterative combinatorial auctions with the standard myopic bidding function, sometimes called *straightforward bidding* [7]. If the agent is currently winning a bundle, then it does not increase its bid. If the agent is not currently winning, then it bids on the bundle or group of bundles that provide the most surplus given the current information, ϕ, revealed by the auctioneer. The function that determines on which bundle to bid is the *best response function*, $\mathrm{BR}_i(\phi)$. The output of the best response function, called the *demand set*, D_i, is the set of bundles that provide maximal surplus under the current information.

The process of solving by simulation is:

1. Announce ϕ.

2. Each agent that is not currently winning computes

$D_i = \mathrm{BR}_i(\phi)$ and bids on one or more elements of D_i in accordance with the auction rules. Typically this involves increasing the agent's bid by ϵ above the announcement.

3. Solve the WDP and compute ϕ.

4. If the termination conditions are met, exit. Otherwise, return to step 1.

Thus, to get accurate results, one has to set ϵ to be small, or to use methods that dynamically adjust ϵ. The latter method has been explored by Hoffman, et al. [5] and they have initial computational results for several methods of adjusting the size of the bid increment in order to reduce the number of iterations required to find the solution. Hoffman's algorithm is heuristic in nature, and has routines for rolling back bids if it determines that the bid increment was too large.

In contrast, ComProx is an iterative process in which each step involves computing the distribution of each bidder's *attention* among the bundles at particular points in time, and then computing the duration for which the former holds. Once we determine the time at which one of the agents changes its behavior, we move the clock forward and compute the new distributions of attention. This process repeats until all agents not winning a bundle drop out of the auction.

The details of an application of the framework vary depending on the rules of the auction, but they share the following skeleton:

1. From the current prices and agent valuations at time t, compute the distribution of attention by each agent.

2. Identify the set of competitive allocations, and the trajectories of all relevant allocations going forward.

3. Identify the trajectories of the prices of the relevant bundles going forward.

4. Compute the time, t', at which an agent will change its behavior.

5. If the termination condition is satisfied, exit. Otherwise, compute the prices at t' and return to step 1.

Rather than model incremental bidding decisions explicitly, ComProx constructs a model of the bid trajectories and keeps track of each agent's contribution to those trajectories. Changes in trajectories correspond to points in time at which agents change their bidding patterns. Attention, measured in monetary units per unit time, equates to the proportion of the agent's bidding opportunities it will use to increase its offer on a bundle. We denote agent i's attention to bundle b as $\theta_{i,b}$, and the proportion of time agent i spends passing as $\theta_{i,\mathrm{pass}}$. Each agent has one unit of attention per unit time to distribute among the bundles and the passing action.

A solution, f, is a feasible assignment of bundles to bidders, that is, $f : \mathcal{J} \to \mathcal{I}$. A solution is *active* if each member of the allocation has bid on the object it receives in f. Let f_i be agent i's allocation in f. Denote the set of all feasible allocations as F, and the subset of those which are active as F^{act}. Further, the algorithm identifies which of these are *competitive allocations* (CAs) at a given point in time, meaning that, given the current bids, these allocations generate the most value. Competitive allocations are simply the solutions to the WDP at an instant in time.

The value of allocation f at time t is

$$V(f) = \sum_i r_i^t(f_i). \tag{1}$$

Let F^* denote the set of allocations that maximize (1).

For auctions that track prices, the trajectory of a bundle's price is simply the sum of the attention being paid to the bundle. Let θ_b^t, the trajectory of the price of bundle b at time t, be computed as

$$\theta_b^t = \sum_i \theta_{i,b}^t.$$

Although the details of computing the distribution of attention is dependent on the particular auction being implemented, the general approach is to construct a mixed integer-linear program whose solution is the distribution of attention, the set of competitive allocations, and the price trajectories, if needed. We provide MILPs for both APA and the Simple Combinatorial Proxy Auction we developed in [14] in Sections 3 and 4, respectively.

The MILP program computes all of the information needed in Steps 1–3 of the framework. The other difficult step in the framework is the fourth, in which we determine the time at which the next inflection occurs. There are two potential events that define the end of an interval:

- the prices of bundles reach a point where one or more bidders change their demand sets, or
- an allocation that was not formerly competitive reaches a value that makes it competitive.

We refer the reader to [14] for a detailed discussion of the process for computing the duration of an interval.

3 ComProx Applied to APA

The application of ComProx to Ausubel and Milgrom's APA mechanism is less complex than the application to SCPA, and so we treat it first. APA is a simple mechanism that produces no prices. At each iteration, the auctioneer informs each agent whether it is currently winning. If currently winning, the agent does not bid. If not winning, the agent computes its demand set using the best response function

$$D_i^t = \arg\max_b [v_i(b) - r_i^t(b)].$$

The agent then submits revised bids for each element of D_i^t where $r_i^{t+1}(b) = r_i^t(b) + \epsilon$. Note that no items are removed from an agent's demand set until they all return zero surplus and the agent withdraws completely from the auction.

Because there are no prices, COMPROX needs only a relatively simple constraint program to compute the trajectories at each inflection point. The constraints capture the interactions of the competitive allocations as the bidders add bundles to their demand sets. At problem construction, we know the set of allocations that are solutions to the WDP at time t. We classify this set as the *potential* competitive allocations, $\hat{F} \subset F^*$, because, although they are instantaneously competitive, they may not remain competitive in the interval going forward.

In the following programs, the integer variable x_f takes the value one if allocation f will remain competitive during the interval, and zero otherwise. The variable β_f is the frequency with which the auctioneer selects solution f as the winning allocation. It is not true that all competitive allocations are selected equally often during an interval; instead, it depends in complex ways on the number of bidders actively bidding on each allocation. If allocation f is not competitive then $\beta_f = 0$.

In addition, we employ several constants that are bound during the construction of each MILP. $G_{f,i} = 1$ if i is allocated a bundle in f, and $G_{f,i} = 0$ otherwise. K_i is a constant that captures whether i has dropped out of the auction: $K_i = 0$ if D_i is empty, and $K_i = 1$ otherwise. N is a sufficiently large constant chosen as a multiple of the smallest expected value for attention.

The constraint program needed to compute the distribution of attention in APA is:

$$\begin{aligned}
\max \quad & 1 \\
\text{s.t.} \quad & \theta_{i,b} = \theta_{i,c}, \quad \forall i,\ b,c \in D_i && (2) \\
& \sum_{i \in \hat{f}} \theta_{i,\hat{f}_i} - \sum_{i \in f} \theta_{i,f_i} + N x_f \leq N, && (3) \\
& \qquad\qquad \forall f, \hat{f} \in \hat{F} \\
& K_i \sum_{f \in \hat{F}} G_{f,i}\beta_f + \theta_{i,b} = K_i, \quad \forall i, b \in D_i && (4) \\
& \sum_{f \in \hat{F}} \beta_f = 1, && (5) \\
& \beta_f \leq x_f, \quad \forall f \in \hat{F} && (6) \\
& \beta_f \geq 0, \quad x_f \in \{0,1\}, \quad \forall f \in \hat{F} \\
& 0 \leq \theta_{i,b} \leq 1, \quad \forall i;\ b \in D_i
\end{aligned}$$

A constant objective function is sufficient in this case because our goal is to find a solution that satisfies the constraints. Constraint (2) requires that the attention paid by the agent to two bundles in its demand set be equal. Constraint (3) requires that a bundle that remains in the competitive allocation set have a trajectory at least as great as any other allocation in the potentially competitive set. Constraint (4) states that if an agent is active, then the sum of its attention is one, and if it is inactive, the sum of its attention is zero. Constraint (5) simply requires that the frequency with allocations are chosen to be winning sums to one. Constraint (6) forces the frequency with which a solution can be selected to be winning to be zero if the solution does not remain competitive going forward. The rest of the constraints simply specify the domains of the variables.

The outcome of this program is an assignment of x_f for every $f \in \hat{F}$ that determines whether the allocation remains competitive, and if so, a non-zero value for β_f which specifies how frequently f is announced as the winner. It also determines how frequently each agent is a member of a winning allocation, and therefore how frequently the agent needs to increase its offers.

4 ComProx Applied to SCPA

We now examine the Simple Combinatorial Proxy Auction (SCPA) [14]. SCPA accepts bids on bundles and generates non-linear bundle-prices, that is, the price of a bundle may not be the sum of the prices of individual items. The auction proceeds in turns, and in each turn a buyer may place an offer on a new bundle or increment one of its previous offers. We denote buyer i's bid at turn t as a collection of mutually exclusive offers on bundles of the form $r_i^t(b)$. The auction remembers each bidder's last offer on each item. Several proposed iterative combinatorial auctions are quite similar to SCPA, including various versions of *i*Bundle [8, 9], and A1BA [13].

After the bids are received, the auction solves the WDP, computes and announces the new bundle-prices, and informs the bidders which bundles, if any, they are winning. The announced prices are simply the highest bid received on a particular item and are *anonymous* in that all bidders are given the same information. Thus,

$$\pi_b = \max_i r_i^t(b).$$

Note that the optimal allocation may include bidders whose last offer on their winning bundles are less than the current price; hence the need to directly inform bidders of their winning status.

The proxy agent submits the incremental bids on behalf of the user by following the straightforward bidding policy.

If it is told by the auction that it is winning, the agent does not increase its bid. Otherwise, the agent bids on bundle, b', that maximizes its real surplus at the given prices,

$$b' = \arg\max_{b} \{v_i(b) - (\pi_b + \delta)\}. \tag{7}$$

where δ is the minimum bid increment and $\boldsymbol{\pi}$ is the set of announced bundle-prices. If more than one bundle satisfies (7), the agent selects one randomly to bid on.

The introduction of prices significantly complicates the math program needed to solve the attention problem at the inflection points as compared to APA. The integer variable $y_{i,b}$ takes the value 1 if bundle b will remain in agent i's demand set during the following interval, and zero otherwise. This new variable is necessary because, unlike in APA, an agent can drop a bundle from its demand set if the price of the bundle increases faster than the price of other bundles in its demand set. Thus, the problem is constructed with all *potential* elements of each agent's demand set, and the solution to the MILP tells us which elements will be active in the next interval. The variable $\alpha_{i,b}$ is used to account for agent i's contribution to the value of competitive allocations. $\alpha_{i,b} = \theta_b$ when b is in i's active demand set, and $\alpha_{i,b} = 0$ otherwise. To simplify the notation, let $\alpha(f) = \sum_{i \in \hat{f}} \alpha_{i,\hat{f}_i}$.

The full math program is

$$\begin{aligned}
\max \quad & 1 \\
\text{s.t.} \quad & y_{i,b} \geq \theta_{i,b}, \quad \forall i,\ b \in \hat{D}_i && (8) \\
& y_{i,b} + N(\theta_b - \theta_c) - N^2 y_{i,c} \geq 1 - N^2, && (9) \\
& \qquad \forall i,\ b, c \in \hat{D}_i \\
& \alpha_{i,b} \leq N y_{i,b}, \quad \forall i,\ b \in \hat{D}_i && (10) \\
& \alpha_{i,b} - \theta_b \leq 0, \quad \forall i,\ b \in \hat{D}_i && (11) \\
& \theta_b - \alpha_{i,b} + N y_{i,b} \leq N, \quad \forall i,\ b \in \hat{D}_i && (12) \\
& \alpha(\hat{f}) - \alpha(f) + N x_f \leq N, \quad \forall f, \hat{f} \in \hat{F} && (13) \\
& \beta_f \leq x_f, \quad \forall f \in \hat{F} \\
& \sum_{f \in \hat{F}} \beta_f = 1 \\
& K_i \sum_{f \in \hat{F}} G_{f,i} \beta_f + \sum_{b \in \hat{D}_i} \theta_{i,b} = K_i, \quad \forall i \\
& \beta_f \geq 0, \quad x_f \in \{0,1\}, \quad \forall f \in \hat{F} \\
& \alpha_{i,b} \geq 0, \quad 0 \leq \theta_{i,b} \leq 1, \quad \forall i,\ b \in \hat{D}_i \\
& y_{i,b} \in \{0,1\}, \quad \forall i,\ b \in \hat{D}_i
\end{aligned}$$

We have numbered only the constraints that differ from those explained in Section 3. Constraint (8) requires that an agent give zero attention to any item that is not in its demand set going forward. Constraint (9) enforces the bidding behavior that requires that any bundle that is in an agent's active demand set have a slope no greater than any other bundle that was in the agent's potential demand set. Contraints (10–12) create the relationship between an agent's activity on a certain bundle and that bundle's contribution to the trajectory of the allocation that it is a part of. This constraint makes sure that an allocation does not get credit for the increase in the price of a bundle if the agent to whom it would be allocated is not actively bidding on it. Constraint (13) is a variation of Constraint (3) from the APA scenario, adapted for the introduction of the $\alpha_{i,b}$ term.

5 Multi-Stage Example

As mentioned in Section 1, the proxied auction may involve repeated interactions between the bidders and their proxy agents. To maintain the integrity of COMPROX in a multi-stage environment, we need only two things. First, buyers are required to submit bids that are improvements over their statements in previous rounds. Second, the prices announced at the end of round n are used as starting prices in round $n+1$.

Table 1 shows an SCPA scenario with four buyers and three items. In the following, the choices of statements made to the proxy agents were chosen for purposes of illustrating the algorithm and not to achieve strategic goals. The buyers are not required to express values for all bundles as they communicate with their respective proxy agents, nor are they required to reveal their true valuations for bundles. Suppose that going into the auction, Buyer 1 believes that she will end up with A, Buyer 2 expects to get B and one other item, Buyer 3 expects C, and Buyer 4 anticipates getting the combination ABC. Based on these expectations, the buyers reveal the information shown in Stage 1 of Table 2. COMPROX-SCPA computes the allocation {A, -, BC, -} with supporting prices {7.833, 0, 14, 7, 14, 14, 18}. Because Buyers 2 and 4 are not winning anything after the first stage, they communicate revised bids to their respective proxy agents, as shown in Stage 2 of Table 2. Based on the stage 2 bids, COMPROX-SCPA computes the allocation {A, BC, -, -} with supporting prices {8, 7, 16, 8, 16, 16, 20}. Now Buyers 3 and 4 are not winning. Buyer 4 can get no positive surplus and so drops out, but Buyer 3 choses to send the revised bids shown in Stage 3 to her proxy agent. Based on the Stage 3 bids, the auctioneer computes that both allocations {A, BC, -, -} and {A, B, C, -} produce the same value with supporting prices {8, 8, 16, 9, 16, 17, 25}. The trajectories of the bundle prices across all three stages are shown in Figure 2.

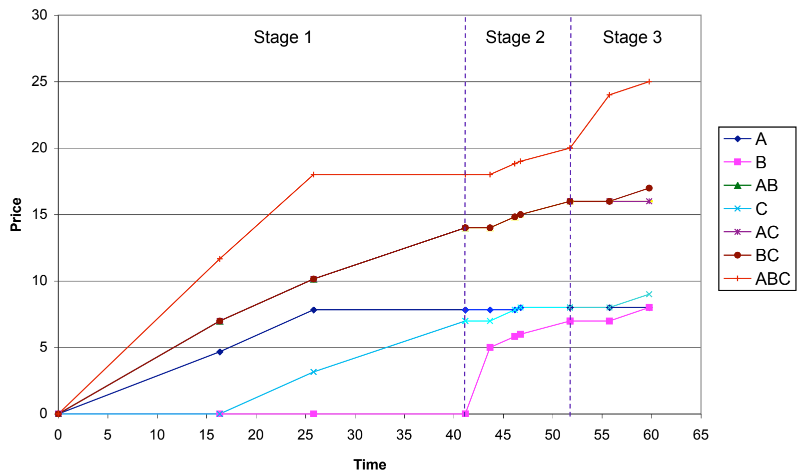

Figure 2. The prices over time in SCPA when buyers reveal the bids in Table 2 to their proxy agents.

	A	B	AB	C	AC	BC	ABC
Buyer 1	10	3	18	2	18	10	20
Buyer 2	4	9	15	3	12	18	20
Buyer 3	1	3	11	9	16	17	25
Buyer 4	7	7	16	7	16	16	20

Table 1. The valuations of four buyers bidding on combinations of three objects.

6 Computational Comparison

We have run preliminary studies to compare the computational costs of our algorithm versus simulation. The introduction of the constant objective function allows us to cast the COMPROX constraint programs as MILPs and solve them using CPLEX. We also used CPLEX to solve the WDP in the simulation approach.

We generated random problems in which the agents are assigned valuations on all bundles according to the following algorithm [13], parametrized by l and $\beta > 0$.

1. Assign values to individual items from the uniform distribution of integers between $[1, l]$.
2. Starting with bundles of size 2, and progressively increasing the bundle size,

	A	B	AB	C	AC	BC	ABC
	Stage 1						
Buyer 1	8	0	8	0	8	0	8
Buyer 2	0	0	14	0	0	14	14
Buyer 3	0	0	0	8	15	15	15
Buyer 4	0	0	0	0	0	0	18
	Stage 2						
Buyer 1	8	0	8	0	8	0	8
Buyer 2	0	9	15	0	0	18	20
Buyer 3	0	0	0	8	15	15	15
Buyer 4	0	7	16	0	16	16	20
	Stage 3						
Buyer 1	8	0	8	0	8	0	8
Buyer 2	0	9	15	0	0	18	20
Buyer 3	0	0	0	9	15	17	25
Buyer 4	0	7	16	0	16	16	20

Table 2. Example statements by buyers to their proxy agents in a three stage proxy auction.

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

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

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

We selected eleven problems sizes, and constructed ten problem instances at each size. The problems were all generated with parameter settings $l = 50$ and $\beta = 1.5$. We ran COMPROX and several instances of the simulation approach with different settings for the minimum bid increment. Figure 3 shows the average computation time over the ten instances at each problem size. The problems sizes are arrayed along the x-axis in an order that is roughly consistent with their difficulty. Note that our method of generating problems creates a value for every agent for every bundle. Thus, the largest problem size tested–thirteen agents and six items–represents 832 bids and 8192 possible allocations.

The graph shows that the exponential effects take over at much smaller problem sizes when the desired accuracy increases. Our approach has behavior similar to the simulation with bid increments of 0.1, which, given $l = 50$ represents an error on the order of 1%. Naturally, we haven't avoided the exponential explosion inherent in the problem. However, these results suggest that we can generate exact solutions that are independent of bid magnitude with similar computational costs.

7 Discussion and Future Work

The work on computing the results of proxy-enabled combinatorial auctions is only just beginning. The work by Hoffman et al. [5] is the only example that has come to our attention. Hoffman's approach is to take larger steps by using a large bid increment early in the auction, and then reduce the bid increment as the prices near the solution. A direct comparison between the computational costs of Hoffman's approach and ours is a subject for future research. We expect our method to involve fewer, but more costly, computations. Another way to speed up an iterative auction is to make use of results from previous solutions to the WDP. An example is the recent work by Kastner, et al. [6], in which the authors study the costs of maintaining previous solutions to the WDP as bidders incrementally increase their bids. While relevant to scenarios in which bidders have unconstrained bidding strategies, when proxy bidding is allowed, maintaining previous solutions to the WDP is unnecessary given the algorithm outlined in this paper.

We intend to continue with this line of research and to expand the framework to encompass the other auctions in the literature. We have begun applying the framework to A1BA [13], and conjecture that the SCPA framework can be used and A1BA's distinctive method of price determination can be applied *ex-post* or is not even necessary. We would also like to try to represent the anonymous price variations of *i*Bundle [8], and it may be possible to capture the discriminatory mechanisms.

We also plan to study the computational complexity of the process. We expect the MILP to be NP-complete, and at least as hard as the WDP. While we clearly benefit by solving for a limited number of inflection points, the value of the overall approach will depend upon how much harder it is to compute the allocation of attention than to solve the WDP. At the same time, we have made no effort yet to improve the tractability of the problem, and our experience so far suggests that there are many ways in which to reduce the computational overhead. For example, there are also obvious ways in which a computer program could avoid creating some instances of the constraints in the MILP. Because these optimizations obfuscate the central ideas in the approach, we left them out.

Acknowledgments

This research was funded in part by NSF CAREER award 0092591 and the E-Commerce program NC State. The views and conclusions contained herein are those of the authors, as are any errors. We thank Gangshu Cai and Ashish Sureka, who helped with the earlier version of this algorithm, David Parkes and Karla Hoffman for their encouragement and suggestions, and the attendees of the 2003 FCC Combinatorial Bidding Conference for their comments.

References

[1] A. Andersson, M. Tenhunen, and F. Ygge. Integer programming for combinatorial auction winner determination. In *Fourth International Conference on Multiagent Systems*, pages 39–46, 2000.

[2] L. M. Ausubel and P. R. Milgrom. Ascending auctions with package bidding. *Frontiers of Theoretical Economics*, 1(1):1–42, 2002.

[3] S. Bikhchanadani. Ascending price vickrey auctions. In *First World Congress of the Game Theory Society (Games 2000)*, Bilbao, Spain, July 2000.

[4] Y. Fujishima, K. Leyton-Brown, and Y. Shoham. Taming the computational complexity of combinatorial auctions: Optimal and approximate approaches. In *Sixteenth International Joint Conference on Artificial Intelligence*, pages 548–553, 1999.

[5] K. Hoffman, D. Menon, and T. Wilson. Observations and near-direct implementations of the ascending proxy auction. December 2003.

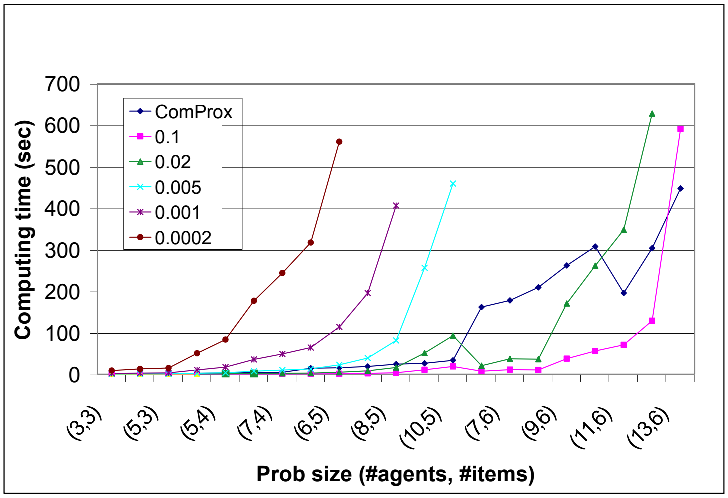

Figure 3. The comparison between COMPROX **and the simulation approach with different bid increments.**

[6] R. Kastner, C. Hsieh, M. Potkonjak, and M. Sarrafzadeh. On the sensitivity of incremental algorithms for combinatorial auctions. In *IEEE International Workshop on Advanced Issues of E-Commerce & Web-Based Information Systems (WECWIS)*, 2002.

[7] P. Milgrom. Putting auction theory to work: The simultaneous ascending auction. *Journal of Political Economy*, 108(2):245–272, 2000.

[8] D. C. Parkes. *i*Bundle: An efficient ascending price bundle auction. *First ACM Conference on Electronic Commerce*, pages 148–157, November 1999.

[9] D. C. Parkes and L. H. Ungar. Preventing strategic manipulation in iterative auctions: Proxy agents and price-adjustment. In *Seventeenth National Conference on Artificial Intelligence*, pages 82–89, 2000.

[10] J. Rodríguez, P. Noriega, C. Sierra, and J. Padget. 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)*, 1997.

[11] A. E. Roth and A. Ockenfels. Last-minute bidding and the rules for ending second-price auctions: Evidence from eBay and Amazon auctions on the Internet. *American Economic Review*, 92(4):1093–1103, June 2002.

[12] T. Sandholm, S. Suri, A. Gilpin, and D. Levine. Cabob: A fast optimal algorithm for combinatorial auctions. In *International Joint Conference on Artificial Intelligence*, Seattle, WA, 2001.

[13] P. R. Wurman and M. P. Wellman. AkBA: A progressive, anonymous-price combinatorial auction. In *Second ACM Conference on Electronic Commerce*, pages 21–29, 2000.

[14] P. R. Wurman, J. Zhong, and G. Cai. Computing price trajectories in combinatorial auctions with proxy bidding. *Electronic Commerce Research and Applications*, to appear.