Iterative Combinatorial Auctions: Theory and Practice

David C. Parkes and Lyle H. Ungar
Computer and Information Science Department
University of Pennsylvania
200 South 33rd Street, Philadelphia, PA 19104
dparkes@unagi.cis.upenn.edu; ungar@cis.upenn.edu

Abstract

Combinatorial auctions, which allow agents to bid directly for bundles of resources, are necessary for optimal auction-based solutions to resource allocation problems with agents that have non-additive values for resources, such as distributed scheduling and task assignment problems. We introduce *i*Bundle, the first iterative combinatorial auction that is optimal for a reasonable agent bidding strategy, in this case myopic best-response bidding. Its optimality is proved with a novel connection to primal-dual optimization theory. We demonstrate orders of magnitude performance improvements over the only other known optimal combinatorial auction, the Generalized Vickrey Auction.

Introduction

Auctions provide useful mechanisms for resource allocation problems with autonomous and self-interested agents. Typical applications include task assignment and distributed scheduling problems, and are characterized with distributed information about agents' local problems and multiple conflicting goals (Wellman 1993; Clearwater 1996). Auctions can minimize communication within a system, and generate optimal (or near-optimal) solutions that maximize the sum value over all agents.

More recently, electronic commerce has generated new interest in auction-based systems, both as dynamic mechanisms to sell items to individuals, and as systems for business-to-business transactions. Many retailers have online consumer auctions, e.g. `www.onsale.com`, and there are nascent auctions for procurement in the supply-chain, e.g. `www.freemarkets.com`. However, at present the vast majority of online auctions are simple variations on the traditional English auction, an ascending-price single-item auction.

We introduce *i*Bundle, an iterative combinatorial auction that allows agents to bid for bundles of items while the auctioneer increases prices and maintains a provisional allocation. Bundles are important in many real-world problems: consider a manufacturer that needs either components A and B, or just component C; consider a mobile agent that needs an interval of compute time; consider a train that needs a bundle of departure and arrival times on tracks across its route. Although combinatorial auctions can be approximated by multiple auctions on single items, this often results in inefficient outcomes (Bykowsky, Cull, & Ledyard 2000).

*i*Bundle is the first iterative combinatorial auction that is optimal for a reasonable agent bidding strategy, in this case myopic utility-maximizing agents that place *best-response* bids to prices. In this paper we prove the optimality of *i*Bundle with a novel connection to primal-dual optimization theory (Papadimitriou & Steiglitz 1982) that also suggests a useful methodology for the design and analysis of iterative auctions for other problems.

*i*Bundle has many computational advantages over the only other known optimal combinatorial auction, the Generalized Vickrey Auction (GVA) (Varian & MacKie-Mason 1995). As an iterative auction, agents can incrementally compute values for different bundles of items as prices change, and make new bids in response to bids from other agents. In comparison, the GVA is a *sealed-bid* auction, in which agents first submit bids simultaneously, and then the auctioneer determines an allocation and payments. In the GVA an agent's optimal strategy is to bid for, and compute the value of, *all* bundles for which it has positive value. This is often impossible, since for $|G|$ items there are $2^{|G|}$ bundles to value, each of which may require solving a difficult optimization problem (Parkes 1999a; Sandholm 1993).

However, combinatorial auctions introduce new computational complexities in mechanism execution. In particular, the auctioneer's *winner-determination* (WD) problem, the problem of choosing bids to maximize revenue, is $\mathcal{NP}$-hard by reduction from the maximal weighted clique problem (Rothkopf *et al.* 1998).

In *i*Bundle the auctioneer must solve a *sequence* of WD problems (one in each round) to maintain a provisional allocation as agents bid. In comparison, in the GVA the auctioneer must solve one WD problem for each agent in the final allocation. Each WD problem in *i*Bundle is smaller than in the GVA, because agents bid for less bundles. In addition, the auctioneer can increase the minimal bid increment and reduce the number of rounds to termination, reducing computation for some loss in economic efficiency. Further speed-ups are achieved through caching of solutions from previous rounds in the auction, and introducing approximate WD algorithms that maintain the same incentives for myopic agents to bid truthfully in each round.

We note that the GVA has stronger truth-revelation properties than *i*Bundle. Truthful bidding is optimal in the GVA whatever the bids of other agents. In comparison, rational agents with lookahead could manipulate the outcome of *i*Bundle to their advantage, and lead to suboptimal allocations. However, there is some evidence that myopic bidding may be a reasonable assumption in practice, perhaps because of the computational complexity of strategic behavior. For example, in the FCC broadband spectrum auction, conducted as a set of simultaneous ascending-price auctions on spectrum licenses, bids were rarely above minimum ask prices and jump bids were the exception (Cramton 1997).

In Parkes & Ungar (2000) we present a simple extension to *i*Bundle that makes it robust to strategic manipulation in several interesting problems; we adjust the final prices in *i*Bundle towards Vickrey prices.

The Ascending-Price Bundle Auction

*i*Bundle is an ascending-price auction that allows agents to bid on arbitrary combinations of items during the auction. The auctioneer increases prices on bundles as bids are received and maintains a set of winning bids that maximize revenue.

Let G denote the set of items to be auctioned, I denote the set of agents, and $S \subseteq G$ denote a bundle of items. The auction proceeds in rounds, indexed $t \geq 1$. We describe the types of bids that agents can place, and the allocations and price updates computed by the auctioneer.[1]

Bids. Agents can place exclusive-or bids for bundles, e.g. S_1 XOR S_2, to indicate than an agent wants either all items in S_1 or all items in S_2 but not both S_1 and S_2.[2]

Agent i associates a *bid price* $p^t_{\mathrm{bid},i}(S)$ with a bid for bundle S in round t, non-negative by definition. The price must either be within ϵ of, or greater than, the *ask price* announced by the auctioneer (see below). Parameter $\epsilon > 0$ defines the *minimal bid increment*, the minimal price increase in the auction. Agents must repeat bids for bundles in the current allocation, but can bid at the same price if the ask price has increased since the previous round.[3]

Winner-determination. The auctioneer solves a winner-determination problem in each round, computing an allocation of bundles to agents that maximizes revenue. The auctioneer must respect agents' XOR bid constraints, and cannot allocate any item to more than one agent. The provisional allocation becomes the final allocation when the auction terminates.

[1] The *i*Bundle auction has three variations, that differ in their price update rules (Parkes 1999b). In this paper, we use *i*Bundle both to refer to the family of auctions in general, and also to variation *i*Bundle(d), which we describe in detail.

[2] Exclusive-or bids provide complete expressibility, but are not necessarily computationally efficient for all problems. We can derive price-update rules for other bid languages (Parkes 1999b).

[3] An agent can also bid ϵ below the ask price for any bundle in any round— but then it cannot bid a higher price for that bundle in the future. This allows an agent to bid for a bundle priced slightly above its value.

Approximate Winner-determination. The auctioneer can also use an approximate algorithm for winner-determination, and still maintain the same incentives for myopic agents to follow the same bidding strategy. To achieve this an approximate algorithm must have the *bid-monotonicity* property:

Definition 1. *Bid monotonicity. An algorithm for winner-determination satisfies bid monotonicity if whenever an agent i is allocated a bundle with bids $\mathcal{B}_i$, it is also allocated a bundle with bids $\mathcal{B}_i \cup B$ that include a bid for an additional bundle B.*

It is straightforward to prove that *optimal* winner-determination algorithms are bid-monotonic.

Prices. The price-update rule generalizes the rule in the English auction, which is an ascending-price auction for a single item. In the English auction the price is increased whenever two or more agents bid for the item at the current price. In *i*Bundle the price on a bundle is increased when one or more agents that do not receive a bundle in the current allocation bid at (or above) the current ask price for a bundle. The price is increased to ϵ (the minimal bid increment) above the greatest failed bid price. The initial ask prices are zero.

The auctioneer announces a new ask price, $p^t_{\mathrm{ask}}(S)$ in round t, for all bundles S that increase in price. Other bundles are implicitly priced at least as high as the greatest price of any bundle they contain, i.e. $p_{\mathrm{ask}}(S') \geq p_{\mathrm{ask}}(S)$ for $S' \supseteq S$. These ask prices are anonymous, the same for all agents.

Price discrimination. In some problems the auctioneer introduces price discrimination based on agents' bids, with different ask prices to different agents, when this is necessary to achieve an optimal allocation. A simple rule dynamically introduces price discrimination on an agent-by-agent basis, when an agent submits bids that are **not** *safe*:

Definition 2. *Safe bids. An agent's bids are* ***safe*** *if the agent is allocated a bundle in the current allocation, or it does not bid at or above the ask price for any pair of* ***compatible*** *bundles S_1, S_2, such that $S_1 \cap S_2 = \emptyset$.*

Suppose agent i bids unsuccessfully for compatible bundles S_1 and S_2 in round t. It is still possible that bids for bundles S_1 and S_2 from two different agents can be successful at the prices. Remember that the XOR bid constraint prevents the auctioneer accepting both bids from agent i.

When an agent's bids are not safe the agent receives *individual ask prices*, $p_{\mathrm{ask},i}(S)$, in future rounds. Individual prices are initialized to the current general prices, $p^t_{\mathrm{ask},i}(S) = p^t_{\mathrm{ask}}(S)$, and increased to ϵ above the agent's bids in future rounds that the agent receives no bundle in the provisional allocation.

Termination. The auction terminates when: [T1] all agents submit the same bids in two consecutive rounds, or [T2] all agents that bid receive a bundle.

A Myopic Best-Response Bidding Strategy

*i*Bundle computes an optimal allocation with *myopically ra-*

tional agents that play a best (utility-maximizing) response to the current ask prices and allocation in the auction. The agents are myopic in the sense that they only consider the current round of the auction.

Let $v_i(S)$ denote agent i's value for bundle S, and assume $v_i(\emptyset) = 0$ and *free disposal* of items, so that $v_i(S') \geq v_i(S)$ for all $S' \supseteq S$. Consider a *risk-neutral* agent, with a quasi-linear utility function $u_i(S) = v_i(S) - p(S)$ for bundle S at price $p(S)$. Further, assume that agents are indifferent to within a utility of $\pm\epsilon$, the minimal bid increment. This is reasonable as $\epsilon \rightarrow 0$.

By definition, a myopic agent bids to maximize utility at the current ask prices (taking an ϵ discount when repeating a bid for a bundle in the provisional allocation or bidding for a bundle priced just above its value). The myopic best-response strategy is to submit an XOR bid for all bundles S that maximize (to within ϵ) utility $u_i(S)$ at the current prices. This maximizes the probability of a successful bid for bid-monotonic WD algorithms.

Theoretical Results

We are now ready to introduce our main theoretical results. Recall that $|G|$ is the number of items, $|I|$ is the number of agents, and ϵ is the minimal bid increment.

Theorem 1. i*Bundle terminates with an allocation that is within* $3\min\{|G|, |I|\}\epsilon$ *of the optimal solution, for myopic best-response agent bidding strategies.*

The auction is *optimal* as the bid increment approaches zero because the error-term goes to zero.

However, *i*Bundle requires price discrimination, and this can be hard to enforce. For example, the auctioneer must prevent agents entering the auction under multiple pseudonyms, and also prevent the transfer of items in an after-market. In a simpler variation, *i*Bundle(*2*), the auctioneer never tests for bid-safety and never price discriminates between agents (Parkes 1999b).[4] From Theorem 1, this variation is optimal at least when bids are **safe** (this condition is sufficient but not necessary):

Theorem 2. i*Bundle(2) terminates with an allocation that is within* $3\min\{|G|, |I|\}\epsilon$ *of the optimal solution when bids are safe, for myopic best-response agent bidding strategies.*

As an example, bids are safe if each agent bids for a set of *conflicting* bundles in every round of the auction. *i*Bundle(*2*) also provably solves the following problems without price discrimination: (1) every agent demands different bundles; (2) agents have additive or superadditive values, i.e. $v(S \cup S') \geq v(S) + v(S')$ for non-conflicting bundles S and S'; (3) the bundles that receive bids throughout the auction are from a single partition of items, e.g. all bids are for pairs of matching shoes, or single items.

In experimental tests *i*Bundle(*2*) performs well in many hard problems, achieving an average of 99% allocative efficiency[5] (Parkes 1999b) compared to 82% allocative efficiency from non-combinatorial auctions in the same problems. We found that price discrimination only had a noticeable effect on allocative efficiency with very small bid increments, and after many rounds of bidding.

Proof of Optimality

The proof of *i*Bundle's optimality is inspired by a proof due to Bertsekas (1990) for a simpler iterative auction, and makes an interesting connection with *primal-dual* theory of linear programming. It helps to motivate the price-update rules, the *safety condition* for introducing price discrimination, and the conditions for termination.

Primal-dual is an algorithm-design paradigm that is often used to solve combinatorial optimization problems (Papadimitriou & Steiglitz 1982). A problem is first formulated both as a *primal* and a *dual* linear program (see the examples below for *i*Bundle). A primal-dual algorithm searches for feasible primal and dual solutions that satisfy *complementary slackness conditions*, instead of searching for an optimal primal (or dual) solution directly. Complementary-slackness (CS for short) expresses logical relationships between the values of primal and dual solutions that are necessary and sufficient for optimality:

Complementary-Slackness Theorem. *Feasible primal and dual solutions are optimal if and only if they satisfy complementary slackness conditions.*

*i*Bundle implements a primal-dual algorithm for a linear-program formulation of the combinatorial resource allocation problem. It does this *without* formulating or solving the primal and dual problems explicitly, but based on information in agents' bids. Remember that the auctioneer does not know agents' values for resources.

Proof of *i*Bundle(*2*)

We first prove Theorem 2, the optimality of *i*Bundle(*2*) (the variation without price discrimination) in problems for which agents' bids are safe. A proof of Theorem 1, the optimality of *i*Bundle in general problems, follows from a simple transformation between *i*Bundle and *i*Bundle(*2*).

Figure 1 presents a standard integer program formulation of the combinatorial resource allocation problem. The objective is to maximize the total value of the allocation, given value $v_i(S)$ for bundle S to agent i. Integer variables $x_i(S) \in \{0, 1\}$ indicate whether or not agent i receives bundle S. Constraints (IP-1) ensure that each agent receives at most a single bundle, constraints (IP-2) ensure that each item is allocated to at most one agent.

Bikchandani & Ostroy (1998) formulate the combinatorial resource allocation problem as a *linear program*, see [LP$_2$] in Figure 2. The integer constraints $x_i(S) \in \{0, 1\}$ in [IP] are relaxed to $x_i(S) \geq 0$, and new variables $k \in K$ are introduced which correspond to a partition of items into bundles. K is the set of all possible partitions. Constraints

[4] Label *2* refers to "second-degree" price discrimination, non-linear prices in bundles of items but identical prices across agents (Bikchandani & Ostroy 1998).

[5] *Allocative efficiency* is a measure of optimality, computed as the ratio of the total value of the allocation across all agents to the value of the optimal allocation.

$$\max_{x_i(S)} \sum_{S \subseteq G} \sum_{i \in I} x_i(S) v_i(S) \qquad \text{[IP]}$$

$$\text{s.t.} \quad \sum_{S \subseteq G} x_i(S) \leq 1, \qquad \forall i \qquad \text{(IP-1)}$$

$$\sum_{S \subseteq G,\, j \in S} \sum_{i} x_i(S) \leq 1, \qquad \forall j \qquad \text{(IP-2)}$$

$$x_i(S) \in \{0, 1\}, \qquad \forall i, S$$

Figure 1: Combinatorial resource allocation problem: Integer program [IP] formulation.

$$\max_{x_i(S), y(k)} \sum_{S \subseteq G} \sum_{i \in I} x_i(S) v_i(S) \qquad [\text{LP}_2]$$

$$\text{s.t.} \quad \sum_{S \subseteq G} x_i(S) \leq 1, \qquad \forall i \qquad \text{(LP-1)}$$

$$\sum_{i \in I} x_i(S) \leq \sum_{k \in K, S \in k} y(k), \qquad \forall S \qquad \text{(LP-2)}$$

$$\sum_{k \in K} y(k) \leq 1 \qquad \text{(LP-3)}$$

$$x_i(S), y(k) \geq 0, \qquad \forall i, S, k$$

$$\min_{p(i), p(S), \pi} \sum_{i \in I} p(i) + \pi \qquad [\text{DLP}_2]$$

$$\text{s.t.} \quad p(i) + p(S) \geq v_i(S), \qquad \forall i, S \qquad \text{(DLP-1)}$$

$$\pi - \sum_{S \in k} p(S) \geq 0, \qquad \forall k \qquad \text{(DLP-2)}$$

$$p(i), p(S), \pi \geq 0, \qquad \forall i, S$$

Figure 2: Combinatorial resource allocation problem: Primal linear program [LP_2] and dual linear program [DLP_2] formulations.

(LP-2) and (LP-3) replace constraints (IP-2), and ensure that a feasible solution does not allocate more than one of each item.

In general, an optimal solution to the linear program [LP_2] can allocate fractional items to agents, and need not be a feasible solution to [IP]. In fact, the optimal solution to [LP_2] is integral and solves [IP] if and only if non-discriminatory bundle prices exist that support the optimal allocation in *competitive equilibrium* with best-response agent bidding strategies (Bikchandani & Ostroy 1998). Competitive equilibrium implies that agents' maximize utility and the auctioneer maximizes' revenue given the final prices and the final allocation.

The dual problem, [DLP_2], to primal [LP_2] is shown in Figure 2. Variables $p(i)$, $p(S)$ and π correspond to constraints (LP-1), (LP-2) and (LP-3) respectively, and dual constraints (DLP-1) and (DLP-2) correspond to primal variables $x_i(S)$ and $y(k)$. When the primal solution is integral the dual problem computes competitive equilibrium bundle prices that minimize the sum of agent utility and auctioneer revenue, see (1) and (2) below.

We prove that *i*Bundle(*2*) implements a primal-dual algorithm for [LP_2] and [DLP_2], and computes integral solutions to [LP_2] when agents follow myopic best-response bidding strategies and bids are safe. First, we show that the allocation and prices in each round of the auction correspond to feasible primal and dual solutions. Then, we show that the primal and dual solutions satisfy CS when the auction terminates.

Let S_i denote the provisional allocation to agent i, and $p_{\text{ask}}(S)$ denote the ask price for bundle S.

Feasible primal. To construct a feasible primal solution assign $x_i(S_i) = 1$ and $x_i(S') = 0$ for all $S' \neq S_i$. Partition $y(k^*) = 1$ for $k^* = [S_1, \ldots, S_{|I|}]$, and $y(k) = 0$ otherwise.

Feasible dual. To construct a feasible dual solution assign $p(S) = p_{\text{ask}}(S)$. Constraints (DLP-1) and (DLP-2) are satisfied with assignments:

$$p(i) = \max\left\{0, \max_{S \subseteq G}\{v_i(S) - p(S)\}\right\} \qquad (1)$$

$$\pi = \max_{k \in K} \sum_{S \in k} p(S) \qquad (2)$$

The value $p(i)$ can be interpreted as agent i's maximum utility at the prices, and π can be interpreted as the maximum revenue that the auctioneer can achieve at the prices (irrespective of the bids placed by agents). The auctioneer does not explicitly compute the value of $p(i)$, rather we prove that the allocation and prices in the auction satisfy CS with these assignments when the auction terminates, based on the bids placed by agents. This is just as well, because the values $v_i(S)$ remain private information to agents during the auction.

Complementary-slackness conditions. The first primal CS condition,[6] (CS-1) is:

$$x_i(S) > 0 \Rightarrow p(i) + p(S) = v_i(S), \qquad \forall i, S \qquad \text{(CS-1)}$$

Given (1) it states that all agents must only receive bundles that maximize utility at the current prices. (CS-1) is maintained throughout the auction because bundles are only allocated according to bids from agents, and agents place best-response bids. Formally, for any bundle S bid by agent i: (i) $p_{\text{ask}}(S) - \epsilon \leq p_{\text{bid},i}(S) \leq p_{\text{ask}}(S)$; (ii) $v_i(S) - p_{\text{bid},i}(S) + \epsilon \geq \max_{S'}\{v_i(S') - p_{\text{bid},i}(S')\}$ because agents bid for bundles that maximize utility within ϵ; (iii) $v_i(S) - p_{\text{bid},i}(S) \geq 0$, because agents only bid for bundles with positive utility. Since $x_i(S_i) = 1$ implies agent i bid for bundle S_i, we have:

$$x_i(S) > 0 \Rightarrow v_i(S) - p_{\text{ask}}(S) + 2\epsilon \geq \max\left\{0, \max_{S'}\{v_i(S') - p_{\text{ask}}(S')\}\right\}$$

[6]Complementary slackness states that if a primal variable is non-zero then its corresponding dual inequality constraint is binding. Similarly for dual variables.

Substituting for $p(i)$ and $p_{\rm ask}(S) = p(S)$, we prove ϵ-CS-1:

$$x_i(S) > 0 \;\Rightarrow p(i) + p(S) \le v_i(S) + 2\epsilon, \qquad \forall i, S \qquad (\epsilon\text{-CS-1})$$

The second primal CS condition, (CS-2), is:

$$y(k) > 0 \Rightarrow \pi - \sum_{S \in k} p(S) = 0, \qquad \forall k \qquad \text{(CS-2)}$$

Given (2) it states that the allocation must maximize the auctioneer's revenue at prices $p(S)$, over all possible allocations and irrespective of bids received from agents. We prove (CS-2) is maintained in all rounds because it is not binding that the auctioneer must allocate bundles according to agents' bids. Through the price-update rules the auctioneer is able to maximize revenue given prices in every round.

Formally: (i) Agent i with one of the highest losing bid for bundle S in round t will continue to bid for bundle S in rounds $t+1$. Let $u_i^t(S)$ denote agent i's utility for bundle S in round t. Then, $u_i^{t+1}(S) = u_i^t(S) - \epsilon$ because the ask price for S increases by ϵ. Also, $u_i^t(S) \ge u_i^t(S')$ for all bundles S' the agent did not bid in round t. Hence, with $u_i^t(S') \ge u_i^{t+1}(S')$ because the price of S' can only increase in round $t+1$, we have $u_i^{t+1}(S) \ge u_i^{t+1}(S') - \epsilon$ and a bid for S' can never exclude a bid for S from agent i's best-response bids in round $t+1$. A similar argument can be made for the utility of bundles that the agent *did* bid in round t; (ii) No single agent causes the price to increase to its current level on a pair of compatible bundles. This follows because price updates are due to safe bids from agents.

Therefore, for partition k^* such that $y(k^*) = 1$, $\sum_{S_i \in k^*} p_{\rm ask}(S_i) \ge \sum_{S_i \in k^*} p_{{\rm bid},i}(S_i)$, because $p_{\rm ask}(S) \ge p_{{\rm bid},i}(S)$, and $\sum_{S_i \in k^*} p_{{\rm bid},i}(S_i) \ge \max_{k \in K} \sum_{S_i \in k} p_{{\rm bid},i}(S_i)$ because of (i) and (ii), i.e. the constraints to allocate to agents' bids are not binding. Finally, with (2) we have $\max_{k \in K} \sum_{S_i \in k} p_{{\rm bid},i}(S_i) \ge \pi - \min\{|G|, |I|\}\epsilon$ because $p_{{\rm bid},i}(S) \ge p_{\rm ask}(S) - \epsilon$ and an allocation can include no more bundles than there are agents or items. We prove ϵ-CS-2:

$$y(k) > 0 \Rightarrow \pi - \sum_{S \in k} p(S) \le \min\{|G|, |I|\}\epsilon, \qquad \forall k \qquad (\epsilon\text{-CS-2})$$

The first dual CS condition, (CS-3), is:

$$p(i) > 0 \Rightarrow \sum_{S \subseteq G} x_i(S) = 1, \qquad \forall i \qquad \text{(CS-3)}$$

Given (1) it states that every agent with positive utility for some bundle at the current prices must receive a bundle in the allocation. (CS-3) is only satisfied during the auction for agents that receive bundles in the provisional allocation, but we prove (CS-3) for all agents when *i*Bundle(*2*) terminates. In termination case [T2] every agent that bids receives a bundle, so we immediately have (CS-3) with myopic best-response agents. In case [T1] some agents may bid and receive no bundles. However, these agents must bid at ϵ below the ask price and have values just below ask prices, otherwise prices would increase and their bids would change.

Finally, the last pair of dual CS conditions, (CS-4) and (CS-5), are:

$$p(S) > 0 \Rightarrow \sum_{i \in I} x_i(S) = \sum_{k \in K, S \in k} y(k), \qquad \forall S \qquad \text{(CS-4)}$$

$$\pi > 0 \Rightarrow \sum_{k \in K} y(k) = 1 \qquad \text{(CS-5)}$$

The assignment $y(k^*) = 1$ for the partition $k^* = [S_1 \ldots S_{|I|}]$ trivially satisfies the RHS of both conditions.

Termination. By contradiction, assume the auction never terminates. Informally, [T1] implies that agents must submit different bids in successive rounds, but with myopic best-response bidding this implies that prices must increase, and agents must eventually bid above their values for bundles. We prove a contradiction with myopic best-response bidding strategies.

Putting it all together. Summing ϵ-CS-1 over all agents in the final allocation, and with $p(i) = 0$ for agents not in the allocation by (CS-3), $\sum_{i \in I} p(i) \le \sum_{i \in I} v_i(S_i) - \sum_{i \in I} p(S_i) + 2\min\{|G|, |I|\}\epsilon$, because an allocation can include no more bundles than there are items or agents. Introducing ϵ-CS-2, because $y(k^*) = 1$ for the bundle-set that corresponds to the final allocation S_i, then $\pi \le \sum_{i \in I} p(S_i) + \min\{|G|, |I|\}\epsilon$. Finally, adding these two equations, we have $\pi + \sum_{i \in I} p(i) \le \sum_{i \in I} v_i(S_i) + 3\min\{|G|, |I|\}\epsilon$. The LHS is the value of the final dual solution, $V_{\rm DLP}$, and the first-term on the RHS is the value of the final primal solution, $V_{\rm LP}$. We know $V^*_{\rm LP} \le V_{\rm DLP}$, where $V^*_{\rm LP}$ is the value of the optimal primal solution by the weak duality property of linear programs. Thus, because $V_{\rm DLP} \le V_{\rm LP} + 3\min\{|G|, |I|\}\epsilon$, it follows that $V_{\rm LP} \ge V^*_{\rm LP} - 3\min\{|G|, |I|\}\epsilon$. Finally, because the primal solution is integral (by construction during *i*Bundle), it is a feasible and optimal solution to the combinatorial resource allocation problem [IP]. □

In addition, it follows immediately that *i*Bundle(*2*) terminates in competitive equilibrium when agents are myopically rational and place safe bids.

Proof of *i*Bundle

A simple transformation of agents' bids allows *i*Bundle to be implemented within *i*Bundle(*2*) and ensures that agents' bids remain safe throughout the auction. Whenever bids from agent i are not safe in *i*Bundle(*2*) we can simulate the price-update rule in *i*Bundle by introducing a new dummy item that is specific to that agent, call it X_i. This item is concatenated by the auctioneer to all bids from agent i in this round and all future rounds. It has the following effects:

1. The outcome of winner-determination, or the allocative efficiency of the auction, is unchanged because no other agent bids for item X_i.
2. Agent i's bids are always safe because every bid includes item X_i, and no pair of bids is compatible.

3. The price increases due to bids from agent i are isolated to that agent in all future rounds because all price increases are for bundles that include item X_i.

The optimality of *i*Bundle follows immediately from the optimality of *i*Bundle(2).

Computational Analysis

As an iterative auction, *i*Bundle has many computational advantages for *agents* over the sealed-bid GVA, as we discussed in the introduction. In Parkes (1999b) we present results that demonstrate savings in agent valuation work in *i*Bundle.

However, the winner-determination (WD) problem that the auctioneer solves in each round of *i*Bundle to compute the provisional allocation is $\mathcal{NP}$-hard, just as in the GVA. The auctioneer must solve one WD problem in each round, and a naive worst-case analysis gives $O(BV_{\max}/\epsilon)$ rounds to converge, for a total of B bundles with positive value over all agents, maximum value $V_{\max}$ for any bundle, and minimal bid increment ϵ. In the worst-case the price of a single bundle must increase by at least ϵ in each round the auction remains open, and prices are bounded by the maximum value over all agents. The number of rounds to termination is inversely proportional to the minimal bid increment. The auctioneer can solve less WD problems by increasing the minimal bid increment, for some loss in economic efficiency.

A number of optimizations are possible within *i*Bundle to speed-up computation on winner-determination in each round. First, the provisional allocation from the previous round provides a good initial solution to the WD problem, because agents must re-bid bundles received in the previous round. This allows pruning of the search for a revenue-maximizing allocation. An additional saving in computation time is achieved by limiting search to an allocation at least ϵ better than the value of the allocation in the previous round. Moreover, although each intermediate WD problem in *i*Bundle may be intrinsically more difficult than each WD problem in GVA because all agents bid at similar prices for bundles (Andersson *et al.* 2000), the problems are typically much smaller than in the GVA.

The auctioneer only announces price *increases* in each round, and need not maintain explicit prices for all possible bundles. Bid prices are verified dynamically in each round, to check that bids are at least as large as the ask price of all contained bundles. With a simple sorted-list implementation, the total work in checking each bid is *linear* in the number P of bundles that have explicit ask prices. Similarly, prices can be maintained in linear-time in P for each new price increase. In addition, $P \leq B$, with agents that have values for B bundles, because only bundles that receive bids can receive explicit ask prices.

Experimental Results

We compare the computation and communication cost of *i*Bundle with the Generalized Vickrey Auction (GVA).

We consider problems *Decay*, *Weighted-random* (WR), *Random* and *Uniform* from Sandholm (1999). Each problem defines a distribution over agents' values for bundles of items, with XOR valuation functions, such that agents want at most one bundle. In our main experiments the number of items, $|G| = 50$, and we scale the problems by increasing the number of agents from 5 to 40, with values for 10 bundles per agent. We set Sandholm's parameter $\alpha = 0.85$ in Decay, and select bundles of size 10 in Uniform.

Results are presented for *i*Bundle(*2*), the auction variation without price discrimination. A variation on Sandholm's depth-first branch-and-bound search algorithm (Sandholm 1999) solves winner-determination (WD) in each round, and computes the allocation and prices in the GVA. We introduce a new heuristic to make search more efficient for XOR bids. The heuristic computes an overestimate of the possible value of a partial allocation based on allocating at most *one* bundle to each remaining agent without a bundle.

In addition, we measure the performance of *i*Bundle with a greedy approximate winner-determination algorithm due to Lehmann *et al.* (1999) that satisfies the bid-monotonicity property (Definition 1).

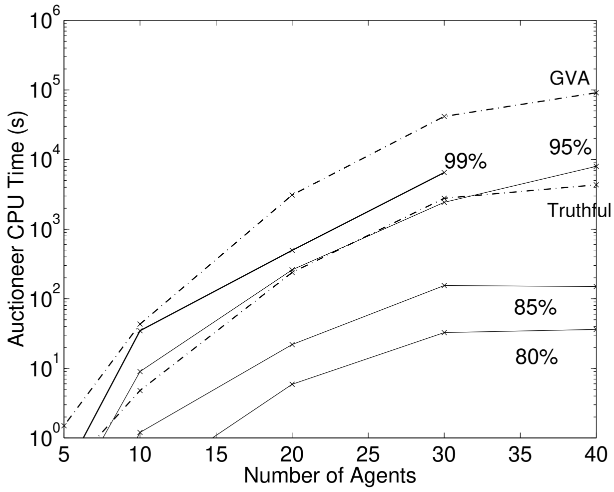

Figure 3: Total computation time in *i*Bundle(*2*), the GVA, and a sealed-bid auction with truthful agents, in problem set Decay. The performance of *i*Bundle is plotted with different bid increments ϵ, selected to give allocative efficiency of 80%, 85%, 95% and 99%.

Figure 3 plots the total auctioneer winner-determination and price-update time[7] in *i*Bundle in the Decay problem set. Performance is measured for different bid increments, with the bid increment selected to give allocative efficiency of 80%, 85%, 95% and 99% ($\pm 1\%$). Figure 3 also plots performance for the GVA, and for a sealed-bid auction in which agents are assumed to bid truthfully.[8] Results are averaged over 10 trials. First, note that the curves are sublinear on the logarithmic value axis as the number of agents increases,

[7] Time is measured as user time in seconds on a 450 MHz Pentium Pro with 1024 MRAM, with *i*Bundle coded in C++.

[8] The GVA proved intractable for 30 and 40 agents. In those problems the run time is estimated as the time to compute the optimal solution in a single WD problem multiplied by the number of agents in the optimal allocation.

indicating polynomial computation time in the number of agents.

The performance improvement of *i*Bundle over GVA is striking, achieving at least one order of magnitude improvement with 99% allocative efficiency and three orders of magnitude with 85% allocative efficiency. For up to 95% efficiency we essentially get the myopic truth-revelation properties of *i*Bundle for free, because *i*Bundle's run-time is approximately the same as for the sealed-bid auction with truthful agents.

Problem		GVA	*i*Bundle			Approx-Bundle
			$\simeq 90\%$	$\simeq 95\%$	$\simeq 99\%$	
Decay	Eff (%)	100	91.5	94.9	98.3	85.1
67.3%[d]	WD-time[a] (s)	41700	831	2400	5650	0
13.4[e]	Pr-time[b] (s)	–	26	34.5	44	39.2
	Comm[c] (kBit)	18.8	221	306	394	377
WR	Eff (%)	100	90.7	94.9	99.2	79.4
71.5%	WD-time (s)	3	0.6	1.7	6	0
1	Pr-time (s)	–	5.4	11.5	40.9	12.2
	Comm (kBit)	18.1	20.5	52.1	144	53.1
Rand	Eff (%)	100	89.3	97	99	95.8
37.8%	WD-time (s)	68	4.4	7.4	11	0
11.2	Pr-time (s)	–	6.5	9.7	12.1	12.9
	Comm (kBit)	18.7	49.5	66.4	82.6	85.6
Unif	Eff (%)	100	–	95.6	99.1	76.2
58%	WD-time (s)	25	–	6.6	18.7	0
3	Pr-time (s)	–	–	14.7	42.0	46
	Comm (kBit)	18.2	–	56.5	120	124

Table 1: Performance in the Decay, WR, random, and uniform problems. [a] Auctioneer WD time. [b] Price-update time. [c] Communication cost. [d] Alloc. eff. of a sealed-bid auction with a greedy WD algorithm and truthful agents. [e] Average number of agents in the optimal solution.

Table 1 compares *i*Bundle with the GVA for all Sandholm's problems, for problems with 30 agents. With our parameters the WR and Uniform problems are quite easy because the optimal allocation sells large bundles to a few agents, which allows considerable pruning during search. The Random and, in particular, Decay problems tend to be harder because the optimal allocation requires coordination across a number of agents, see also Sandholm (1999) and Andersson *et al.* (2000). In all problems *i*Bundle has less WD time at 95% allocative efficiency than the GVA. Note that price-update is relatively expensive in the otherwise easy weighted-random (WR) problem, because bid prices for large bundles must be checked for price consistency against the price of all included bundles.

There is a communication cost[9] penalty in using *i*Bundle compared with the GVA in these problems (Table 1) because of repeated bids across a number of rounds. This would change in problems with agents that have values for many bundles because all values must be reported in the GVA, or in easier problems because *i*Bundle will terminate quickly with less bids.

The performance of *i*Bundle with the greedy WD algorithm is noteworthy: *i*Bundle performs well in the hard Decay problem set, with allocative efficiency 85.1%, giving at least a 1000-fold reduction in WD time. We believe that other, slightly less greedy, approximate algorithms will give even further performance improvements.

Speeding up *i*Bundle In addition to using the allocation from the previous round to prune search, it is also useful to cache all previous provisional allocations and select the best cached allocation as an initial solution for WD. A simple linear program is used to select the best allocation from the cache, and requires negligible computation. In our main trials we use a cache size of 1, i.e. take the solution from the previous round as an initial solution to the WD problem.

Problem		WD Time				% Cache Correct			
		0	1	T	$T!$				
Decay	50/15/150[a]	415	371	355	291	0	28	47	59
WR	50/50/1000	253	243	231	163	0	11	57	57
Rand	50/30/600	1823	1616	1491	864*	0	6	30	78
Unif	50/40/800	343	337	336	110	0	14	29	49

Table 2: Winner-determination time with caches of size 0, 1 (last round), and T (all previous rounds). In cache $T!$ revenue maximizing cached solutions from previous rounds are assumed optimal. *Eff* $> 99\%$ in all problems except *, where *Eff* $= 96.8\%$. [a] $|G|$ / $|I|$ / # bundle values.

Table 2 compares the WD time in each problem with and without caching of previous allocations. Although a full cache can provide an additional speed-up over using no cached solutions, or just the allocation from the previous round, the effect is not very dramatic. The reason is that it remains expensive to *verify* that a cached solution is optimal. For example, although an extended cache in the Decay problem provides the correct allocation in 47% of problems, the speed-up is limited to around 14%.

In an attempt to leverage the correct solutions from the cache, we tested the performance of *i*Bundle under an additional assumption that if a cached solution from before round $t-1$ generates more revenue than the solution from round $t-1$, this is adopted as the new provisional allocation without further computation. The rule is designed to capture "flip-flop" competition between a number of good allocations during an auction.

Labeled $T!$, the rule proves useful in Decay, WR and Uniform, reducing computation by 30%, 36% and 68% from the time with no cache for a negligible drop in allocative efficiency. However, one must be careful: although we also see a speed-up in Random, the allocative efficiency falls from 99% to 96.8%. Further analysis shows that cached solutions prove optimal in 54%, 97% and 49% of rounds in Decay, WR and Uniform, but only optimal 34% of rounds in Random.

Further optimizations should be possible, for example

[9] We assume that bids and price information in *i*Bundle must only specify a bundle, because bids are usually at the current ask price, and ask prices only increase by the minimal bid increment. We also assume a broadcast network infrastructure for price updates. A bundle is specified with $|G|$ bits. In the GVA a bid specifies both a bundle and a value. We assume that values require 10 bits, enough to specify a value to 3 significant figures ($\log_2(1000) \simeq 10$.)

using cached solutions once a large enough cache is constructed, and solving WD when an auction is about to terminate with cached solutions. Another useful approach is ϵ-scaling, that adjusts the bid increment during an auction (Bertsekas 1990).

Related Work

Rassenti *et al.* (1982) describe an early single-round combinatorial mechanism for airport slot allocation, while Banks *et al.* (1989) describe AUSM, an early iterative bundle auction. AUSM has no explicit price-update rules, and agents must solve hard problems to bid effectively. DeMartini *et al.* (1998) describe, RAD, an iterative extension of Rassenti *et al.*, also with linear prices. No optimal properties have been proved for any of these auctions in general problems.

The A*k*BA (Wurman 1999, chapter 5) auctions are conceptually similar to *i*Bundle, but have different price-update rules and no price discrimination. A*k*BA shares many of *i*Bundle's computational properties, but is not known to be optimal for any reasonable bidding strategy.

There have been a number of proposals to reduce the computational costs of combinatorial auctions while maintaining incentives for truth-revelation; e.g. limit the types of bundles that agents can bid for (Rothkopf *et al.* 1998); or introduce an approximate solution for winner-determination (Lehmann *et al.* 1999), but little success in designing good auctions for general bundle problems. Moreover, most previous work focuses on sealed-bid auctions.

Conclusions

*i*Bundle is a new iterative combinatorial auction that is optimal for myopically-rational agents. As an iterative auction, *i*Bundle is particularly useful when agents have hard local valuation problems because it allows agents to compute estimates of the value of different outcomes incrementally, in response to bids from other agents. We proved *i*Bundle's optimality within a primal-dual framework, which we believe will provide a useful conceptual basis for the design and analysis of iterative auctions for other problems.

It remains expensive to compute *optimal* solutions with *i*Bundle in many problems, because the auctioneer's winner-determination (WD) problem is $\mathcal{N}P$-hard. We suggested a number of techniques to reduce computation, possibly for some loss in allocative efficiency, for example: increase the bid increment, use cached allocations, and introduce approximate winner-determination algorithms. We demonstrated orders of magnitude performance improvements over the GVA, the only other known optimal combinatorial auction, in some hard problems.

In future work we plan to test *i*Bundle in some real problems, and experiment with additional bid restrictions and alternative approximate WD algorithms. An interesting open problem is to adapt *i*Bundle for two-sided markets, with multiple buyers and sellers.

Acknowledgments

This research was funded in part by National Science Foundation Grant SBR 97-08965.

References

Andersson, A.; Tenhunen, M.; and Ygge, F. 2000. Integer programming for auctions with bids for combinations. *Forthcoming, Proc. ICMAS'00.*

Banks, J. S.; Ledyard, J. O.; and Porter, D. 1989. Allocating uncertain and unresponsive resources: An experimental approach. *The Rand Journal of Economics* 20:1–25.

Bertsekas, D. P. 1990. The auction algorithm for assignment and other network flow problems: A tutorial. *Interfaces* 20(4):133–149.

Bikchandani, S., and Ostroy, J. M. 1998. The package assignment model. Technical report, Anderson School of Management and Department of Economics, UCLA.

Bykowsky, M. M.; Cull, R. J.; and Ledyard, J. O. 2000. Mutually destructive bidding: The FCC auction design problem. *Journal of Regulatory Economics*.

Clearwater, S. H., ed. 1996. *Market-Based Control*. World Scientific.

Cramton, P. 1997. The FCC spectrum auctions: An early assessment. *J. Economics and Management Strategy* 6:431–495.

DeMartini, C.; Kwasnica, A. M.; Ledyard, J. O.; and Porter, D. 1998. A new and improved design for multi-object iterative auctions. Technical Report SSWP 1054, California Institute of Technology. Revised March 1999.

Lehmann, D.; O'Callaghan, L.; and Shoham, Y. 1999. Truth revelation in rapid, approximately efficient combinatorial auctions. In *Proc. ACM Conf. on Electronic Commerce (EC-99)*.

Papadimitriou, C. H., and Steiglitz, K. 1982. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall.

Parkes, D. C., and Ungar, L. H. 2000. Preventing strategic manipulation in iterative auctions: Proxy agents and price-adjustment. In *Proc. 17th National Conference on Artificial Intelligence (AAAI-00)*. To appear.

Parkes, D. C. 1999a. Optimal auction design for agents with hard valuation problems. In *Proc. IJCAI-99 Workshop on Agent Mediated Electronic Commerce*. Stockholm.

Parkes, D. C. 1999b. *i*Bundle: An efficient ascending price bundle auction. In *Proc. ACM Conf. on Electronic Commerce (EC-99)*, 148–157.

Rassenti, S. J.; Smith, V. L.; and Bulfin, R. L. 1982. A combinatorial mechanism for airport time slot allocation. *Bell Journal of Economics* 13:402–417.

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

Sandholm, T. 1993. An implementation of the Contract Net Protocol based on marginal-cost calculations. In *Proc. 11th National Conference on Artificial Intelligence (AAAI-93)*, 256–262.

Sandholm, T. 1999. An algorithm for optimal winner determination in combinatorial auctions. In *Proc. 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, 542–547.

Varian, H., and MacKie-Mason, J. K. 1995. Generalized Vickrey auctions. Technical report, University of Michigan.

Wellman, M. P. 1993. A market-oriented programming environment and its application to distributed multicommodity flow problems. *Journal of Artificial Intelligence Research* 1:1–23.

Wurman, P. R. 1999. *Market Structure and Multidimensional Auction Design for Computational Economies*. Ph.D. Dissertation, University of Michigan.