Truthful Approximation Mechanisms for Restricted Combinatorial Auctions

Ahuva Mu'alem Noam Nisan

January 21, 2002

Abstract

When attempting to design a truthful mechanism for a computationally hard problem such as combinatorial auctions, one is faced with the problem that most efficiently computable heuristics can not be embedded in any truthful mechanism (e.g. VCG payment rules will not ensure truthfulness).

We develop a set of techniques that allow constructing efficiently computable truthful mechanisms for combinatorial auctions in the special case where only the valuation is unknown by the mechanism (the single parameter case). For this case we extend the work of Lehmann O'Callaghan, and Shoham, who presented greedy heuristics, and show how to use IF-THEN-ELSE constructs, perform a partial search, and use the LP relaxation. We apply these techniques for several types of combinatorial auctions, obtaining truthful mechanisms with provable approximation ratios.

1 Introduction

Recent years have seen a surge of interest in combinatorial (also called combinational) auctions, in which a number of non-identical items are sold concurrently and bidders express preferences about combinations of items and not just about single items. Such combinatorial auctions have been suggested for a host of auction situations such as those for spectrum licenses, pollution permits, landing slots, computational resources, online procurement and others. See [19] for a survey.

Beyond their direct applications, combinatorial auctions are emerging as the central representative problem for a whole new field of research that is sometimes called algorithmic mechanism design. This field deals with the interplay of algorithmic considerations and game-theoretic considerations that stem from computing systems that involve participants (players, agents) with differing goals. Many leading examples are motivated by Internet applications, e.g., various networking protocols, electronic commerce, and non-cooperative software agents. See e.g. [15] for an early treatment, and [10, 14] for more recent surveys. The combinatorial auction problem is attaining this central status due to two elements: First, the problem is very expressive (e.g. a competition for network resources needed for routing can be modeled as a combinatorial auction of bandwidth on the various communication links). Second, dealing with combinatorial auctions requires treating a very wide spectrum of issues.

Indeed implementation of combinatorial auctions faces many challenges ranging from purely representational questions of succinctly specifying the various bids, to purely algorithmic challenges of efficiently solving the resulting, NP-hard, allocation problems, to pure game-theoretic questions of bidders' strategies and equilibria. Much work has recently been done on these topics, see e.g., [19, 16, 11, 7] and many references therein.

Perhaps the most interesting questions are those that intimately combine computational considerations and game theoretic ones. Possibly the most central problem of this form is the difficulty of getting algorithmically efficient truthful mechanisms. The basic game-theoretic requirement in mechanism design is that of "truthfulness" (incentive compatibility), i.e. that the participating agents are motivated to cooperate with the protocol [1]. The basic algorithmic requirement is computational efficiency. Each of these requirements can be addressed separately: "VCG mechanisms" [18, 3, 4] – the basic possibility result of mechanism design – ensure truthfulness, and a host of algorithmic techniques (e.g. [16, 19, 20]) can achieve reasonably good allocations for most practical purposes (despite the general NP-hardness of the allocation problem). Unfortunately, these two requirements do conflict with each other! It has been noticed [7, 12] that when VCG mechanisms are applied to non-optimal allocation algorithms (as any computationally efficient algorithm must be), truthfulness is not obtained. This problem was studied further in [13, 9].

The key positive result known so far is due to [7]. They restrict the set of preferences that agents may have to be what they call "single minded", i.e. agents that are only interested in a single bundle of items. For this class of bidders they present a family of simple greedy mechanisms that are both algorithmically efficient and truthful! They also show that one mechanism in this family has provable approximation properties. In this paper we continue this line of research. We slightly further restrict the agents, but we obtain a much richer class of algorithmically efficient truthful mechanisms. In fact we present a set of general tools that allow the creation of such mechanisms. Many of our results, but not all of them, apply also to the general class of single-minded bidders of [7].

In our model, termed "known single minded bidders", we not only assume that each agent is only interested in a single bundle of goods, but also that the identity of this bundle can be verified by the mechanism (in fact it suffices that the cardinality of the bundle can be verified). This assumption is reasonable in a wide variety of situations where the required set of goods can be inferred from context, e.g. messages that needs to be routed over a set of network links, or bundles of a given cardinality. Our model captures the general case where only a single parameter (a one-dimensional valuation) is unknown to the mechanism and must be handled in a truthful way. (The single parameter case has also been studied from a computational point of view in a different context in [1].) We first present an array of general algorithmic techniques that can be used to obtain truthful algorithms:

- Generalizations of the greedy family of algorithms suggested by [7].
- A technique based on linear programming.
- Finitely bounded exhaustive search.
- A "MAX" construct: this construction combines different truthful algorithms and takes the best solution.
- An If-Then-Else construct: this construction allows branching, according to a condition, to one of many truthful algorithms.

The combination of these techniques provides enough flexibility to allow construction of many types of truthful algorithms. In particular it allows many types of partial search algorithms –

[1] We defer the exact game theoretic definitions to section 2. In general one only needs equilibria, but the revelation principle allows concentration on truthful mechanisms.

the basic heuristic approach in many applications. We demonstrate the generality and power of our techniques by constructing polynomial-time truthful algorithms for several important cases of combinatorial auctions for which we prove approximation guarantees:

- An $\epsilon\sqrt{m}$-approximation for the general case for any fixed $\epsilon > 0$. This improves over the $\sqrt{m}$ ratio proved in [7], where m is number of items. This is, in fact, the best algorithm (due to [5]) known for combinatorial auctions even without requiring truthfulness!
- A very simple 2-approximation for the homogeneous (multi-unit) case. Despite the extensive literature on multi unit auctions (starting with the seminal paper [18]) this is the first known truthful mechanism with bids that are not downward sloping!
- An $m+1$-approximation for multi-unit combinatorial auctions with m types of goods.

The rest of this paper is structured as follows. In section 2 we formally present our model and notations. In section 3 we also provide a simple algorithmic characterization of truthful mechanisms. In section 4 we present our basic techniques and prove their correctness, and in section 5 we present our operators for combining truthful mechanisms. In section 6 we present our applications and prove their approximation properties. Finally, in section 7, we shortly mention which of our results generalize to the single-minded case.

2 The Model

2.1 Combinatorial Auctions

We first briefly describe the general standard setting of combinatorial auctions.

We consider an auction of a set U of m distinct items to a set N of n bidders. We assume that bidder's value combinations of items, where items may be complements or substitutes of each other. Formally, each bidder j has a *valuation function* $v_j()$ that describes his valuation for each subset $S \subseteq U$ of items, i.e. $v_j(S) \geq 0$ is the maximal amount of money bidder $j \in N$ is willing to pay for the subset S.

An *allocation* $S_1, ..., S_n$ is a partition of the items U among the bidders. We consider here auctions that aim to maximize the total *social welfare*, $w = \sum_j v_j(S_j)$, of the allocation. The auction rules describe a *payment* p_j for each bidder j. We assume the bidders have *quasi linear utilities*, so bidder j's overall utility for winning the set S_j and paying p_j is $u_j = v_j(S_j) - p_j$.

2.2 Known Single Minded Bidders

In this paper we only discuss a limited class of bidders, single minded bidders, that were introduced by [7].

Definition 1 *[7] Bidder j is single minded if there is a set of goods $S_j \subseteq U$ and a value $v_j^* \geq 0$ such that* $v_j(S) = \begin{cases} v_j^* & S \supseteq S_j \\ 0 & \textit{otherwise.} \end{cases}$

I.e., the bidder is willing to pay v_j^* as long as he is allocated S_j. We assume that each v_j^* is privately known to bidder j. We deviate from [7] and assume that the subsets S_j's are known to the mechanism (or alternatively can be independently deduced or authenticated by the mechanism). We call this case, *known single minded bidders*. It is easy to verify that all our results apply even if only the cardinality of S_j is known. Some of our results hold even if the S_j's are only privately known (as in [7]). We shortly describe this case in section 7.

2.3 The Mechanism

We consider only closed bid auctions where each bidder $j \in N$ sends his bid v_j to the mechanism, and then the mechanism computes an allocation and determines the payments for each bidder. The allocation and payments depend on the bidders' declarations $v = (v_1, \ldots, v_n)$. Thus the auction mechanism is composed of an *allocation algorithm* $A(v)$, and a *payment rule* $p(v)$.

Treated as an algorithm, the allocation algorithm A is given as input not only the bids $v_1 ... v_n$, but also the sets $S_1 ... S_n$ that are desired by the bidders. Its output specifies a subset $A(v) \subseteq N$ of *winning bids* that are pair-wise disjoint, $S_i \cap S_j = \emptyset$ for each $i \neq j \in A(v)$. Thus bidder j wins the set S_j if $j \in A(v)$ and wins nothing otherwise.

Let v_{-j} be the partial declaration vector $(v_1, ..., v_{j-1}, v_{j+1}, ..., v_n)$, and let $v = (v_j, v_{-j})$. For given valuations v_{-j} and allocation algorithm A, we say that v_j is a *winning declaration* if $j \in A(v_j, v_{-j})$. Otherwise we say that v_j is a *loosing declaration*. Sometimes we shall simply say that j wins S_j if $j \in A(v)$.

The *social welfare* obtained by the algorithm is thus $w_A(v) = \Sigma_{j \in A(v)} v_j$. While our allocation algorithms attempt maximizing this social welfare, they of course can not find optimal allocations since that it is NP-hard ("weighted set packing") and we are interested in computationally efficient allocation algorithms.

2.4 Bidders' strategies

Bidder's j utility in a mechanisms (A, p) is thus $u_j(v) = v_j - p_j$ if $j \in A(v)$, and $u_j(v) = -p_j$ otherwise. A mechanism is *normalized* if non-winners pay zero, i.e. $p_j = 0$ for all $j \notin A(v)$. In this case $u_j = 0$ for all $j \notin A(v)$.

Bidder j may strategically prefers to declare a value $v_j \neq v_j^*$ in order to increase his utility. We are interested in truthful mechanisms where this does not happen.

Definition 2 *A mechanism is called truthful (equivalently,* incentive compatible*) if truthfully declaring $v_j = v_j^*$ is a dominant strategy for each bidder. I.e. for any declarations of the other bidders v_{-j}, and any declaration v_j of bidder j, $u_j(v_j^*, v_{-j}) \geq u_j(v_j, v_{-j})$.*

2.5 Multi unit Auctions

In a *Multi Unit Combinatorial Auction* we have many types of items and many identical items of each type. We consider a multiset U with m different types of items, where m_i is the number of identical items of type $i = 1, \ldots, m$. Let M be the total number of goods, that is $M = \Sigma_{i=1}^{m} m_i = |U|$.

The special case $m = 1$ where all items are identical, is called a *Multi Unit Auction*. The allocation problem of Multi Unit Auction is a special case of the knapsack problem.

All our results apply also to multi-unit combinatorial auctions, and so we assume that a single minded bidder is interested in fixed number of goods of each type. I.e. instead of having a single set S_j, a single-minded bidder has a tuple $q_1 ... q_m$, specifying that he desires (has value v_j) a multiset of items that contains at least q_i items of type i, for all i.

3 Characterization of Truthful Mechanisms

It is well known that truthful mechanisms are strongly related to certain monotonicity conditions on the allocation algorithm. This was formalized axiomatically in the context of combinatorial auctions with single minded bidders in [7]. We present here a very simple characterization for the case of known single minded bidders. This characterization reduces the problem of designing truthful mechanisms to that of designing monotone algorithms, which is then considered throughout the rest of the paper.

3.1 Monotone Allocation Algorithms

An allocation algorithm is monotone if, whenever S_j is allocated and the declared valuation of j increases, then S_j remains allocated to j. Formally:

Definition 3 *An allocation algorithm A is monotone if, for any bidder j and any v_{-j}, if v_j is a winning declaration then any higher declaration $v'_j \geq v_j$ also wins.*

Lemma 1 *Let A be a monotone allocation algorithm. Then, for any v_{-j} there exists a single critical value $\theta_j(A, v_{-j}) \in (R_+ \cup \infty)$ such that $\forall v_j > \theta_j(A, v_{-j})$, v_j is a winning declaration, and $\forall v_j < \theta_j(A, v_{-j})$, v_j is a loosing declaration.*

Proof: Non-existence of critical value for v_{-j} means that for any $c \in [0, \infty]$ there are v''_j and v'_j such that $v''_j \leq c < v'_j$ or $v''_j < c \leq v'_j$, where v''_j, v'_j are winning and loosing declarations, respectively. This contradicts A's monotonicity. If $\theta_1 \neq \theta_2$ are distinct critical values, then $\frac{\theta_1+\theta_2}{2}$ is simultaneously winning and loosing declaration for j, a contradiction. ∎

Fix an algorithm A and bids of the other bidders, v_{-j}. Note that $\theta_j = \theta_j(A, v_{-j})$ is the infimum value that j should declare in order to win S_j. In particular, note that θ_j is independent of v_j. Consider an auction of a single item. It is easy to see that the winner's critical value is the value of the 2nd highest bid. Note that the 2nd price (Vickerey) auction fixes this value as the payment scheme. This can be generalized.

Definition 4 *The payment scheme p_A associated with the monotone allocation algorithm A that is based on the critical value is defined by: $p_j = \theta_j(A, v_{-j})$ if j wins S_j, and $p_j = 0$ otherwise.*

3.2 The Characterization

It turns out that monotone allocation algorithms with critical value payment schemes capture essentially all truthful mechanisms. Formally they capture exactly truthful normalized mechanisms, (those where losers pay zero), but any truthful mechanism can be easily converted to a normalized one (by adjusting the payment scheme in the following way: $p'_j = p_j(S) - p_j(\emptyset)$).

Theorem 1 *A normalized mechanism is truthful if and only if its allocation algorithm is monotone and its payment scheme is based on critical value.*

The theorem is a consequence of the following two lemmas.

Lemma 2 *Let (A, p) be a truthful normalized mechanism. Then A is monotone and p is based on critical value.*

Proof: Here p_j should be no more than the declared value v_j (otherwise the bidder would prefer to gain zero utility by simply declaring untruthfully zero). Fix v_{-j}, and consider the values $v_j < v'_j$. Assume by contradiction that v_j, v'_j are winning and loosing declarations, respectively, and that

$v'_j = v^*_j$. In this case j would prefer to declare untruthfully v_j in order to gain positive utility. The monotonicity follows.

Consider the following scenario: Bidder j truthfully declares v_j, wins S_j and pays $p_j > \theta_j$. However, $\theta_j + \epsilon$ is a winning declaration with a lower payment and hence a higher utility. Therefore $p_j \leq \theta_j$. For the case $p_j < \theta_j$, consider j with true value v^*_j, where $p_j < v^*_j < \theta_j$. Bidding truthfully results in zero utility. Bidding untruthfully $\theta_j + \epsilon$, the bidder wins S_j, pays p_j and gains positive utility. Hence the payment for S_j is exactly $p_j = \theta_j$. ∎

Lemma 3 *Let A be monotone allocation algorithm and p_A the associated critical value payment scheme. Then the mechanism (A, p_A) is truthful.*

Proof: Fix v_{-j}. Let S', S be the sets bidder j wins, and u'_j, u_j the utilities j gains when bidding v'_j, v^*_j, respectively. $u'_j \leq u_j$ in each of the following cases, as a result truthful bidding is a dominant strategy.

1. $S' = S$. The payment is independent of j's declaration, hence the utilities are equal.
2. $S' = \emptyset$, $S = S_j$. Clearly $v'_j < \theta_j \leq v^*_j$. Then, $u'_j(\emptyset) = 0 \leq v^*_j - \theta_j = v_j(S_j) - p_j(S_j) = u_j(S_j)$.
3. $S' = S_j$, $S = \emptyset$. Clearly $v'_j \geq \theta_j > v^*_j$. Then, $u'_j(S_j) = v^*_j - \theta_j = v_j(S_j) - p_j(S_j) \leq 0 = u_j(\emptyset)$.

∎

3.3 Bitonic Allocation Algorithms

We use a special case of monotone allocation algorithms, called *bitonic*. Given a monotone algorithm A, the property of bitonicity involves the connection between v_j and the social welfare of the allocation $A(v_j, v_{-j})$. What it requires is that the welfare does not increase with v_j when v_j looses, $v_j < \theta_j$, and does increase with v_j when v_j wins, $v_j > \theta_j$ (see fig. 1).

Definition 5 *A monotone allocation algorithm A is bitonic if for every bidder j and any v_{-j}, one of the following conditions holds: (i) The welfare $w_A(v_{-j}, v_j)$ is a non-increasing function of v_j for $v_j < \theta_j$ and a non-decreasing function of v_j for $v_j \geq \theta_j$; or, (ii) The welfare $w_A(v_{-j}, v_j)$ is a non-increasing function of v_j for $v_j \leq \theta_j$ and a non-decreasing function of v_j for $v_j > \theta_j$.*

One would indeed expect that a given bid does not affect the allocation between the other bids, and thus for $v_j < \theta_j$ we would expect w_A to be constant, and for $v_j > \theta_j$ we would expect w_A to grow linearly with v_j. Most of our examples, as well as the optimal allocation algorithm, indeed follow this pattern. This need not hold in general though and there do exist non-bitonic monotone algorithms.

Example 1 *As a simple example for non-bitonic monotone allocation algorithm consider the following algorithm.*

XOR-algorithm(Y, i, j, k)

Input: $Y \geq 1$ and $i, j, k \in N$.
If the valuation v_i of bidder i is below Y then
bidder j wins.

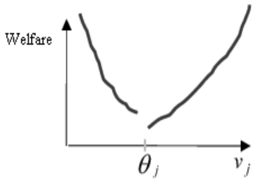

Figure 1: A curve of a bitonic allocation algorithm.

Else if v_i is below $2Y$ then
bidder k wins.
Else bidder i wins.

The XOR-algorithm is monotone for all the bidders (where the critical value for a bidder l other then i is zero or infinity, and the critical value for bidder i is $2Y$). Focusing on bidder i. observe that the welfare in the interval $[0, 2Y)$ may be increasing, and so the XOR-algorithm is not bitonic in general for bidder i.

4 Some Basic Truthful Mechanisms

In this section we present several monotone allocation algorithms. Each of them may be used as a basis for a truthful mechanism. They can also be combined between themselves using the operators described in section 5.

4.1 Greedy

The main algorithmic result of [7] was the identification of the following scheme of greedy algorithms as truthful. First the bids are reordered according to a certain "monotone" ranking criteria. Then, considering the bids in the new order, bids are allocated greedily. We start with a slight generalization of their result.

Definition 6 *A ranking r is a collection of n real valued functions $(r_1(), r_2(), \ldots, r_n())$, where $r_j() = r_j(v_j, S_j), j \in N$. A ranking r is monotone if each $r_j()$ is non-decreasing in v_j.*

We will use the following monotone rankings.

1. The **value ranking** [7]: $r_j(.) = v_j$.
2. The **density ranking** [7]: $r_j(.) = \frac{v_j}{|S_j|}$.
3. The **compact density ranking by k**: Ties are broken lexicographically.

$$r_j(.) = \begin{cases} v_j & |S_j| \leq \sqrt{\frac{M}{k}} \\ 0 & otherwise \end{cases}$$

Where $k > 0$ is fixed.

Greedy Algorithm G_r based on ranking r

1. Reorder the bids by decreasing value of $r_j(v_j, S_j)$.
2. *WinningBids* $\leftarrow \emptyset$, *NonAllocItems* $\leftarrow U$.
3. For $j = 1..n$ (in the new order) if ($S_j \subseteq$ *NonAllocItems*) then
   - $WinningBids \leftarrow WinningBids \cup \{j\}$.
   - $NonAllocItems \leftarrow NonAllocItems - S_j$.
4. Return *WinningBids*.

Observe that G_r runs in polynomial time as long as its ranking function is efficient.

Lemma 4 *(essentially due to [7]) Any greedy allocation scheme G_r that is based on a monotone ranking r is monotone.*

Proof: For any v_{-j} and $v_j \leq v'_j$, the monotonicity of r implies $r_j(v_j) \leq r_j(v'_j)$. If v_j is a winning declaration for j, then when G_r considers satisfying S_j, say in the k'th iteration, no conflict occurs, that is $S_j \subseteq$ *NonAllocItems*. Moreover, considering S_j in any former iteration would imply no such conflict, since *NonAllocItems* in any former iteration is a superset of *NonAllocItems* in the beginning of the k'th iteration. The ranking values of all other bids are identical for the two possible inputs. Hence declaring v'_j ensures considering S_j in an iteration $\leq k$. It follows that v'_j is a winning declaration. ∎

It turns out that a greedy algorithm is in fact bitonic.

Lemma 5 *Any greedy allocation scheme G_r that is based on a monotone ranking r is bitonic.*

Proof: Consider any v_{-j}. By Lemma 4 G_r is monotone with the critical value $\theta = \theta_j(G_r, v_{-j})$. Let $f(y) = w_{G_r}(v_{-j}, y)$, that is $f(y)$ is the welfare of the allocation $G_r(v_{-j}, y)$. Assume w.l.o.g that declaring θ by bidder j is a winning declaration. $\forall y < \theta : f(y) = f(0)$, since declaring y or zero by bidder j results in the same allocation (without S_j). Hence $f()$ is a constant function on $[0, \theta)$. In addition, $\forall y \geq \theta$ declaring y or θ result in the same allocation with S_j, and so $f(y) = f(\theta) + y - \theta$. We conclude that $f()$ is a linear increasing function on $[\theta, \infty)$. ∎

4.2 Partial Exhaustive Search

The second algorithm we present, performs an exhaustive search over all combinations of at most k bids. The running time is polynomial for every fixed k.

Exhaustive-k Search Algorithm

1. *WinningBids* $\leftarrow \emptyset$, *Max* $\leftarrow 0$.
2. For each (subset $J \subseteq \{1, \ldots, n\}$ such that $|J| \leq k$):
   if ($\forall i, j \in J,\ i \neq j : S_i \cap S_j = \emptyset$) then

- if $(\Sigma_J v_i > Max)$ then
  $Max \leftarrow \Sigma_J v_i$ and $WinningBids \leftarrow J$.

3. Return *WinningBids*.

The extreme cases are of interest: Exhaustive-1 simply returns the bid with the highest valuation; and Exhaustive-n is the naive optimal algorithm which searches the entire solution space. We shall use Exhaustive-1, and hence give it an additional name.

Largest Algorithm Return the bid with the highest valuation $v_h = max_{j \in N} v_j$.

Lemma 6 *For every k, Exhaustive-k is monotone and bitonic.*

Proof: Consider any v_{-j} and values $v_j < v'_j$, where we assume by contradictory that v_j, v'_j are winning and loosing declarations, respectively. Let *WinningBids*, *WinningBids'* be the respective allocations and *Max*, *Max'* be the respective welfares. Algorithm Exhaustive-k considers *WinningBids* and *WinningBids'* for both possible declarations. It must be the case that $Max' \geq Max - v_j + v'_j > Max$. A contradiction, since Exhaustive-k on the input (v_{-j}, v_j) should output *WinningBids'* instead of *WinningBids*. The monotonicity of Exhaustive-k follows. The rest of the proof is similar to the proof of lemma 5. ∎

4.3 LP based

Since the combinatorial auction problem is an integer programming problem, many authors have tried heuristics that follow the standard approach of using the linear programming relaxation [11, 20, 19]. In general such heuristics are not truthful. In this section we present a very simple heuristic based on the LP relaxation that is truthful.

In this subsection we use general notation of multi unit combinatorial auctions. The multiset S_j can be regarded as the m-tuple $(q_{1j}, \ldots, q_{mj})$, where q_{ij} it the number of items of type i in S_j. The optimal allocation problem can be formulated as the following integer program, denoted $IP(v)$.

$$\text{maximize} \quad \Sigma_{j=1}^{n} z_j v_j$$

subject to:

$$\begin{array}{ll} \Sigma_{j=1}^{n} z_j\, q_{ij} \leq m_i & i = 1, \ldots, m \\ z_j \in \{0,1\} & j = 1, \ldots, n \end{array}$$

Removing the integrality constraint we get the following linear program relaxation, denoted $LP(v)$:

$$\text{maximize} \quad \Sigma_{j=1}^{n} x_j v_j$$

subject to:

$$\begin{array}{ll} \Sigma_{j=1}^{n} x_j\, q_{ij} \leq m_i & i = 1, \ldots, m \\ x_j \in [0,1] & j = 1, \ldots, n \end{array}$$

Natural heuristics for solving the integer program would attempt using the values of x_j in order to decide on the integral allocation. We show that the following simple rule does indeed provide a monotone allocation rule.

LP-Based Algorithm

1. Compute an optimal basic solution x for $LP(v)$.
2. Satisfy all bids j for which $x_j = 1$.

Theorem 2 *Algorithm LP-Based is monotone.*

Observe the for the case of combinatorial auction (where $m_i = 1, i = 1..m$), as opposed to the general case of multi unit combinatorial auction, we could have taken the threshold of 1/2, that is to satisfy all bids with $x_j > \frac{1}{2}$.

Notations: Let $v_j \leq v'_j$ and so $\Delta = v'_j - v_j$ is nonnegative. Let x, x' be optimal feasible solutions to $LP(v)$, $LP(v')$, respectively, where $v' = (v_{-j}, v'_j)$. The proof of theorem 2 is a consequence of the following lemma.

Lemma 7 *For any v_{-j}, x_j is a non-decreasing function of v_j.*

Proof: x' is feasible solution to $LP(v)$ and so $\Sigma_{l=1}^n x'_l v_l \leq \Sigma_{l=1}^n x_l v_l$. Similarly, x is feasible to $LP(v')$ and so $x_j \Delta + \Sigma_{l=1}^n x_l v_l \leq x'_j \Delta + \Sigma_{l=1}^n x'_l v_l$. Then, $0 \leq \Sigma_{l=1}^n (x_l - x'_l) v_l \leq (x'_j - x_j)\Delta$. Thus $x_j \leq x'_j$. ∎

Proof (of theorem): v_j is a winning declaration for bidder j iff $x_j = 1$. Using lemma 7 we get that $1 = x_j \leq x'_j \leq 1$, and hence $v'_j \geq v_j$ is a winning declaration as well. ∎

We will also later require the follow property of the LP solution.

Lemma 8 *Let $\sigma = \Sigma_{l=1}^n x_l v_l$ be the optimal objective function value of $LP(v)$, and similarly $\sigma' = x'_j \Delta + \Sigma_{l=1}^n x'_l v_l$. Then for any v_{-j}, and $v_j \leq v'_j$,*
$\sigma \leq \sigma' \leq \sigma + \Delta$.

Proof: The optimal solution vector x to $LP(v)$ is a feasible solution to $LP(v')$, as both have the same set of constraints. Thus we have $\sigma \leq \sigma'$. Similarly, x' is a feasible solution vector to $LP(v)$, and hence $\Sigma_{l=1}^n x'_l v_l \leq \Sigma_{l=1}^n x_l v_l$. Finally, $\sigma' = \Sigma_{l=1}^n x'_l v_l + x'_j \Delta \leq \Sigma_{l=1}^n x_l v_l + x'_j \Delta \leq \sigma + \Delta$. ∎

5 Combining Truthful Mechanisms

In this section we present two techniques for combining monotone allocation algorithms as to obtain an improved monotone allocation algorithm. These combination operators together with the previously presented algorithms provide a general algorithmic toolbox for constructing monotone allocation algorithms and thus also truthful mechanisms. This toolbox will be used on section 6 in order to construct truthful approximation mechanisms for various special cases of combinatorial auctions.

5.1 The MAX Operator

Perhaps the most natural way to combine two allocations algorithms is to try both and pick the best one – the one providing the maximal social welfare.

MAX $(A_1,\, A_2)$ **Operator**

1. Run algorithm A_1.

2. Run algorithm A_2.

3. if $w_{A_1}(v) \geq w_{A_2}(v)$ return $A_1(v)$ else return $A_2(v)$.

Unfortunately this algorithm is not in general guarenteed to be monotone. For example the maximum of two XOR-algorithms (see example 1) with the parameters sets Y, i, j, k and Y', i, j', k', is not monotone in general for bidder i. We are able to identify a condition that does ensure that that this algorithm is monotone.

Theorem 3 *Let A_1 and A_2 be two monotone bitonic allocation algorithms. Then, $M = MAX(A_1, A_2)$ is a monotone bitonic allocation algorithm.*

Proof: Fix v_{-j}, and set $\theta' = \theta_j(A_1, v_{-j})$ and $\theta'' = \theta_j(A_2, v_{-j})$. Let $f_1(y) = w_{A_1}(v_{-j}, y)$, $f_2(v) = w_{A_2}(v_{-j}, y)$ and $f_m(v) = w_M(v_{-j}, y)$ be the respective welfares as a function of the declaration y of bidder j.

Recall the simple fact: if $f(y)$ and $g(y)$ are decreasing functions then so is $h(y) = \max\{f(y), g(y)\}$. Assuming w.l.o.g that $\theta' \leq \theta''$, we conclude that $\forall y < \theta'$ the function $f_m(y)$ is decreasing, and $\forall y > \theta''$ the function $f_m(y)$ is increasing. For the interval $I = [\theta', \theta'']$ consider the cases:

- $\theta' = \theta''$. Here f_1 and f_2 share the same global minimum. Clearly M is a bitonic allocation algorithm, and $\theta_j(M, v_{-j}) = \theta' = \theta''$.

- $\theta' < \theta''$ and $\forall y \in I : f_1(y) \geq f_2(y)$. Here $f_m() \equiv f_1()$. Then M is bitonic as algorithm A_1, and $\theta_j(M, v_{-j}) = \theta'$.

- $\theta' < \theta''$ and $\forall y \in I : f_1(y) < f_2(y)$. Here $f_m() \equiv f_2()$. Then M is bitonic as A_2, and $\theta_j(M, v_{-j}) = \theta''$.

- Otherwise, it must be the case that $f_1(\theta') < f_2(\theta')$ and $f_2(\theta'') \leq f_1(\theta'')$. Let J be the maximal interval $\{y \,|y \in I,\ f_1(y) < f_2(y)\}$ and let $\theta_0 = \sup J$. J is not empty, and $\theta' \leq \theta_0 \leq \theta''$. We argue that $\forall y \in [\theta',\, \theta_0) :\ f_2(y) \geq f_1(y)$ and $f_m() = f_2()$ is decreasing. Similarly, $\forall y \in (\theta_0, \theta''] :\ f_1(y) \geq f_2(y)$ and $f_m() = f_1()$ is increasing. Thus M is bitonic, and $\theta_j(M, v_{-j}) = \theta_0$.

∎

Since the maximum of two bitonic algorithms is also bitonic, then inductively the maximum of any number of bitonic algorithms is monotone.

5.2 The If-Then-Else Operator

The max operator had to run both algorithms. In many cases we wish to have conditional execution and only run one of the given algorithms, where the choice depends on some condition. This is the usual If-The-Else construct of programming languages.

If cond() Then $A1$ Else A_2 Operator

If $cond(v)$ holds
  return the allocation $A_1(v)$.
Else
  return the allocation $A_2(v)$.

The monotonicity of the two algorithms does not by itself guarentee that the combination is monotone. As a simple example consider the following algorithm: If $\Sigma_{i=1}^{n} v_i$ is even then the bid with largest valuation wins, otherwise bidder 1 wins. For a fixed v_{-j}, observe that if v_j is a winning declaration (j wins) then $v_j + 1$ is a loosing declaration (1 wins instead), and so the algorithm is not monotone for bidder j. We require a certain "alignment" between the condition and the algorithms in order to ensure monotonicity of the result.

Definition 7 *The boolean function cond() is aligned with the allocation algorithm A if for any v_{-j} and any two values $v_j \leq v'_j$ the following hold:*

1. *If $cond(v_{-j}, v_j)$ holds and $v_j \geq \theta_j(A, v_{-j})$ then $cond(v_{-j}, v'_j)$ holds.*
2. *If $cond(v_{-j}, v'_j)$ holds and $v'_j \leq \theta_j(A, v_{-j})$ then $cond(v_{-j}, v_j)$ holds.*

Theorem 4 *If A_1 and A_2 are monotone allocation algorithms and cond() is aligned with A_1 then the operator If-Then-Else $(cond, A_1, A_2)$ is monotone.*

Proof: For any v_{-j} and $v'_j \geq v_j$, consider the cases:

- $cond(v)$ holds and $j \in A_1(v)$. Here $cond(v')$ holds, where $v' = (v_{-j}, v'_j)$. But then the output of If-Then-Else$(cond, A_1, A_2, v')$ would be $A_1(v')$. Algorithm A_1 is monotone, and hence if v_j is a winning declaration for $A_1(v)$ then so is v'_j.
- $cond(v)$ fails, $j \in A_2(v)$ and $cond(v')$ fails. Here the output of If-Then-Else$(cond, A_1, A_2, v')$ is $A_2(v')$. The monotonicity of A_2 implies that if v_j is a winning declaration then so is v'_j.
- $cond(v)$ fails, $j \in A_2(v)$ and $cond(v')$ holds. Assume by contradiction, that $j \notin A_1(v')$. But then $cond(v)$ must hold (as the 2nd requirement in def. 7).

∎

6 Applications: Approximation Mechanisms

In this section we use the toolbox previously developed to construct truthful approximation mechanisms for several interesting cases of combinatoiral auctions (all with known single minded bidders). These mechanims all run in polynomial time and obtain allocations that are within a provable gap from the optimum.

6.1 Multi Unit Auctions

In multi-unit auctions we have a certain number of identical items, and each bidder is willing to offer the price v_j for the quantity q_j. Since bids cannot be partially fulfilled, we are in fact required to solve the NP-complete knapsack problem. Indeed, despite the vast economic literature, starting with Vickrey's seminal paper [18], that deals with multi-unit auctions, this case was never studied, and attention was always restricted to "downward sloping bids" that can always be partially fulfilled. While the knapsack problem has fully polynomial approximation schemes, these are not monotone and thus do not yield truthful mechanisms. We provide a truthful 2-approximation mechanism.

Let G_v be the algorithm Greedy based on a value ranking. Let G_d be the algorithm Greedy based on a density ranking.

Approx-MUA Algorithm

Return the allocation determined by MAX(G_v, G_d).

Theorem 5 *The mechanism with Approx-MUA as the allocation algorithm and the associated critical value payment scheme is 2-approximation truthful mechanism for multi unit auctions.*

The proof of theorem is a consequence of the following two lemmas.

Lemma 9 *Algorithm Approx-MUA is monotone.*

Proof: The proof is immediate from Theorem 3 as G_v and G_d are bitonic. ∎

Lemma 10 *Approx-MUA is a 2-approximation for the Multi Unit Auction allocation problem.*

The proof is similar to that in [8] and is omitted.

6.2 General Combinatorial Auctions

The general combinatorial auction allocation problem is NP-hard to approximate to within a factor of $m^{\frac{1}{2}-\epsilon}$ (for any fixed $\epsilon > 0$) [6, 17]. A truthful allocation mechanism with a nearly matching performance guarantee of $\sqrt{m}$ is given in [7] for the case of single minded bidders. We narrow the gap between the upper bound and lower bound even further and present truthful mechanisms with performance guarantee of $\epsilon\sqrt{m}$, for every fixed $\epsilon > 0$.

Let G_k be the Greedy algorithm Greedy based on the compact density ranking by k.

k-Approx-CA Algorithm

Return the allocation determined by MAX(Exhaustive-k, G_k).

Theorem 6 *The mechanism with $\lfloor \frac{4}{\epsilon^2} \rfloor$- Approx-CA as the allocation algorithm and the associated critical value payment scheme is an $(\epsilon\sqrt{m})$-approximation truthful mechanism.*

The proof of theorem is a consequence of the following two lemmas.

Lemma 11 *Algorithm k-Approx-CA is monotone.*

Proof: The proof is immediate from Theorem 3 as G_k and Exhaustive-k are bitonic. ∎

Lemma 12 *Choosing $k = \lfloor \frac{4}{\epsilon^2} \rfloor$, algorithm k-Approx-CA provides $\epsilon\sqrt{m}$-approximation for the combinatorial auction allocation problem.*

Proof: The proof follows the proof in [5].

6.3 Multi unit Combinatorial Auctions

Here we consider the general case of multi-unit combinatorial auctions. We provide a monotone allocation algorithm that provides good approximations in the case that the number of types of goods, m, is small.

<u>$(m+1)$**-Approx-MUCA Algorithm**</u>

1. Compute an optimal basic solution x to $LP(v)$.
2. Let $v_h = max_j v_j$.
3. If $\Sigma_{l=1}^{n} x_l v_l < (m+1)v_h$ Then return Largest(v); Else return LP-Based(v).

Theorem 7 *The mechanism with $(m+1)$-Approx-MUCA as the allocation algorithm and the associated critical value payment scheme is an $(m+1)$-approximation truthful mechanism for multi unit combinatorial auctions with m types of goods.*

The theorem is a consequence of the following lemmas.

Define the boolean function $f_h()$ to be *true* if $\sigma(LP(v)) < (m+1)v_h(v)$, and $false$ otherwise.

Lemma 13 *$(m+1)$-Approx-MUCA is monotone.*

The proof of this lemma follows immediately from the following lemma:

Lemma 14 *The condition $\Sigma_{l=1}^{n} x_l v_l < (m+1)v_h$ is aligned with the algorithm Largest.*

Proof: For the first requirement of alignment, clearly if $j \in Largest(v)$ then $v_j = v_h(v)$. Hence $v'_j = v_h(v')$ as $v'_j \geq v_j$ is the maximum among $v' = (v_{-j}, v'_j)$. Using the upper bound of lemma 8 and the fact that $f_h(v)$ holds, we get: $\sigma' \leq \sigma + \Delta < (m+1)v_h(v) + \Delta \leq (m+1)(v_h(v)+\Delta) = (m+1)v_h(v')$. Thus $f_h(v')$ holds.

For the second requirement, assume that $f_h(v')$ holds and $j \notin Largest(v')$. Clearly, $v_j \neq v_h(v) = v_h(v')$. By the lower bound of lemma 8: $\sigma \leq \sigma' < (m+1)v_h(v') = (m+1)v_h(v)$. Thus $f_h(v)$ holds. ∎

Lemma 15 *Algorithm $(m+1)$-Approx-MUCA provides an $(m+1)$-approximation for the Multi Unit Combinatorial Auction allocation problem.*

Proof: Let v^o be the optimal value. Consider the cases:

- The allocation is by Largest. It follows that $f_h(v)$ holds, that is $\sigma < (m+1)\ v_h$. Hence: $v^o \leq \sigma < (m+1)\ v_h$.
- The allocation is by LP-Based. Then $\sigma \geq (m+1)\ v_h$. Any basic solution has at most m strictly fractional components (see e.g. [2]). Recall that $w_{LP-Based}(v)$ is the value of all bids j with $x_j = 1$. Hence $(m+1)v_h \leq \sigma \leq w_{LP-Based}(v) + m \cdot v_h$ and so $v_h \leq w_{LP-Based}(v)$. Now, $v^o \leq \sigma \leq w_{LP-Based}(v) + m \cdot v_h \leq (m+1)\ w_{LP-Based}(v)$.

∎

7 Single Minded Bidders

Some (but not all) of our techniques apply to the more general model of Single Minded Bidders of [7]. In this section we shortly mention which techniques do generalize and how.

Let us recall that a single minded bidder j has a privately known (S_j, v_j^*), and it then submits to the mechanism a single bid of the form (T_j, v_j), where $T_j \subseteq U$. The definition of truthfulness of a mechanism, for single minded bidders, is that bidding the truth $(T_j, v_j) = (S_j, v_j^*)$ is a dominant strategy for all bidders j. An allocation algorithm A is *monotone* if for any bidder j and declarations of the other bidders (T_{-j}, v_{-j}), whenever (T_j, v_j) is a winning declaration for j so is any bid (T'_j, v'_j) where $T'_j \subseteq Tj$ and $v'_j \geq v_j$.

Theorem 8 *(essentially due to [7]) A normalized mechanism is truthful if and only if its allocation algorithm is monotone and its payment scheme is based on critical value.*

We mention whether and how each of our results generalizes.

- **Characterization:** The characterization of truthful mechanisms is now modified to include algorithmic monotonicity in T_j.
- **Basic algorithms**: All 3 basic algorithms (Exhaustive-k, LP-based, and Greedy) generalize. Greedy is due to [7] and requires the ranking r to be also monotone in T_j.
- **Operators:** If-Then-Else is monotone. The proof goes through once the definition of alignment is modified to take into account the declared sets. MAX is not monotone in general for single minded bidders.
- **Applications:** The approximation mechanisms presented previously in subsections 6.1 and 6.2 are not necessarily truthful for single minded bidders. However, we provide an alternative 2-approximation mechanism for multi-unit auctions with single minded bidders.

SMB-Approx-MU Algorithm

1. Re-index the bids so that: $\frac{v_1}{|T_1|} \geq \frac{v_2}{|T_2|} \geq \cdots \geq \frac{v_n}{|T_n|}$
2. Compute k to be the index such that $\sum_{j=1}^{k-1} |T_j| \leq M$, and $\sum_{j=1}^{k} |T_j| > M$, where M is the total number of identical items.
3. Compute $\sigma = \sum_{j=1}^{k-1} v_j$. Compute $v_h = \max_{j \in N} v_j$.
4. If $v_h \geq \sigma$ return Largest, otherwise return $K = \{1, \ldots, k-1\}$

Theorem 9 *The Mechanism with SMB-Approx-MU as the allocation algorithm and the associate critical value payment scheme is a 2-approximation truthful mechanism for multi unit auctions.*

The proof of theorem is a consequence of the following two lemmas.

Lemma 16 *SMB-Approx-MU is monotone allocation algorithm.*

Proof: Fix $b_{-j} = (T_{-j}, v_{-j})$. Let $b_j = (T_j, v_j)$ be a winning declaration. Consider $b'_j = (T'_j, v'_j)$ where $v_j \le v'_j$ and $|T'_j| \le |T_j|$, that is the average value per item of b'_j is higher than b_j. We say that b'_j is denser that b_j. Define $b = (b_{-j}, b_j)$ and $b' = (b_{-j}, b'_j)$. Let $\Delta = v'_j - v_j$. and consider two cases.

- $v_h(b) < \sigma(b)$. In this case $j \in K(b)$. Since b'_j is denser than b_j we conclude that $j \in K(b')$ as well. Thus $v_h(b') \le v_h(b) + \Delta < \sigma(b) + \Delta \le \sigma(b')$. Thus $v_h(b') < \sigma(b')$ and so b'_j is a winning declaration.

- $v_h(b) \ge \sigma(b)$. Here $v_j = v_h(b)$ and $v'_j = v_h(b')$. If $j \in K(b')$ then j is in both allocations and the result follows. Otherwise, $j \notin K(b')$. Since b_j is less denser it follows that $j \notin K(b)$, and so $\sigma(b) = \sigma(b')$. Hence $v_h(b') = v'_j = v_h(b) + \Delta \ge \sigma(b) = \sigma(b')$. Thus $v_h(b') \ge \sigma(b')$ and so b'_j is a winning declaration..

Lemma 17 *SMB-Approx-MU provides 2-approximation for the Multi Unit allocation problem.*

The proof is similar to that in [8] and is omitted.

8 Acknowledgements

We thank Ron Lavi and Liad Blumrosen for helpful comments and Gil Kalai for an early discussion.

References

[1] A. Archer and E. Tardos. Truthful mechanisms for one-parameter agents. In *to appear in the 42nd Annual Symposium on Foundations of Computer Science*, 2001.

[2] A. Caprara, H. Kellerer, U. Pferschy, and D. Pisinger. Approximation algorithms for knapsack problems with cardinality constraints. *European Journal of Operational Research*, 123:333–345, 2000.

[3] E. H. Clarke. Multipart pricing of public goods. *Public Choice*, pages 17–33, 1971.

[4] T. Groves. Incentives in teams. *Econometrica*, pages 617–631, 1973.

[5] Magnús M. Halldórsson. Approximations of weighted independent set and hereditary subset problems. *J. Graph Algorithms Appl.*, 4:no. 1, 16 pp., 2000.

[6] Johan Hastad. Clique is hard to approximate to within $n^{1-\epsilon}$. *Acta Mathematica*, 182, 1999.

[7] Daniel Lehmann, Liadan Ita O'Callaghan, and Yoav Shoham. Truth revelation in rapid, approximately efficient combinatorial auctions. In *1st ACM conference on electronic commerce*, 1999.

[8] S. Martello and P. Toth. *Knapsack Problems.* Wiley, 1990.

[9] Dov Monderer, Ron Holzman, Noa Kfir-Dahav, and Moshe Tennenholtz. Bundling equilibrium in combinatorial auctions. Technical report, Faculty of of Industrial Engineering and Managemen, Technion, 2001.

[10] Noam Nisan. Algorithms for selfish agents. In *STACS*, 1999.

[11] Noam Nisan. Bidding and allocation in combinatorial auctions. In *ACM Conference on Electronic Commerce*, 2000.

[12] Noam Nisan and Amir Ronen. Algorithmic mechanism design. In *STOC*, 1999.

[13] Noam Nisan and Amir Ronen. Computationally feasible VCG mechanisms. In *ACM Conference on Electronic Commerce*, pages 242–252, 2000.

[14] C. H. Papadimitriou. Algorithms, games, and the internet. In *STOC*, 2001.

[15] Jeffrey S. Rosenschein and Gilad Zlotkin. *Rules of Encounter: Designing Conventions for Automated Negotiation Among Computers.* MIT Press, 1994.

[16] T. Sandholm, S. Suri, A. Gilpin, and Levine D. Cabob: A fast optimal algorithm for combinatorial auctions. In *IJCAI*, 2001.

[17] Tuomas Sandholm. An algorithm for optimal winner determination in combinatorial auctions. In *IJCAI-99*, 1999.

[18] W. Vickrey. Counterspeculation, auctions and competitive sealed tenders. *Journal of Finance*, pages 8–37, 1961.

[19] Rakesh Vohra and Sven de Vries. Combinatorial auctions: A survey, 2000. Available from http://www.kellogg.nwu.edu/faculty/vohra/htm/res.htm.

[20] Edo Zurel and Noam Nisan. An efficient approximate allocation algorithm for combinatorial auctions. In *To appear in ACM conference on electronic commerce*, 2001. Available from http://www.cs.huji.ac.il/˜noam/mkts.html.