Single-Value Combinatorial Auctions and Implementation in Undominated Strategies

(Extended Abstract) *

Moshe Babaioff† Ron Lavi‡ Elan Pavlov§

Abstract

In this paper we are interested in general techniques for designing mechanisms that approximately maximize the social welfare in the presence of selfish rational behavior. We demonstrate our results in the setting of Combinatorial Auctions (CA). Our first main result is a general deterministic technique to decouple the algorithmic allocation problem from the strategic aspects, by a procedure that converts any *algorithm* to a dominant-strategy ascending *mechanism*. This technique works for any single value domain, in which each agent has the same value for each desired outcome, and this value is the only private information. In particular, for "single-value CAs", where each player desires any one of several different bundles but has the same value for each of them, our technique converts any approximation algorithm to a dominant strategy mechanism that almost preserves the original approximation ratio. Our second main result provides the first computationally efficient deterministic mechanism for the case of single-value multi-minded bidders (with private value and private desired bundles). The mechanism achieves an approximation to the social welfare which is close to the best possible in polynomial time (unless ZPP=NP). This mechanism is an *implementation in undominated strategies*, as well as an *algorithmic implementation*, notions that we justify and are of independent interest.

1 Introduction

Algorithmic Mechanism Design [18] studies the design of computationally efficient algorithms under the assumption that the input is distributed among a set of rational selfish players. One successful approach to this problem is to design *truthful* mechanisms, in which a player always maximizes his utility by simply revealing his true input. To date, the classic VCG [9, 10, 21] scheme is the only general technique to create truthful mechanisms. Unfortunately, it is computationally infeasible in most interesting domains, including Combinatorial Auctions. Thus, the question of designing computationally-efficient algorithms for a setting of selfish behavior is an important task.

In this context, this paper makes several contributions. We first describe in Section 1.1 a general technique for converting arbitrary algorithms into strategic mechanisms. We then describe in Section 1.2 how this technique may be applied in the significantly less restrictive setting of non-single-minded bidders, for which no non-trivial deterministic mechanism was previously known. This mechanism is an *implementation in undominated strategies*, as well as an *algorithmic implementation*, notions that we discuss in Section 1.3.

1.1 The Iterative Ascending Wrapper We first consider the issue of general methods for creating dominant-strategy mechanisms. In principle we would like to decouple the algorithmic problem (solving or approximating the solution) from the strategic problem (eliciting the preferences) thus allowing us to utilize the entire field of approximation algorithms without reference to the strategic properties. Unfortunately, the literature lacks such general techniques. Awerbuch, Azar and Meyerson [3] give a general technique for the case where the only private information of each player is his value. However their method is randomized, and needs a-priori knowledge of the range of possible values.

Our first contribution is a deterministic technique, that needs no a-priori bound on the maximal value of a player, and converts any given *algorithm* into a dominant-strategy ascending *mechanism* for any single value domain. We demonstrate our technique on the extensively studied CA domain (see e.g. the text-

*The full paper is available at the authors' websites.

†moshe@sims.berkeley.edu. The School of Information Management and Systems (SIMS), University of California at Berkeley. Supported by a National Science Foundation Grant Number ANI-0331659. This work was carried out while this author was a student at The Hebrew University.

‡ronlavi@ist.caltech.edu. Social and Information Sciences Laboratory, California Institute of Technology.

§elan@cs.huji.ac.il. School of Engineering and Computer Science, The Hebrew University of Jerusalem, Israel.

book [19]): Allocating a set Ω of m items to n players[1], where player i has value $\bar{v}_i(s)$ for every subset of items s. We assume monotonicity, i.e $\bar{v}_i(s) \subseteq \bar{v}_i(t)$ for every $s \subseteq t$, and that $\bar{v}_i(\emptyset) = 0$. The goal is to maximize the sum of true values of players for the subsets they receive. For CAs with "known" single minded bidders [16], where each player desires one specific publicly known subset of items for a private value of $\bar{v}_i$, our method preserves the original approximation ratio of the underlying algorithm up to a logarithmic factor:

Theorem: *Any c-approximation algorithm for known single-minded players can be deterministically converted to a dominant-strategy mechanism with $O(log(\bar{v}_{max})\cdot c)$-approximation.*[2]

We extend this result and construct dominant strategy mechanisms for the more general case in which each player is multi-minded (with publicly known desired bundles), but has the same private value for all desired bundles. The resulting approximation in this case is $O(log^2(\bar{v}_{max})\cdot c)$. While our method cannot improve the existing results for single-minded CA, as these already obtain the best possible $\sqrt{m}$-approximation [15], it is fruitful for special cases in which some further structure on the bundles is assumed, allowing to break the $\sqrt{m}$ lower bound (see e.g. [1]). It is additionally fruitful if one wishes to use the many well-known heuristics that work well in practice but are not truthful to begin with.

The construction: Our technique is based on wrapping an algorithm with an iterative ascending procedure in which players compete by gradually increasing their offers, and the winners pay their final bids. Specifically, a vector of player values, initialized to $\vec{1}$ (the minimal possible value), is iteratively handed in as input to the algorithm, which, in return, outputs a set of winners. Every loser is then required to either double his value or to permanently retire. This is iterated until all non-retired players are declared winners by the algorithm. These are the winners of the mechanism, and each pays his final bid. The analysis shows that, if the original algorithm is a c-approximation to the social welfare, and the wrapper procedure performs at most J iterations, then the resulting mechanism is an $O(c \cdot J)$-approximation. Surprisingly, for known single minded bidders it is possible to show that the number of iterations is independent of the number of players, and is at most $O(log(\bar{v}_{max}))$, hence the result follows.

This method also has the interesting property of converting the given algorithm to an ascending auction: players compete by placing bids, and winners pay their last offer. This has a more realistic structure than standard direct revelation mechanisms (in which each player simply reveals his value), and, in particular, has the advantage that winners do not reveal their true values. It has been argued many times (see e.g. [20]) that such indirect mechanisms should be preferred over the more common "direct revelation" mechanisms.

1.2 The Single Value Multi Minded Combinatorial Auction Domain

No non-trivial approximation mechanism which is deterministically truthful for general combinatorial auctions is known. By restricting the valuations to single minded players, Lehmann et. al. [15] were able to achieve a truthful $\sqrt{m}$-approximation mechanism. We suggest to look at a more general class of valuations, in which a player desires several different bundles, all for the same value:

DEFINITION 1.1. (SINGLE-VALUE PLAYERS) *Player i is a* single-value *(multi-minded) player if there exists a real value $\bar{v}_i > 0$ such that for any bundle s, $\bar{v}_i(s) \in \{0, \bar{v}_i\}$.*

In other words, the player desires any bundle that belongs to a *collection* of bundles $\bar{S}_i$, each for a value of $\bar{v}_i$. Both the player's value and his collection of desired bundles are assumed to be private information, known only to the player.

This model is significantly richer than single-minded players, as a single value player may be multi-minded, and may desire even an exponential number of bundles that are not contained one in the other. We also remark that this model does not fall into the family of single parameter domains defined by Archer and Tardos [2] since the desired bundles are not public information, and hence value monotonicity by itself is no longer sufficient for dominant strategy implementation. In [5] we discuss general single value domains and the sufficient conditions for dominant strategy implementations in such domains.

A natural single value multi-minded CA domain is the Edge Disjoint Paths (EDP) problem: given a graph, (graph edges may be thought of as the items for sale), each player i obtains a value $\bar{v}_i$ from receiving any path from a source node s_i to a target node t_i. The algorithm is required to allocate edge disjoint paths to maximize the sum of values of players that receive a desired path. In this model, players are naturally single valued (a player obtains the same value from any source-target path), but are **not** single minded, and in general the source-target pair of a player is **not** publicly known.

[1] An allocation is a tuple of disjoint subsets $s_1, ..., s_n$ of items, where the meaning is that player i gets the items in s_i. Some items may be left unallocated.

[2] We assume that $\bar{v}_i \geq 1$, and let $\bar{v}_{max}$ be the maximal value of any player (this need not be known a-priori).

Though much work has been devoted to CAs (see for example [15, 17, 7, 8]), no polynomial time dominant strategy mechanism with a non-trivial approximation ratio is known to date for general CAs, nor for CAs with single-value players. In particular, no dominant strategy mechanism for EDP is known. In fact, when considering *multi minded* players, only two strategic mechanisms are known: the deterministic mechanism of [6] (where they assume $B \geq 3$ copies of each item), and the randomized mechanism of [14], which for our case has truthfulness-in-expectation as an ex-post Nash equilibrium.

Our second contribution in this paper is a deterministic strategic mechanism for single value players, with an $O(log^2(\bar{v}_{max}) \cdot \sqrt{m})$ approximation. The approximation here is obtained for all rational strategic choices of the players: the mechanism admits more than one reasonable strategic choice for a rational player (and not just one as implied by dominant strategy implementation), but the approximation is guaranteed for *any* rational choices that the players make. We formally define and justify this concept in the next subsection.

Theorem: *There exists a deterministic mechanism for CAs with single-value multi minded players, that obtains an $O(log^2(\bar{v}_{max}) \cdot \sqrt{m})$ approximation for every rational strategic choices that players make.*

No a-priori knowledge of $\bar{v}_{max}$ is assumed. This method immediately applies also to the EDP problem.[3] In the full paper we show that this additionally applies to cases where players are only δ-close to single valued, where a player has different values for different bundles, and the ratio of any two values of the same player is at most δ. This results in an additional approximation loss of δ.

The construction: This mechanism is composed of our general iterative wrapper technique, on top of a modification of the 1-CA algorithm of [16]. Since now the collection of desired bundles of a player is also private information, we maintain an "active set" for each player, and a winner is allocated his last active set. The active set may be shrinked by a losing player, gradually focusing on one of his desired bundles. This additional building block of active sets is the reason for the transition from dominant strategies to undominated strategies (a formal definition is given below): if a player shrinks his active subset, he loses the ability to win some of his desired bundles, as any subset he will eventually win must be contained in his active set, and the active set is not allowed to expand. Therefore, a losing player faces the trade-off of doubling his value, by this increasing his price, or shrinking his subset. A mechanism with dominant strategies implies that the player has a specific optimal choice, no matter what the other players choose. In our mechanism there is no such "clear cut" decision for a player, hence we cannot predict what the player will choose. The crucial point in the analysis shows that such a unique optimal choice is also not necessary: to guarantee the approximation, all we need to ensure is that a *retiring* player reaches both his true value *and* one of his true desired subsets, the exact "path" is not important! This is indeed what our mechanism ensures for every undominated strategy, hence the approximation is achieved.

1.3 Implementation in Undominated Strategies

Requiring that players have dominant strategies limits the family of allocation algorithms that can be used [2, 5, 13]. Clearly, this requirement is not the real essence, but rather a tool to obtain the underlying goal: reaching approximately optimal outcomes in the presence of selfish behavior. In this paper we obtain this goal although we allow the mechanisms to leave in several "reasonable" strategies for the players to choose from. The basic assumption still remains that a player prefers not to choose a *dominated* strategy, since this means that there exists another strategy that *always* performs at least as well[4]. Thus, it is rational for a player to move from a given strategy to another strategy that dominates it. Our notion of algorithmic implementation captures this intuition, even if players are computationally bounded:

DEFINITION 1.2. *A mechanism M is an* algorithmic implementation of a c-approximation *if there exists a set of strategies, D, with the following properties:*

1. *M obtains a c-approximation for any combination of strategies from D, in polynomial time.*
2. *For any strategy that does not belong to D, there exists a strategy in D that dominates it. Furthermore, we require that this "improvement step" can be computed in polynomial time.*

Thus, while this definition leaves some uncertainty on the game-theoretic side, it compensates by strengthening the algorithmic analysis, showing that the mechanism performs well for *any combination of strategies from the set D*. This moves some burden from the game

[3] The currently best non-truthful algorithm for EDP has an approximation ratio of $O(min(|V|^{\frac{2}{3}}, \sqrt{m}))$ [11]. For a directed graph it is NP-hard to approximate within $m^{1/2-\epsilon}$, $\forall \epsilon > 0$ [11].

[4] Formally, let $u_i(s_i, s_{-i})$ denote the resulting utility of player i when he plays s_i and the others play s_{-i}. A strategy s'_i of player i is (weakly) *dominated* by another strategy s_i if, for every s_{-i}, $u_i(s_i, s_{-i}) \geq u_i(s'_i, s_{-i})$.

theoretic constraints on mechanism design to the algorithmic part in mechanism design. The game-theoretic arguments should only show that a player will indeed play some strategy in D, an argument that follows easily since we provide a polynomial time algorithm that moves from any given strategy to a strategy in D that dominates it. This ensure that even a computationally bounded player will indeed play a desired strategy.

Our new notion is related to the concept of implementation in undominated strategies[5] in the following way. The domination relation casts a partial order over the set of strategies. In a dominant strategy implementation, there exists a single equivalence set of all maximal elements, and the approximation is achieved assuming each player plays one specific maximal element. In a game without dominant strategies, there exist several equivalence sets of maximal elements. [6] In an *undominated strategy implementation*, there exists a specific choice of a representative strategy from each equivalence set of maximal elements, such that the approximation is achieved *for every profile of representative strategies of all players*.

Notice that our notion of algorithmic implementation with a set D implies the required conditions from an implementation in undominated strategies, since our requirements imply that D must contain at least one representative from every equivalence set of maximal elements. As we are interested in a computationally bounded players, we will focus on the notion of algorithmic implementation, yet by the above argument we also provide an implementation in undominated strategies. Although implementation in undominated strategies is a well-known game-theoretic concept, very few positive results have been achieved by using it [12]; We are not aware of any positive results for incomplete information settings, and, in particular, we are the first to adapt it to the context of algorithmic mechanism design and to demonstrate its usefulness.

We argue that the concept of algorithmic implementation captures all the truly important ingredients of the dominant strategies notion: First, the only assumption is that a player is willing to replace any chosen strategy with a strategy that dominates it. Indeed, this guarantees at least the same utility, even in the worst-case, and can be done in polynomial time. In addition, as in dominant strategies, our notion do not require any form of coordination among the players (unlike Nash equilibrium), or that players have any assumptions on the rationality behavior of the others (e.g. as in "iterative deletion of dominated strategies"). Our mechanisms also ensure that the resulting utility is non-negative, thus players have no risk in participating.

However, two differences do exist: (I) in our case, a player might regret his chosen strategy, realizing in retrospect that another strategy from D would have performed better (it is an implementation concept, not an equilibrium concept), and (II) while in dominant strategies it is a straight-forward task for a player to find a strategy to play, here finding the strategy to play is not that easy. While we make sure that a player does not end up playing a strategy that does not belong to D, we do not specify how to choose one out of the strategies of D. This may depend for example on the player's own beliefs about the other players, or on the computational power of the player.

In a companion paper [4], we present an additional technique for algorithmic implementation, completely different than our iterative wrapper method. This further demonstrates its potential usefulness.

The above results and discussion raise several open problems. First, the limitations of dominant strategy mechanisms for CA are unclear:

Open question 1: *Does there exist a deterministic polynomial time dominant strategy implementation for single value CAs with a $\sqrt{m}$ approximation?*

Second, the power of algorithmic implementations vs. dominant strategy mechanisms is unclear:

Open question 2: *Does there exist a problem domain in which an algorithmic implementation achieves a better approximation ratio than any dominant strategy implementation?*

The rest of the paper is organized as follows. Section 2 describes our Iterative Wrapper technique for known single minded players. In Section 3 we move to the general single-value bidders case, and describe our algorithmic implementation. In Section 4 we give the analysis. All missing details are given in the full paper.

2 The Iterative Ascending Wrapper for Known Single Minded Players

In this section we present our iterative wrapper technique for domains in which the only private information is the players value, with a focus on known single minded CAs. For this domain, our method converts *any* given c-approximation to a dominant strategy mechanism with $O(\log(\bar{v}_{max}) \cdot c)$ approximation, showing how to incorporate value monotonicity [7], which is equivalent

[5] A strategy is undominated if no other strategy dominates it.

[6] For two undominated strategies that are not in the same equivalence set s_i and s'_i, there exists two profiles of the other players' strategies s_{-i} and s'_{-i}, such that $u_i(s_i, s_{-i}) > u_i(s'_i, s_{-i})$, and $u_i(s_i, s'_{-i}) < u_i(s'_i, s'_{-i})$ so neither one dominates the other.

[7] For our domains, an algorithm is value monotone if a winner continues to win when he plays the dominant strategy with respect

The Iterative Wrapper Mechanism:

Allocation:

1: Let $j = 0$, $W_0 = \emptyset$, $L_0 = \emptyset$.
2: For any player $i \in N$ let $v_i^0 = 1$.
3: While $(W_j \cup L_j \neq N)$
4: $W_{j+1} = ALG(v^j)$
5: if $v(W_{j+1}, v^j) < v(W_j, v^j)$ then $W_{j+1} = W_j$.
6: For any $i \notin W_{j+1}$,
i chooses $v_i^{j+1} = 2v_i^j$ or $v_i^j = 0$ (retire).
7: $L_j = \{i \in N | v_i^j = 0\}$, j=j+1
8: J=j

Payments:

Each winner $i \in W_J$ pays his reported value v_i^J, losers pay 0.

Figure 1: The Iterative Wrapper Mechanism for Known Single-Minded CA. Note that $v(W, v) = \sum_{i \in W} v_i$.

to truthfulness [16], into any given algorithm.

A formal description of our method for the case of "known single-minded players" is given in Figure 1. Informally, suppose that ALG is an algorithmic procedure such that, when given as input a set of player values, outputs a c-approximation to the social welfare. We assume w.l.o.g[8] that ALG outputs a pareto efficient outcome, i.e. that there does not exist a loser that can be added to the set of winners. The Iterative Wrapper Mechanism is a simple wrapper to ALG: a vector of player values, initialized to $v^0 = \vec{1}$, is iteratively handed in as input to ALG, who, in return, outputs a set of winners W_{j+1} (where j is the iteration number). If the new allocation W_{j+1} has a value lower than the previous allocation W_j, we keep the previous allocation (step 5). Every loser is then required to either double his value or to permanently retire (this is denoted by setting $v_i^j = 0$). A retired player will not be able to win any bundle. This is iterated until all non-retired players are declared winners by ALG. These are the winners of the mechanism. Each winner pays his last bid, v_i^J, where J denotes the total number of iterations.

PROPOSITION 2.1. *It is a dominant strategy of any player to increase his reported value when asked, as long as the increased value is lower than his true value $\bar{v}_i$.*

Proof. If player i bids up to some value larger than $\bar{v}_i$ his utility will be non-positive in any case, which can be improved by retiring at the last value not larger than $\bar{v}_i$. If i retires at a value smaller than $\bar{v}_i/2$, then the mechanism ensures that he will not receive any item, hence his utility will be zero, while if i continues to bid and retires at the largest value smaller than $\bar{v}_i$, his utility is non negative. ∎

This implies that if player i retires at iteration j then $\bar{v}_i/2 \leq v_i^j \leq \bar{v}_i$.

In order to analyze the approximation bounds of this mechanism, the crucial point is that the number of iterations is low, independent of the number of players:

LEMMA 2.1. *If all players are single minded then the number of iterations is at most $2 \cdot \log \bar{v}_{max} + 1$.*

Proof. Suppose by contradiction that at iteration $2 \log \bar{v}_{max} + 1$ there exists a loser, i_1, who is willing to increase his value. This implies that there exists a winner, i_2, such that his desired bundle intersects the desired bundle of i_1. Hence in every previous iteration at least one of them was a loser, and doubled his value. Since each player doubles his value at most v_{max} times before retiring (in the dominant strategy), and since they doubled their value $2 \log \bar{v}_{max}$ times, the only possibility is that both i_1 and i_2 have value v_{max} at iteration $2 \log \bar{v}_{max} + 1$. But then this contradicts the assumption that i_1 is willing to increase his value in this iteration.

This also implies that if ALG has polynomial running time then the Iterative Wrapper has polynomial running time. For the case of single-minded players, not much work is left to show that the mechanism almost preserves the original approximation ratio:

Notation: Throughout the paper we use the following notations. If W is a set of players that can be simultaneously satisfied[9], we define $v(W, v) = \sum_{i \in W} v_i$. For a general set of players X, we define $OPT(X, v)$ to be the maximal $v(W, v)$ over all sets $W \subseteq X$ of players that can be jointly satisfied.

THEOREM 2.1. *Given any c-approximation algorithm for single-minded CAs, the Iterative Wrapper obtains an $O(\log \bar{v}_{max} \cdot c)$-approximation in dominant strategies for known single minded players, in polynomial time.*

Proof. We show that in every iteration, the winners are a $(2 \cdot c)$-approximation to the players that retire. Using the bound on the maximal number of iterations (Lemma 2.1), the approximation follows. Formally, for any $1 \leq j \leq J$, let R_j be the set of players that retired at iteration j. Note that $\cup_{j=1}^{J} R_j$ is exactly the set of all losing players, and for any j, $OPT(R_j, v^j) \leq c \cdot v(W_j, v^j) \leq$

to a higher value, fixing the strategies of the others.

[8]This is not necessarily w.l.o.g for non single minded CA domains.

[9]A player is satisfied by some allocation if he receives one of his desired bundles in that allocation.

$c \cdot v(W_J, \bar{v})$, where the last inequality holds by step 5 (improvement of the allocation). For any player $i \in R_j$ we have that $v_i^j \geq \bar{v}_i/2$, thus $\frac{1}{2}OPT(R_j, \bar{v}) \leq OPT(R_j, v^j)$, and therefore $OPT(R_j, \bar{v}) \leq 2 \cdot c \cdot v(W_J, \bar{v})$. We can now bound the optimal value of the entire set of players $N = W_J \cup (\cup_{j=1}^{J} R_j)$:

$$OPT(N, \bar{v}) = OPT(W_J \cup (\cup_{j=1}^{J} R_j), \bar{v}) \leq$$
$$OPT(W_J, \bar{v}) + \sum_{j=1}^{J} OPT(R_j, \bar{v}) \leq$$
$$v(W_J, \bar{v}) + J \cdot 2 \cdot c \cdot v(W_J, \bar{v})$$

Thus, $OPT(N, \bar{v}) \leq (J \cdot 2 \cdot c + 1) \cdot v(W_J, \bar{v})$. Since $J \leq 2 \cdot \log \bar{v}_{max} + 1$, the theorem follows. ∎

Remarks: (1) For optimizing the constants one needs to multiply a loser's value by e instead of by 2. For ease of notation we use the constant 2. (2) The analysis here is tight, see details in the full paper. (3) The proof actually shows that the mechanism obtains a $(2 \cdot c \cdot J + 1)$-approximation for *any* algorithm in *any* single-value domain, if the number of iterations is J. For known single minded players J is bounded, hence the result.

In addition to being the framework for our undominated strategies implementation, our construction is the first general deterministic technique to convert any algorithm to a dominant strategy mechanism, i.e. to convert any given algorithm to be value-monotonic. Most useful approximation techniques (e.g. LP rounding, or taking the maximum over several algorithms) do not generally yield value monotonicity, and our method converts such techniques to strategic dominant strategy mechanisms (with some additional approximation loss). The method can also be used to create dominant-strategy mechanisms from algorithms based on heuristics (algorithms which preform well on many instances, but with no worse case guarantee). In particular, it enables us to pick the best result over a polynomial number of heuristics and approximation algorithms (by a MAX operator), enjoying the benefits of all methods (with a small approximation loss), while having dominant strategies.

3 The Generalization to Single-Value Players

The Iterative Wrapper is the basis of our mechanism for the case of single-value players. In this case, every player i is multi-minded and desires one bundle out of a *set* of desired bundles $\bar{S}_i$ (minimal to containment). If i will receive any $s \in \bar{S}_i$, or any t that contains such an s, he will obtain a value $\bar{v}_i$. We denote such a player by $(\bar{v}_i, \bar{S}_i)$. We assume that both the value and the desired set of bundles are private information. Since $\bar{S}_i$ may be of size exponential in m, we need to assume that an oracle access is provided. The specific query type that we use is detailed below.

The Iterative Wrapper for multi-minded players, formally defined below in Def. 3.1, is a generalization of the construction for single minded players from above: given a procedure that outputs an approximately optimal allocation, we use the iterative wrapper in order to turn it to an algorithmic implementation, which is also an implementation in undominated strategies. Two differences are now introduced since players are multi-minded, and we do not know their desired bundles: First, a player has an "active bundle", s_i^j, which contains all possible items that a player may receive in the end of the auction. It is initialized to the entire set of items, Ω, and is gradually shrank by the given procedure. Second, the given procedure now interacts with the players in order to decide how to shrink the active bundle, in each step j, from s_i^j to s_i^{j+1}. We first define the Iterative Wrapper, and then the 1-CA-SUB procedure which it uses as the sub-procedure.

Definition 3.1. (The General Iterative Wrapper) *Let the set of winners in iteration j be $W_j = \emptyset$, and the set of retired players up to iteration j be $L_j = \emptyset$. Set the iteration counter to $j = 0$. For every player i, initialize i's value to $v_i^0 = 1$, and i's active bundle to $s_i^0 = \Omega$. Perform:*

While $(W_j \cup L_j \neq N)$:

1. *$(W_{j+1}, s^{j+1}) \leftarrow 1 - CA - SUB(v^j, s^j, W_j)$.*
2. *For any $i \notin W_{j+1}$, i chooses whether to double his value ($v_i^{j+1} = 2v_i^j$) or to retire ($v_i^j = 0$).*
3. *Update retired players ($L_{j+1} = \{i \in N | v_i^{j+1} = 0\}$), and increase the iteration counter $j = j + 1$.*

Outcome: *Let $J = j$ denote the total number of iterations. Each winner $i \in W_J$ receives his final active bundle s_i^J and pays his final bid v_i^J. All other players lose, do not receive anything, and pay* 0.

The 1-CA-SUB is essentially a modification of the 1-CA algorithm of Mu'alem and Nisan [16]. While the original 1-CA is truthful only for known single-minded players, our wrapping technique and the switch to algorithmic implementation make it suitable for unknown multi-minded players, with a loss of $O(log^2 v_{max})$ in the approximation factor.

The 1-CA-SUB procedure is given in Def. 3.2 below. The procedure receives the current player values and the previous allocation W_{j-1}, and constructs two allocations, MAX_j and $GREEDY_j$, by iterating over all

non-retired players in descending order of values. Every player is first given a possibility to shrink his active bundle (step 2a). If the player does choose to shrink it, the new bundle must have size at most $\sqrt{m}$, and must be contained in the set of items currently not allocated to anyone by the $GREEDY_j$ allocation. Then (step 2b) the new active set of the player is added to any of the allocations $W_{j-1}, MAX_j, GREEDY_j$ for which adding it is valid. This also maintains the pareto property. Once all players have been considered, the allocation with maximal value out of the three, and the updated active sets, are outputted. Recall that after the procedure completes, each losing player is required by the wrapper to either double his value or retire.

Notation: For a set of players X and an allocation $s^j = (s_1^j, ..., s_n^j)$, we denote by $Free(X, s^j)$ the set of goods that are not allocated to any player in X, i.e. the items that are not in $\cup_{i \in X} s_i^j$.

DEFINITION 3.2. (THE 1-CA-SUB PROCEDURE) *Receive as input a vector of values v^j, a vector of bundles s^j (with one element for each player), and an allocation W_{j-1} which is valid w.r.t. s^j. Then perform:*

1. *Let $MAX_j = argmax_{i \in N}\{ v_i^j \}$, $GREEDY_j = \emptyset$.*
2. *Go over players with $v_i^j > 0$ in descending order of values. Let player i be the current player, and,*
   (a) **Shrinking the active set:** *If $i \notin W_{j-1}$ allow him to pick a bundle $s_i^{j+1} \subseteq Free(GREEDY_j, s^{j+1}) \cap s_i^j$ such that $|s_i^{j+1}| \leq \sqrt{m}$. In any other case ($i \in W_{j-1}$ or i does not pick) set $s_i^{j+1} = s_i^j$.*
   (b) **Updating the current winners:** *If $|s_i^{j+1}| \leq \sqrt{m}$, add i to any of the allocations $W \in \{W_{j-1}, MAX_j, GREEDY_j\}$ for which $s_i^{j+1} \subseteq Free(W, s^{j+1})$.*
3. **Output:** *The vector of bundles s^{j+1} and an allocation $W \in \{W_{j-1}, MAX_j, GREEDY_j\}$ with maximal value of $\sum_{i \in W} v_i^j$.*

Note that the basic assumption is that a player can answer the following query in polynomial time: Given a set of items, return a desired bundle contained in this set, of size at most $|\sqrt{m}|$, if such a bundle exists. Our query in step 2a presents to a player the set $Free(GREEDY_j, s^{j+1}) \cap s_i^j$. In the Edge-Disjoint-Paths problem, this query can clearly be answered in polynomial time. We now state our main result:

THEOREM 3.1. *The Iterative Wrapper with the 1-CA-SUB is an algorithmic implementation of an $O(log^2(\bar{v}_{max}) \cdot \sqrt{m})$-approximation.*

In addition, as explained above, the Iterative Wrapper with the 1-CA-SUB is also an implementation in undominated strategies of the same approximation.

We give the full analysis in Section 4 below, but since the transition to algorithmic implementation is a bit subtle, we first give an intuition. Consider the case of EDP, being solved with the Iterative Wrapper and the 1-CA-SUB. Each edge is an item, and the entire set of edges is to be allocated to the different source-target players. As we go over players, e.g. in the first iteration, each player is asked to specify a desired subset of edges. Suppose that the player has two edge disjoint paths in the original graph, but some edges in the first path are not available when the player is reached. At this point, the player may shrink his active subset, removing the currently occupied edges, or he may choose not to shrink the active set. By the first alternative, he avoids the need to double his price, but this also limits his choices in future iterations, as he will no longer be able to win the other path. The second option will cause him to double his payment. Losing players face such a decision in every iteration. A mechanism with dominant strategies implies that the player has a specific optimal choice, no matter what the other players choose. In our case, we do not rule out either of these choices, but rather show that the approximation ratio will be maintained *in either case*. All we need in order to guarantee the approximation is that in the iteration that the player actually retires, he discloses one of his true paths if such a path is still available.

4 Analysis

We analyze the above mechanism in three parts. We first define the set D in Section 4.1. We then analyze the approximation that these strategies produce. We give general conditions on the procedure used inside the wrapper, that lead to an approximation, in Section 4.2, and show that the 1-CA-SUB indeed satisfies these conditions in Section 4.3.

4.1 Analysis of Strategies

We first characterize the set D of strategies that lead to a good approximation. For this we need the following intuitive property, which involves the behavior of players that are about to retire at the current iteration, if they will lose:

DEFINITION 4.1. (LOSER-IF-SILENT) *Player i is a "loser-if-silent" at iteration j if, when he is asked to shrink his bundle at step 2a of the 1-CA-SUB procedure, all the following hold:*

1. *(Retires if losing) $v_i^j \geq \bar{v}_i/2$.*
2. *(Not a possible winner) $i \notin W_{j-1}$ and $i \notin MAX_j$.*

3. *(Remains a loser after Pareto stage)* $s_i^j \cap (\cup_{i \in W_{j-1}} s_i) \neq \emptyset$ *and* $s_i^j \cap (\cup_{i \in MAX_j} s_i) \neq \emptyset$.

In other words, i is loser-if-silent if he will lose unless he will shrink his active set when he is approached at step $2a$, no matter what the others will declare.

DEFINITION 4.2. (THE SET D) *Let D to be all strategies that satisfy the following, in every iteration j:*

1. s_i^j *contains some desired bundle, and* $\bar{v}_i \geq v_i^j$.
2. *If i retires at iteration j then* $v_i^j \geq \bar{v}_i/2$.
3. *If player i is a "loser-if-silent", then i will declare some desired bundle s_i^{j+1} that satisfies the conditions of step $2a$, if such a bundle exists.*

LEMMA 4.1. *There exists a poly-time algorithm to find a strategy $st' \in D$ that dominates a given strategy st.*

Proof. Consider the following algorithm that checks, at each decision point of the strategy st, if the three conditions of Def. 4.2 are satisfied. More explicitly, when a player is asked to shrink his active subset, check if the new set contains some desired bundle. If not, change st so that the player will not shrink his the active set. If the player is loser-if-silent, additionally check if the third property is satisfied (this can be done in polynomial time by our assumption on the query capabilities). If not, shrink the active set to some arbitrary valid desired set that does satisfy the requirement, if such a set exists. Since a player receives his last active bundle, this will change the utility from a zero utility to a non-negative utility, hence this changes creates a strategy that dominates the former. When st decides to double the value v_i^j, check if the new value is lower than the true value. If not, change st so that it will instead decide to retire. When st decides to retire and not to double the value, check if twice the value is higher than the true value. If not, change st so that it will instead decide to double the value (after that, shrink the active set to some arbitrary and valid desired set, and keep on in the straight-forward way). Since a winner pays his last reported value, and can only gain by not retiring at a lower value (has no risk if he stays active), this again creates a strategy that dominates the former one. ∎

The next two sections show that if all players play strategies in D (i.e. satisfy these three conditions) then the approximation holds.

4.2 Analysis of the Iterative Wrapper We next give conditions on the sub-procedure that enable the Iterative Wrapper to obtain good performance.

Notation: Given a set of players, W, and a vector of player subsets, $s = (s_1, ..., s_n)$, we say that (W, s) is a valid allocation if $s_i \cap s_{i'} = \emptyset$ for any $i, i' \in W$.

DEFINITION 4.3. (A PROPER PROCEDURE) *Fix a procedure that receives as input (1) the players' current values $\tilde{v}$, (2) the players' active sub-sets s, and (3) a subset of the players, W, such that (W, s) is a valid allocation. The output is a set of winning players, W', and a set of active sets, s', such that (W', s') is a valid allocation. This procedure is termed a* "proper procedure" *if it satisfies:*

1. *(Pareto) For any $i \notin W'$, $s'_i \cap (\cup_{l \in W'} s'_l) \neq \emptyset$.*
2. *(Shrinking sets) For every player i, $s'_i \subseteq s_i$.*
3. *(Improvement) $v(W', \tilde{v}) \geq v(W, \tilde{v})$.*
4. *(First time shrink) For any $i_1, i_2 \in \{i : |s_i| = m \ \& \ |s'_i| < m\}$, $s'_{i_1} \cap s'_{i_2} = \emptyset$.*

The first two properties ensure that the number of iterations will be at most $2 \log \bar{v}_{max} + 1$, using arguments similar to the single minded case (Lemma 2.1). The "first time shrink" condition implies that all players that first restrict their bundle in the same iteration are disjoint. It will enable us to bound the value of all these players together by the value of the winners.

LEMMA 4.2. *Given any proper procedure, the number of iterations of the General Iterative Wrapper of Definition 3.1 is at most* $2 \log v_{max} + 1$.

To ensure the approximation, we additionally require a "local approximation": the procedure is not able to produce a global approximation with respect to all the true desired bundles of a player, as it is limited to allocate to player i a subset that is contained in s_i^j (his active bundle). Instead, a "local" approximation guarantees the quality of the outcome only with respect to players that retire at iteration j, and, furthermore, only with respect to their desired bundles that are contained in their active bundle. Our analysis shows that ignoring all their other subsets incurs an additional approximation loss of at most $O(\log(\bar{v}_{max}))$.

More formally, recall that a single value player is a pair (v_i, S_i), where v_i is the value of any desired bundle $s \in S_i$. Let $\bar{S}_i|_{s_i^j} = \{s \in \bar{S}_i \mid s \subseteq s_i^j\}$, where s_i^j is the active set of player i at iteration j. In other words, $\bar{S}_i|_{s_i^j}$ includes all the bundles in $\bar{S}_i$ contained in s_i^j. For any iteration $1 \leq j \leq J$, define:

$$R_j = \{ \ (v_i^j, \bar{S}_i|_{s_i^j}) \mid \ i \text{ retired at iteration } j \ \}, \quad (4.1)$$

i.e. for every player i that retired at iteration j the set R_j contains a single-value player, with value v_i^j, that desires some of the true bundles that i desires – exactly those that are contained in s_i^j.

Notation: Let us restate our notion of $OPT(\cdot)$ for the case where we have single value players. Let X be an instance of players and their valuation (value plus desired bundles). Then $OPT(X)$ is the maximal value of an allocation of the items to the players of X. Thus for example $OPT(R_j) = \max_{\text{all allocations } (s_1,\ldots,s_n) \text{ s.t.} s_i \in \bar{S}_i|_{s_i^j}} \{\sum_{i:\ s_i \neq \emptyset} v_i^j\}$.

DEFINITION 4.4. (LOCAL APPROXIMATION) *A proper procedure is a $\tilde{c}$-local-approximation with respect to a set of strategies D, if, for any combination of strategies in D, and any iteration j,*

1. *(Algorithmic approximation) $OPT(R_j) \leq \tilde{c} \cdot v(W_j, v_i^j)$*
2. *(Value bounds) s_i^j contains some desired bundle, $\bar{v}_i \geq v_i^j$, and, if i retires at j then $\bar{v}_i \leq 2 \cdot v_i^j$.*

THEOREM 4.1. *Given any proper procedure which is a $\tilde{c}$ local approximation with respect to D, the General Iterative Wrapper obtains an $O(\log^2(\bar{v}_{max}) \cdot \tilde{c})$ approximation, for any profile of strategies in D.*

Proof. Let $P = \{\ (\bar{v}_i, \bar{S}_i)\ :$ i lost, and $|s_i^J| < m\ \}$, i.e. P contains all the losing players who shrank their bundles, with their true values and their true set of desired bundles. Let $\bar{R}_j = \{\ (\bar{v}_i, \bar{S}_i|_{s_i^j})\ :$ i retired at iteration j $\}$, i.e. $\bar{R}_j$ contains all losing players that retired at iteration j, with their true values, but only with the desired bundles that are contained in their last active set (note that it is different than R_j, since players in R_j have values v_i^j). Let $\bar{R} = \cup_{j=1}^{J} \bar{R}_j$. Note that players in P appear with all their true desired bundles while players in $\bar{R}$ appear with only part of their desired bundles. The first claim shows that ignoring the bundles that are in P but not in $\bar{R}$ incurs only a small loss:

CLAIM 1. $OPT(P) \leq J \cdot OPT(\bar{R})$.

Proof. Define P_j to be all players in P that first shrank their bundle at iteration j. By the "first time shrink" condition of the proper procedure and by the fact that active bundles only shrink, we have that $s_{i_1}^j \cap s_{i_2}^j = \emptyset$ for every $i_1, i_2 \in P_j$. Therefore $OPT(\bar{R}) \geq \sum_{i \in P_j} \bar{v}_i$: every player i in P_j corresponds to a player in $\bar{R}$, and all these players have disjoint bundles, each contained in s_i^j. We also trivially have $OPT(P_j) \leq \sum_{i \in P_j} \bar{v}_i$. Thus we have that for any j, $OPT(P_j) \leq OPT(\bar{R})$. Hence $OPT(P) \leq \sum_j OPT(P_j) \leq J \cdot OPT(\bar{R})$. ∎

On the other hand, the optimal allocation of the "truncated players" $\bar{R}$ is bounded by the value of the winners:

CLAIM 2. $OPT(\bar{R}) \leq 2 \cdot J \cdot \tilde{c} \cdot \sum_{i \in W_J} \bar{v}_i$.

Proof. We first claim that $OPT(\bar{R}_j) \leq 2 \cdot \tilde{c} \cdot \sum_{i \in W_J} \bar{v}_i$: since for any player that retires at iteration j, we have $v_i^j \geq \bar{v}_i/2$ by the value bounds assumption, it follows that $OPT(\bar{R}_j) \leq 2 \cdot OPT(R_j)$. Since $OPT(R_j) \leq \tilde{c} \cdot \sum_{i \in W_j} v_i^j$ by the algorithmic approximation assumption, and $\sum_{i \in W_j} v_i^j \leq \sum_{i \in W_{j+1}} v_i^{j+1}$ by the improvement assumption, we get that $OPT(\bar{R}_j) \leq 2 \cdot c \cdot \sum_{i \in W_J} v_i^J$. Since $\bar{v}_i \geq v_i^J$ (again by the value bounds assumption) we conclude that $OPT(\bar{R}_j) \leq 2 \cdot \tilde{c} \cdot \sum_{i \in W_J} \bar{v}_i$.

Hence $OPT(\bar{R}) \leq \sum_{j=1}^{J} OPT(\bar{R}_j) \leq J \cdot 2 \cdot \tilde{c} \cdot \sum_{i \in W_J} \bar{v}_i$, and the claim follows. ∎

The theorem now follows almost immediately. First notice that the true input is contained in $P \cup \bar{R} \cup W_J$: all retiring players belong to $\bar{R} \cup P$ (if a player shrank his bundle then he belongs to P with all his true bundles, and if a player did not shrink his bundle at all then he belongs to $\bar{R}$ with all his true bundles) and all non-retiring players belong to W_J. From the above we have $OPT(P \cup \bar{R}) \leq OPT(P) + OPT(\bar{R}) \leq J \cdot OPT(\bar{R}) + OPT(\bar{R}) \leq 4 \cdot J^2 \cdot \tilde{c} \cdot \sum_{i \in W_J} \bar{v}_i^J$. Since s_i^J contain some desired bundle of player i, we have that $OPT(W_J) = \sum_{i \in W_J} \bar{v}_i$. Thus we get that $OPT(P \cup \bar{R} \cup W_J) \leq 5 \cdot J^2 \cdot \tilde{c} \cdot \sum_{i \in W_J} \bar{v}_i^J$. Since $J \leq 2 \cdot \log(\bar{v}_{max}) + 1$ by Lemma 4.2, the theorem follows. ∎

Remark: In the full paper we show, for the "known" case, how to convert any given c-approximation to a proper procedure which is a c-local approximation (assuming that the type space "includes single minded players", for details and definitions see the full paper).

4.3 Analysis of the 1-CA-SUB It now only remains to show that the 1-CA-SUB is a proper procedure and a local approximation.

LEMMA 4.3. *The 1-CA-SUB is a proper procedure.*

Proof. Step 2*b* verifies that all three allocations $W_{j-1}, MAX_j, GREEDY_j$ are pareto with respect to the active bundles of the players. Hence the pareto requirement is satisfied. Step 2*a* explicitly verifies the shrinking sets requirement by requiring the new active set to be contained in the old one. The Improvement requirement is taken care of by the output step, where it is explicitly verified that the final allocation has value not less than W_{j-1}, the previous allocation. The First-time-shrink requirement holds since any two players that shrink their subsets belong to the $GREEDY_j$ allocation, hence their active subsets are disjoint. ∎

LEMMA 4.4. *The 1-CA-SUB procedure is an $O(\sqrt{m})$-local-approximation with respect to set D of Def. 4.2.*

Proof. The "Value bounds" requirement is immediate from the definition of D. For the algorithmic approximation, fix an iteration j. Recall that we need to show that $OPT(R_j) \leq \tilde{c} \cdot \sum_{i \in W_j} v_i^j$, where $R_j = \{(v_i^j, \bar{S}_i|_{s_i^j}) \mid i \text{ retired at iteration } j\}$, and $\tilde{c} = O(\sqrt{m})$. For any $W \in \{MAX_j, GREEDY_j, W_{j-1}\}$, $\sum_{i \in W} v_i^j \leq \sum_{i \in W_j} v_i^j$ by construction. We next bound the value of players in R_j that are not in one of these three sets. Explicitly, let $X = \{(v_i^j, \bar{S}_i|_{s_i^j}) \mid i \notin MAX_j \cup GREEDY_j \cup W_{j-1}$ and retired at iteration $j\}$.

CLAIM 3. $OPT(X) \leq 2 \cdot \sqrt{m} \cdot \sum_{i \in W_j} v_i^j$.

Proof. Let X_M contain all players that receive a bundle of size at least $\sqrt{m}$ in $OPT(X)$. We will show a mapping $f : X_M \to MAX_j$ such that f is $\sqrt{m}$ to 1, and for any $i \in X_M$, $v_i^j \leq v_{f(i)}^j$: simply map i to the player with highest value in MAX_j. There can be at most $\sqrt{m}$ players in X_M since each player receives a bundle of size at least $\sqrt{m}$ and these bundles do not intersect. Thus the first property of f holds. The second property holds since the player with maximal value in MAX_j has the highest value among all players (according to v^j).

Let X_G contain all players that receive a bundle of size at most $\sqrt{m}$ in $OPT(X)$. We show a mapping $f : X_G \to GREEDY_j$ such that f is $\sqrt{m}$ to 1, and for any $i \in X_G$, $v_i^j \leq v_{f(i)}^j$. Let s_i^* be the bundle that i received in $OPT(X)$. Since $i \notin W_{j-1} \cup MAX_j$, it follows that i is loser-if-silent. Since he did not enter $GREEDY_j$, there exists $i' \in GREEDY_j$ such that $s_{i'}^j \cap s_i^* \neq \emptyset$ and $v_i^j \leq v_{i'}^j$. We map i to i'. It remains to show that f is $\sqrt{m}$ to 1: for any i_1, i_2 that were mapped to i' we have that $s_{i_1}^* \cap s_{i_2}^* = \emptyset$ since both belong to $OPT(X)$. Since the size of $s_{i'}^j$ is at most $\sqrt{m}$ it follows that at most $\sqrt{m}$ players can be mapped to i'.

Therefore $OPT(X) = v(X_M, v^j) + v(X_G, v^j) \leq v(MAX_j, v^j) + v(GREEDY_j, v^j) \leq 2\sqrt{m} \cdot v(W_j, v^j)$, and the claim follows. ∎

From this we conclude that $OPT(R_j) \leq \sum_{i \in W_{j-1} \cup MAX_j \cup GREEDY_j} v_i^j + OPT(X) \leq (3 + 2 \cdot \sqrt{m}) \sum_{i \in W_j} v_i^j$, and the Lemma follows. ∎

Acknowledgments

We are grateful to Noam Nisan for his invaluable help. We also thank Liad Blumrosen, Edith Elkind, Jason Hartline, Daniel Lehmann, and Chaitanya Swamy, for many helpful comments.

References

[1] K. Akcoglu, J. Aspens, B. DasGupta, and M. Kao. An opportunity-cost algorithm for combinatorial auctions. In E. J. Kontoghiorghes, B. Rustem, and S. Siokos, editors, *Applied Optimization: Computational Methods in Decision-Making, Economics, and Finance*. Kluwer Academic, 2002.

[2] A. Archer and E. Tardos. Truthful mechanisms for one-parameter agents. In *FOCS*, 2001.

[3] B. Awerbuch, Y. Azar, and A. Meyerson. Reducing truth-telling online mechanisms to online optimization. In *STOC*, 2003.

[4] M. Babaioff, R. Lavi, and E. Pavlov. Impersonation-based mechanisms, 2005. Working paper.

[5] M. Babaioff, R. Lavi, and E. Pavlov. Mechanism design for single-value domains. In *AAAI*, 2005.

[6] Y. Bartal, R. Gonen, and N. Nisan. Incentive compatible multi-unit combinatorial auctions. In *TARK*, 2003.

[7] L. Blumrosen and N. Nisan. On the computational power of iterative auctions. In *ACM-EC*, 2005.

[8] P. Briest, P. Krysta, and B. Vocking. Approximation techniques for utilitarian mechanism design. In *STOC*, 2005.

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

[10] T. Groves. Incentives in teams. *Econometrica*, 1973.

[11] V. Guruswami, S. Khanna, R. Rajaraman, B. Shepherd, and M. Yannakakis. Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems. *Journal of Computer and System Sciences*, 67(3):473–496, 2003.

[12] M. O. Jackson. Implementation in undominated strategies: A look at bounded mechanisms. *Review of Economic Studies*, 1992.

[13] R. Lavi, A. Mu'alem, and N. Nisan. Towards a characterization of truthful combinatorial auctions. In *FOCS*, 2003.

[14] R. Lavi and C. Swamy. Truthful and near optimal mechanism design via linear programming. In *FOCS*, 2005.

[15] D. Lehmann, L. O'Callaghan, and Y. Shoham. Truth revelation in approximately efficient combinatorial auctions. *Journal of the ACM*, 49(5):1–26, 2002.

[16] A. Mu'alem and N. Nisan. Truthful approximation mechanisms for restricted combinatorial auctions. In AAAI 2002.

[17] N. Nisan. The communication complexity of approximate set packing and covering. In *ICALP*, 2002.

[18] N. Nisan and A. Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 2001.

[19] Y. Shoham P. Cramton and R. Steinberg. *Combinatorial Auctions*. The MIT press, 2005.

[20] D. Parkes. Iterative combinatorial auctions. In P. Cramton, Y. Shoham, and R. Steinberg, editors, *Combinatorial Auctions*. The MIT press, 2005.

[21] W. Vickrey. Counterspeculation, auctions and competitive sealed tenders. *Journal of Finance*, 1961.