Incentive Compatible Multi Unit Combinatorial Auctions

Yair Bartal
School of Computer Science and Engineering,
Hebrew University, Jerusalem 91904, Israel
yair@cs.huji.ac.il

Rica Gonen
School of Computer Science and Engineering
Hebrew University, Jerusalem 91904, Israel
rgonen@cs.huji.ac.il

Noam Nisan
School of Computer Science and Engineering,
Hebrew University, Jerusalem 91904, Israel
noam@cs.huji.ac.il

February 9, 2003

Abstract

The problem we deal with in this paper is a multi-unit combinatorial auction: there are n types of goods for sale, and for each good there is some fixed number of units. We focus on the case where each bidder desires a relatively small number of units of each good. In particular, this includes the case where each good has exactly k units, and each bidder desires at most a single unit of each good.

We provide incentive compatible mechanisms for combinatorial auctions, for the general case in which bidders are not limited to be *single minded*.

The mechanisms we give have approximation ratios close to the best possible for both the on-line and off-line scenarios. This is the first result where non-VCG mechanisms are derived for non-single minded bidders for a natural model of combinatorial auctions.

1 Introduction

1.1 Combinatorial Auctions

In this paper we study *multi-unit* combinatorial auctions. In a combinatorial auction n non-identical items are auctioned simultaneously to m bidders; in a multi-unit combinatorial auction for each good $i \in \{1, \ldots, n\}$, there is a number k_i of identical units. The combinatorial nature of the auctions comes from the fact that bidders value bundles of items. A *bundle* of items is a vector $(d_1...d_n)$, where $0 \leq d_i \leq k_i$ is the number of items of good i in the bundle. Specifically, each bidder j has a *valuation* function v_j that assigns a non-negative value for each *bundle* of items, $v_j : \{0...k_1\} \times ... \times \{0...k_n\} \rightarrow R^+$. The goal of the auction is to maximize the total social welfare (surplus): find an allocation that partitions the available items into bundles $S_1...S_m$ that maximizes $\sum_j v_j(S_j)$. Most of the previous work on combinatorial auctions deals with the case of single-unit goods, i.e., when for all i, $k_i = 1$.

The combinatorial auction problem has recently received much attention due both to its many applications and to its wide expressibility powers that allow it to represent a wide range of situations that combine algorithmic issues with incentive issues – a combination that seems to be central to many "Internet-Computing" tasks. The reader is referred e.g. to [17, 20, 23] for a general overview, and to, e.g., [18, 27, 29, 25] and the many references therein for an introduction combinatorial auctions.

The key challenge in combinatorial auctions comes from a combination of the computational intractability of the general problem together with "incentive compatibility" requirements. Finding the optimal allocation in a combinatorial auction (with reasonable encoding of the valuations) is computationally intractable (even to approximate within $O(n^{\frac{1}{2}-\epsilon})$ [10, 27]). However, many algorithmic techniques and heuristics may be applied to the problem yielding algorithms that either work for interesting special cases or that work reasonably well (from an experimental point of view) for hundreds and even thousands of items (e.g. [6, 7, 30, 29]).

The notion of incentive compatibility takes into account the fact that the auctioneer does not really have access to the input, i.e. to the valuations of the bidders. Rather, the auctioneer only has access to the bids, which may be manipulated strategically by the bidders. An auction is called incentive compatible if bidders have no incentive to "lie" about their valuation, i.e. if rational bidders always bid their valuation – in which case the auctioneer at least has the proper input according to which he may allocate the items. (Formal definitions are postponed to section 2.) The key technique used for designing incentive compatible mechanisms is the, so called, VCG-mechanism [28, 5, 9] that first determines the optimal allocation and then charges carefully chosen payments for the allocated bundles of items.

Here comes the key clash between incentive compatibility and computational complexity: finding the optimal allocation is computationally intractable and hence the VCG-mechanism is intractable. Furthermore, the VCG technique really relies on the fact that the allocation is optimal and using a non-optimal allocation together with the VCG payment scheme does not yield incentive compatible mechanisms. The problem was first noticed in [14, 20] was shown to be essentially universal in [21], and was further studied in [12].

The only positive results known are for very restricted classes of bidders, "single-minded" bidders [14], those that desire only a single subset of items. For the special case of single-minded combinatorial auctions several polynomial-time incentive compatible mechanisms are known [14, 16, 1], including one that achieves an $O(\sqrt{n})$ approximation (the best possible). Nothing is known for the general case, and to date, no non-optimal incentive compatible combinatorial auction mechanism is known for general bidders (where optimality may be within any fixed family of allocations [21, 12]).

This lack of knowledge is part of a general problem in mechanism design. The only non-VCG mechanisms known to date for any type of mechanism design problem are for those with an essentially single dimensional space of valuations – as is essentially the case with single-minded bidders. On the other hand, it is known that for problems where the valuation space is rich enough ("complete" over at least 3 outcomes) indeed only VCG mechanisms exist [8, 24]. Nothing is known for the intermediate cases where the valuation space is more than single dimensional but is not complete. Most mechanism design problems that arise from computational scenarios lie in this intermediate range.

In this paper we present the *first* such incentive compatible mechanism for a natural subset of muti-unit combinatorial auction problems. The mechanism runs in polynomial time even though finding the optimal solution is NP-hard. The approximation ratio of this mechanism is nearly as good as can be achieved in polynomial time. Following our result[1] [2] presented an online randomized version of such incentive compatible mechanism. Their result depends on the ratio of highest and lowest valuation. As apposed to the case in [2] our general mechanism does not depend on that ratio and apply deterministic techniques.

1.2 Multi-Unit Combinatorial Auctions with Bounded Demand

The problem we deal with in this paper is a multi-unit combinatorial auction: there are n types of goods for sale, and for each i there are k_i units of good i. We are interested in the case where each bidder desires a relatively small number of units of each good. Formally let the set of bundles U $U = \{0, \ldots, k_1\} \times \{0, \ldots, k_2\} \times \{0, \ldots, k_n\}$. Each bidder has a valuation function $v_j : U \rightarrow R^+$, and the aim is to maximize $\sum_j v_j(S_j)$ subject to the constraint that for each good i there exist at most k_i bundles S_j that have requirement for good i. We assume that there exists a lower bound θ and an upper bound Θ such that each bidder desires at most Θk_i units of good i and either no unit of good i or at least θk_i units of good i. The simplest case is where each type of good has exactly k units and each bidder desires at most a single unit of each good, $\theta = \Theta = 1/k$. We call this a combinatorial auction with k-duplicates.

This case was recently studied in [1] who developed an incentive compatible approximation

[1]Our work was presented at Dagstuhl Workshop: Electronic Mechanism Design, June 2002, Wardern, Germany.

mechanism for the *single-minded bidder* case. We derive an incentive compatible algorithm for the general non-single-minded case that achieves an approximation factor that is close to the best that can be achieved in polynomial time.

Our first step is to characterize incentive compatible mechanisms for combinatorial auctions (both multi-unit, and regular), for the general case that bidders are not limited to be *single minded*.

Theorem 1: A mechanism for combinatorial auctions is incentive compatible if and only if for every bidder j and every vector of bids of the other players v_{-j} it:

1. fixes a price $p_j(S)$ for every possible allocation S to bidder j, and whenever bidder j is allocated S his payment is $p_j(S)$. (Note that $p_j(S)$ does not depend on v_j.)

2. allocates to j the S that maximizes the value of $v_j(S) - p_j(S)$ over all allocations S (that can be allocated to j for any choice of v_j.)

Our main result is an efficient incentive compatible mechanism for multi-unit combinatorial auctions with bounded demand:

Theorem 3: There exists an incentive compatible mechanism for multi-unit combinatorial auctions with (θ, Θ)-bounded demand that is computationally efficient and achieves an $O\left(\frac{1}{\Theta}\left(\frac{n}{\theta}\right)^{\frac{\Theta}{1-2\Theta}}\right)$ approximation to the optimal allocation.

In the special case of k-duplicates, $\theta = \Theta = 1/k$, the approximation factor is $O\left(kn^{1/(k-2)}\right)$. It is worth noting that this bound becomes logarithmic for $k = O(\log n)$. This bound is close to being best possible in polynomial time as we prove, in theorem 4, that as a purely computational problem, a combinatorial auction with k duplicates is NP-hard to approximate to within a factor of $O(n^{1/(k+1)-\epsilon})$. This generalizes the known inapproximability result for combinatorial auctions ($k = 1$).

En-route to this mechanism we pass through an *online* incentive compatible mechanism, that is of independent interest. In the online case we must have some a priori bounds on the maximal valuation.

Theorem 2: There exists an *online* incentive compatible mechanism for multi-unit combinatorial auctions with (θ, Θ)-bounded demand that is computationally efficient, incentive compatible, and achieves an $O\left(\frac{1}{\Theta}\left(\frac{\rho n}{\theta}\right)^{\frac{\Theta}{1-\Theta}}\right)$ approximation to the optimal allocation, where ρ is the ratio of the maximum and minimum a priori known bounds on the maximal valuation.

Our mechanisms employ the basic technique of the network algorithms of [3] and [4] of having item prices that increase exponentially as units of the item are handed out. As opposed to the case in [3], our bidders are potentially interested in many possible allocations, and the choice between the possibilities is made in an incentive compatible manner. For the final (offline) algorithm we add yet another ingredient that resembles the notion of a 2nd price auction. The single-minded mechanism of [1] for the k-duplicates problem is completely different and uses randomized rounding of the LP-relaxation of the problem.

2 Definitions

Instead of discussing an auction where each good i has k_i indivisible units, It will be simpler to normalize by k_i, and assume that each good has 1 unit, but is divisible. (Our formalism will allow expressing the underlying indivisibility of each original unit.) Thus our problem is as follows: There are n goods, each has 1 divisible unit.

2.1 Valuations

A bundle of items is a vector of quantities in $U = [0,1]^n$. A valuation is a function v specifying how much each bundle of items is worth to a bidder. I.e. for each vector of quantities $\lambda = (\lambda^{(1)} ... \lambda^{(n)}) \in U$, $v(\lambda)$ is how much money is this player willing to pay to get $\lambda^{(i)}$ fraction of a unit of good i, for all i.

Definition 1 *(valuation) A valuation is a function $v : [0,1]^n \to R^+$ with the following properties:*

- *v is monotone in each coordinate (free disposal).*
- *$v(\vec{0}) = 0$ (normalization)*

In our formalism v need not be continuous. For instance in the combinatorial auction problem with k-duplicates the natural way to represent the indivisibility of good number 1 other than into fractions of size exactly $1/k$ is to have for all valuations $v(\alpha, ...) = v(0, ...)$ for all $\alpha < 1/k$.

We say that a valuation is (θ, Θ)-bounded if it is not interested in getting less than θ units of any item nor in getting more than Θ units of any item. Formally:

Definition 2 *((θ, Θ)-bounded valuation)A valuation v is (θ, Θ)-bounded if for all i and $\lambda^{(i)} < \theta$, we have that $v(\lambda^{(1)} ... \lambda^{(n)}) = v(\lambda^{(1)} ... \lambda^{(i-1)}, 0, \lambda^{(i+1)} ... \lambda^{(n)})$, and for all i and $\lambda^{(i)} > \Theta$, we have that $v(\lambda^{(1)} ... \lambda^{(n)}) = v(\lambda^{(1)} ... \lambda^{(i-1)}, \Theta, \lambda^{(i+1)} ... \lambda^{(n)})$.*

2.2 Representation of Valuations

In this paper our mechanisms use an abstract black-box representation of valuations. The natural black-box representation would be to output the value $v(\lambda^{(1)} ... \lambda^{(n)})$ on a query $(\lambda^{(1)} ... \lambda^{(n)})$. This is called a valuation oracle for v. It turns out that this is not enough; additionally we need the "demand oracle" representation of the valuation [15].

Definition 3 *(demand oracle) A demand oracle for valuation v accepts as input a vector of item prices $(p^{(1)}...p^{(n)})$ and outputs the demand for the items at these prices, i.e. it outputs the vector $\lambda = (\lambda^{(1)}...\lambda^{(n)})$ that maximizes the surplus $v(\lambda) - < \lambda, p >= v(\lambda^{(1)}...\lambda^{(n)}) - \sum_i \lambda^{(i)} p^{(i)}$.*

Notice that a demand oracle for a $((\theta, \Theta)$-bounded valuation always returns for every good i, $\theta \le \lambda^i \le \Theta$, or $\lambda^i = 0$.

In a concrete algorithmic implementation, the valuations will be given in some "bidding language", and our algorithms will work efficiently (in polynomial time) as long as the bidding language allows efficient computation of answers to valuation oracle and to demand oracle queries. In appendix A we shortly discuss the range of such languages. In particular note that these two types of oracle queries can be answered easily for the case that each bidder puts forward an arbitrary list of mutually exclusive bids for packages.

2.3 Allocations

Definition 4 *(allocation) An allocation is a collection of m non-negative vectors $\lambda_1...\lambda_m$, where $\lambda_j^{(i)}$ specifies the amount of good i that bidder j has received. An allocation is feasible if for all i, $\sum_j \lambda_j^{(i)} \le 1$.*

Definition 5 *(value of an allocation) The value of an allocation A is $V(A) = \sum_j v_j(\lambda_j)$. An allocation is optimal if it achieves the maximal value of any feasible allocation.*

2.4 Mechanisms

Definition 6 *(direct revelation mechanism) A direct revelation mechanism receives as input a vector of declared valuations $v_1...v_m$, and produces as output an allocation $\lambda_1...\lambda_m$ and a vector of payments $P_1...P_m$, where bidder j receives λ_j and pays P_j.*

Definition 7 *(Incentive compatibility) A direct revelation mechanism is incentive compatible if for every bidder j, every valuation v_j, all declarations of the other bidders v_{-j}, and all possible "false declarations" $v`_j$ we have that bidder $j$'s utility with bidding $v`_j$ is no more than his utility with bidding the truth v_j. I.e. Let λ_j and P_j be the mechanism's output with input (v_j, v_{-j}) and $\lambda`_j$ and $P`_j$ be the mechanism's output with input $(v`_j, v_{-j})$, then $v_j(\lambda_j) - P_j \ge v_j(\lambda`_j) - P`_j$.*

3 A Characterization of Incentive Compatible Mechanisms

Theorem 1 *A direct revelation mechanism is incentive compatible if and only if for every bidder j and every vector of bids of the other players v_{-j} it:*

1. *fixes a price $p_j(\lambda)$ for every possible allocation λ to bidder j, and whenever bidder j is allocated λ his payment is $p_j(\lambda)$. (Note that $p_j(\lambda)$ does not depend on v_j.)*

2. *allocates to j, λ that maximizes the value of $v_j(\lambda) - p_j(\lambda)$ over all λ that can be allocated to j (for any choice of v_j.)*

The proof follows the usual characterizations of incentive compatibility and appears in appendix B.

The simplest way to use this theorem for constructing incentive compatible mechanisms is to choose for each bidder j item prices $p = (p^{(1)}...p^{(n)})$, set $p_j(\lambda) =< p, \lambda >$, and then use a demand oracle for v_j to choose λ that maximizes the utility $u_j(\lambda) = v_j(\lambda)- < p, \lambda >$. In the second part of the paper we introduce a slight modification to this simple form, where we set the price of the complete bundle, $p_j((\Theta, \Theta, , , , , \Theta))$, separately, still staying within the general characterization.

4 The Generic Algorithm

Both our online and offline algorithms are based on a generic algorithm described here.

4.1 Description of the Algorithm

The logic of the generic algorithm is simple: At any point in time good i has price $P^{(i)}$. The bidders arrive one after the other, and when bidder j is considered he chooses which bundle he prefers according to the current prices. The prices $P^{(i)}$ are initialized to some parameter P_0 and are increased whenever a quantity of that good is allocated. The increase in price is exponential with a rate r per unit allocation.

Generic Algorithm with parameters P_0 and r

for each good i

  $l_1^{(i)} = 0$

for each bidder $j = 1...m$

- for each good i

  $P_j^{(i)} = P_0 \cdot r^{l_j^{(i)}}$
- Query j's demand oracle on the current prices and allocate:

  $D_j\left(P_j^{(1)}...P_j^{(n)}\right) \rightarrow \left(x_j^{(1)}...x_j^{(n)}\right)$
- determine bidder's j payment as $P_j^{tot} = \sum_i x_j^{(i)} P_j^{(i)}$
- update $l_{j+1}^{(i)} = l_j^{(i)} + x_j^{(i)}$

end

4.2 Analysis of Generic Algorithm

The correctness of algorithm involves three elements: incentive compatibility, validity (i.e. that no item is over-allocated), and approximation ratio. We start with incentive compatibility:

Lemma 1 *For any exogenous choice of parameters P_0 and r, the generic algorithm is incentive compatible.*

The proof that this algorithm is incentive compatible derives directly from theorem 1. Note that the values of the parameters can not depend on the valuations since otherwise incentive compatibility may be lost.

Our next step is to prove the validity of the algorithm, i.e. that it never allocates more than the available quantity of each good. This is true as long the values of P_0 and r satisfy a certain condition. Let $l_j^{(i)} = \sum_{t=1}^{j-1} x_t^{(i)}$ denote the total allocation of good i to all players preceding j and let $l_*^{(i)} = l_{m+1}^{(i)}$ denote the total allocation of good i to all players. Let v_max be the highest valuation in the auction, i.e. $v_max = \max_{j,\lambda} v_j(\lambda)$.

Lemma 2 *Let P_0, r be such that the condition $P_0 r^\gamma \geq \frac{v_{max}}{\theta}$ holds, then $l_*^{(i)} \leq \gamma + \Theta$. In particular for $\gamma = 1 - \Theta$ the algorithm is valid.*

Proof: Assume to the contrary that $l_*^{(i)} > \gamma + \Theta$, and let j be the first player that caused this to happen for some good i, i.e., $l_{j+1}^{(i)} > \gamma + \Theta$. Since no player is allocated more than Θ units of each good, we have that $l_j^{(i)} > \gamma$. It follows that $P_j^{(i)} > P_0 r^\gamma \geq \frac{v_{max}}{\theta}$. Since player j is allocated at least θ units of good i, his payment is at least $\theta P_j^{(i)} > v_{max} \geq v_j(x_j)$. Thus player j's payment is more than his valuation for the bundle allocated, in contradiction to the definition of the demand oracle and the possibility of choosing the empty bundle and paying nothing. ∎

Our final step is to prove a bound on the approximation quality. For an allocation algorithm A, $V(A)$ denote the total sum of bidders' valuations for the allocation produced. I.e. $V(A) = \sum_j v_j(x_j)$, where $(x_1, ..., x_m)$ is the allocation produced by A.

Lemma 3 $V(ALG)\left(1 + \frac{r^\Theta - 1}{\Theta}\right) \geq V(OPT) - nP_0$.

The proof of this lemma is given in appendix C.

Combining these three lemmas we immediately obtain a incentive compatible, valid, approximation algorithm as long as the following two conditions on the parameters P_0 and r hold:

1. $nP_0 \leq \frac{V(OPT)}{2}$.
2. $r^{1-\Theta} \geq \frac{v_max}{\theta P_0}$.

Under these conditions no item is over allocated and the approximation ratio achieved is $C = 2\left(1 + \frac{r^\Theta - 1}{\Theta}\right)$.

4.3 An Online Algorithm

Notice that the generic algorithm is online in the sense of [13]: players arrive one at a time, and the allocation and the payment of each player is determined as he arrives. The only thing missing in order to obtain a complete online algorithm is the choice of parameters. In our algorithm, this choice needs to be made before any players arrive. This can only be done if we have some a priori bounds on the possible valuations of players. The following assumption suffices:

A priori known bound on maximal valuation: There exists a priori known bounds v_{min} and v_{max} such that: $v_{min} \leq \max_j v_j((\Theta, ..., \Theta)) \leq v_{max}$.

Theorem 2 *There exists an* online *incentive compatible mechanism for multi-unit combinatorial auctions with (θ, Θ)-bounded demand that is computationally efficient, incentive compatible, and achieves an $O\left(\frac{1}{\Theta}\left(\frac{\rho n}{\theta}\right)^{\frac{\Theta}{1-\Theta}}\right)$ approximation to the optimal allocation, where $\rho = v_{max}/v_{min}$.*

Proof: Use the generic algorithm with parameters $P_0 = \frac{v_{min}}{2n}$ and $r = \left(\frac{v_max}{\theta P_0}\right)^{\frac{1}{1-\Theta}}$. ∎

5 The Final Algorithm

In this section we give an incentive compatible approximation algorithm that does not require a priori known bounds on the valuations of players. It is easy to see that such an algorithm can not be online, and indeed we will choose the parameters P_0 and r as a function of the valuations. If we do not need to worry about incentive compatibility then a choice of $P_0 = v_{max}/(2n)$ and $r = (2n/\theta)^{1/(1-\Theta)}$ will satisfy the two conditions of section 4.2. Moreover, this is "almost" incentive compatible: since these parameters only depend on v_{max}, incentive compatibility is maintained for all players *except for the one with the maximum valuation* v_{max}. (In the case where there is more than one bidder with the maximum valuation v_{max} the incentive compatibility is maintained.)

Our challenge is to design an algorithm that is also incentive compatible for the player with highest valuation. The basic idea is to use a "2nd" price type mechanism for this player. Specifically, when calculating the allocation and payments for this player use a different set of parameters, those calculated using the second highest price v_{sec} rather than the first highest price. In order to make this idea work we need to make a few delicate changes in the algorithm so as to fix the problems caused by treating the highest price player differently than the others.

Final Algorithm

for each bidder $j = 1...m$

- Compute maximum valuation of others: $v_{max}^{-j} = \max_{k \neq j, \lambda} v_k(\lambda)$, and let b be the first player that achieves this maximum.
- Compute parameters: $r = \left(\frac{6n}{\theta}\right)^{\frac{1}{1-2\Theta}}$ and $P_0 = \frac{v_{max}^{-j}}{6n}$.
- Run the Generic algorithm on all players excluding b, and obtain j's allocation and payment: $x_j = \left(x_j^{(1)} ... x_j^{(n)}\right)$, $P_j^{tot} = \sum_i P_j^{(i)} x_j^{(i)}$, (and ignore all allocations or payments of other players.)
- if j prefers instead the whole bundle at price v_{max}^{-j}, i.e.
  If $v_j((\Theta, ..., \Theta)) - v_{max}^{-j} > v_j(x_j) - P_j^{tot}$ Then
  change j's allocation: $x_j = (\Theta, ..., \Theta)$ and $P_j^{tot} = v_{max}^{-j}$

While it may seem from this description of the algorithm that we are running m different copies of the generic algorithm, in reality we are only running two copies. Notice that for all players except b, the value v_{max}^{-j} is the same and equals to v_{max}. Note also that none of these players will change their allocation by choosing the whole bundle at price v_{max} in the last line of the algorithm, since their valuation is no more than v_{max}. Thus the same generic algorithm is run on all of them. For player b, we run a different algorithm based on $v_{max}^{-b} = v_{sec}$, where v_{sec} is the second highest valuation, and excluding the player with the second highest valuation.

The original description of the algorithm makes incentive compatibility very clear:

Lemma 4 *The algorithm is incentive compatible.*

Proof: Notice that each player is simply offered a price for each possible allocation, a price that is independent of its own bid. Specifically the price for every $x_j \neq (\Theta, ..., \Theta)$ is $P_j(x_j) = \sum_i P_j^{(i)} x_j^{(i)}$, and the price offered for $x_j = (\Theta, ..., \Theta)$ is $P_j((\Theta, ..., \Theta)) = \min(v_{max}^{-j}, \sum_i P_j^{(i)} x_j^{(i)})$. The mechanism allocates x_j that maximizes j's utility, and thus Theorem 1 applies. ∎

The validity of the algorithm follows from the fact that there are really only two copies of the generic algorithm.

Lemma 5 *The algorithm does not allocate more than 1 unit of each good.*

Proof: First note that the allocation to b is at most Θ units from each good. We can bound the allocation to all other players by noting that we are running a single generic algorithm for all of them with parameters $r = \left(\frac{6n}{\theta}\right)^{\frac{1}{1-2\Theta}}$ and $P_0 = \frac{v_{max}}{6n}$. Thus the conditions of lemma 2 apply with $\gamma = 1 - 2\Theta$ and thus the total allocation to these players is bounded by $1 - \Theta$. ∎

Finally we can prove an approximation ratio.

Lemma 6 *The approximation ratio of this algorithm is* $3\left(1+\frac{r^{\Theta}-1}{\Theta}\right)$.

Proof: Denote ALG^{-b} to be the algorithm that runs on all the players except b. We have that $V(ALG) \geq V(ALG^{-b}) + v_b(x_b)$. Notice that ALG^{-b} is exactly the generic algorithm running on all players except for b. Let $C = \left(1+\frac{r^{\Theta}-1}{\Theta}\right)$, then using Lemma 3 we get $C \cdot V(ALG^{-b}) \geq V(OPT^{-b}) - nP_0 = V(OPT^{-b}) - v_{max}/6$.

The last line of the algorithm implies that player b's benefit, $v_b(x_b) - P_b^{tot}$, is bounded below by $v_{max} - v_{sec}$. Therefore,

$$V(ALG) \geq V(ALG^{-b}) + v_{max} - v_{sec} \geq \frac{1}{C}V(OPT^{-b}) + (1-\frac{1}{6C})v_{max} - v_{sec}.$$

On the other hand,

$$V(OPT) \leq V(OPT^{-b}) + v_{max}.$$

So if $v_{sec} < \frac{3}{4}v_{max} \leq (1-\frac{1}{6C}-\frac{1}{3C})v_{max}$ we are done. Otherwise,

$$\begin{aligned} V(ALG) &\geq V(ALG^{-b}) \geq \frac{1}{C}\left(V(OPT^{-b}) - \frac{v_{max}}{6}\right) \\ &\geq \frac{1}{3C}\left(V(OPT^{-b}) + 2v_{sec} - \frac{v_{max}}{2}\right) \geq \frac{1}{3C}\left(V(OPT^{-b}) + v_{max}\right), \end{aligned}$$

which completes the proof. ∎

We thus get:

Theorem 3 *There exists an incentive compatible mechanism for multi-unit combinatorial auctions with (θ,Θ)-bounded demand that is computationally efficient and achieves an* $O\left(\frac{1}{\Theta}\left(\frac{n}{\theta}\right)^{\frac{\Theta}{1-2\Theta}}\right)$ *approximation to the optimal allocation.*

6 On the Hardness of Approximation of the k-Duplicates Problem

Theorem 4 *For every fixed $K \geq 1$, approximating the Multi Unit Combinatorial auctions with K-duplicates problem to within a factor of* $O\left(n^{\frac{1-\epsilon}{K+1}}\right)$ *is NP-hard unless NP=ZPP, for any fixed $\epsilon > 0$.*

The proof is given in Appendix D.

Acknowledgements

The authors would like to thank Irit Dinur. We are in particular thankful to Mira Gonen for her help in studying the inapproximability questions in this paper.

References

[1] Aaron Archer, Christos Papadimitriou, Kunal Talwar and Eva Tardos. An approximate truthful mechanism for combinatorial auctions with single parameter agents, *to appear in Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms (2003).*

[2] B. Awerbuch, Y. Azar, and A. Meyerson. Routing Truth-telling Online Mechanisms to Online Optimization. Unpublished draft 2003.

[3] B. Awerbuch, Y. Azar, and S. Plotkin. Throughput-Competitive On-Line Routing. *Proceeding of 34th FOCS 1993*, pp. 32-40.

[4] Y. Azar, and O. Regev. Strongly Polynomial Algorithm for the Unsplittable Flow Problem. *Proceeding of 8th IPCO 2001*, pp. 15-29.

[5] E. H. Clarke Multipart Pricing of Public Goods. *In journal Public Choice 1971*, volume 2, pages 17-33.

[6] R. Gonen, and D. Lehmann Optimal Solutions for Multi-Unit Combinatorial Auctions: Branch and Bound Heuristics. *Proceeding of ACM Conference on Electronic Commerce EC'00*, pp. 13-20, Minneapolis, Minnesota, October 2000.

[7] R. Gonen, and D. Lehmann Linear Programming helps solving Large Multi-unit Combinatorial Auctions. *In Proceeding of INFORMS 2001*, November 4-7, 2001, Miami Beach, Florida.

[8] Green, J. R., and J. J. Laffont Characterization of satisfactory mechanisms for the reveraltion of preferences in public goods. Econometrica 45, 1977.

[9] T. Groves Incentives in teams. *In journal Econometrica 1973*, volume 41, pages 617-631.

[10] J. Hastad Clique is Hard to Approximate to within $n^{1-\epsilon}$ *Journal Acta Mathematica 1999*, volume 182.

[11] Hofmeister, and Lefmann. Approximating Maximum Independent Sets in Uniform Hypergraphs Erscheinungsjahr. 1998

[12] Holzman, R., N. Kfir-Dahav, D. Monderer, and M. Tennenholtz On Bundling Equilibrium in Combinatorial Auctions mimeo, Technical report, Technion, http://iew3.technion.ac.il/˜moshet/rndm11.ps

[13] R. Lavi and N. Nisan. Competitive Analysis of Online Auctions. in *Proceeding of ACM Conference on Electronic Commerce, 2000.*

[14] D. Lehmann, L. I. O'Callaghan, and Y. Shoham. Truth revelation in rapid, approximately efficient combinatorial auctions. In *Proceedings of the First ACM Conference on Electronic Commerce. EC'99*, pages 96–102, Denver, Colorado, November 1999. SIGecom, ACM Press.

[15] B. Lehmann, and D. Lehmann, and N. Nisan. Combinatorial Auctions With Decreasing Marginal Utilities. *In Proceeding of ACM conference on electronic commerce, 2001*

[16] A. Mu'alem, and N. Nisan. Truthful Approximation Mechanisms for Restricted Combinatorial Auctions. *AAAI*, 2002.

[17] N. Nisan Algorithms for Selfish Agents. In *Proceedings of STACS, 1999*

[18] N. Nisan Bidding and Allocation in Combinatorial Auctions. *Proceeding of ACM Conference on Electronic Commerce, 2000*

[19] N. Nisan. The Communication Complexity of Approximate Set Packing and Covering. in *ICALP*, 2002.

[20] N. Nisan and A. Ronen. Algorithmic Mechanism Design. In *Proceedings of STOC. 1999*

[21] N. Nisan, and A. Ronen Computationally Feasible VCG-based Mechanisms. *In Proceeding ACM Conference on Electronic Commerce 2000*

[22] N. Nisan, and I. Segal. The Communication Complexity of Efficient Allocation Problems. Working paper.

[23] C. Papadimitriou Algorithm, games, and the Internet. *In STOC 2001.*

[24] Roberts, K. The characterization of implementable choice rules. in aggregation and revelation of preferences, editor, J.J. Laffont, 1979.

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

[26] T. Sandholm Limitations of the Vickrey Auction in Computational Multiagent Systems. *Proceedings of the Second International Conference on Multiagent Systems (ICMAS-96)*, pages 299-306, Kyoto, Japan, December 1996.

[27] T. Sandholm An Algorithm for Optimal Winner Determination in Combinatorial Auctions. *Proceeding of IJCAI-99*

[28] W. Vickrey Counterspeculation, Auctions and Competitive Sealed Tenders. *In Journal of Finance 1961*, volume 16, pages 8-37.

[29] Vohra, R., and S. de Vries Combinatorial auctions: A survey. Available from http://www.kellogg.nwu.edu/faculty/vohra/htm/res.htm.

[30] E. Zurel and N. Nisan An Efficient Approximate Allocation Algorithm for Combinatorial Auctions. *Proceeding of ACM conference on electronic commerce EC'01*

A Languages allowing Demand Oracle Queries

In this appendix we shortly discuss concrete "bidding languages" where demand oracle queries can be easily answered.

Our first set of such languages comes from combining the following basic representational operations all allow efficient answering of such oracle queries:

- **Atomic Bids:** Offering a price p for the combination $(\lambda^{(1)}...\lambda^{(n)})$. I.e. $v(x^{(1)}...x^{(n)}) = p$ if for all i, $x^{(i)} \geq \lambda^{(i)}$, and $v(x^{(1)}...x^{(n)}) = 0$ otherwise. In this case the demand for query $p^{(1)}...p^{(n)}$ is $(\lambda^{(1)}...\lambda^{(n)})$ if $p > \sum_i \lambda^{(i)} p^{(i)}$ and (0...0) otherwise.

- **XOR of bids** The XOR of two bids is to take the best one of them, but not both. I.e. if $v = v_1 XOR v_2$ then $v(x^{(1)}...x^{(n)}) = \max(v_1(x^{(1)}...x^{(n)}), v_2(x^{(1)}...x^{(n)}))$. In this case the demand for query $p^{(1)}...p^{(n)}$ would be obtained by getting the demand λ_1 for v_1, the demand λ_2 for v_2, comparing the surpluses $v_1(\lambda_1) - < \lambda_1, p >$ vs. $v_2(\lambda_2) - < \lambda_2, p >$, and choosing the larger one.

- **Non-conflicting-OR of bids** The OR of two non-bids is to treat them independently. The bids are non-conflicting if they can always be satisfied concurrently. I.e. v_1 and v_2 are non-conflicting if for all $p^{(1)}...p^{(n)}$, we have that $\lambda_1^{(i)} + \lambda_2^{(i)} \leq 1$, where λ_1 and λ_2 are the respective demands of v_1 and v_2. In this case the demand for $v = v_1 OR v_2$ is simply $\lambda_1 + \lambda_2$.

Our second class of languages concerns auctions for network links: Fix an arbitrary (multi-)graph, and consider a combinatorial auction for the links on this graph. The following natural classes of bids can all efficiently support demand oracle queries, using the standard algorithms on graphs.

- **Path Bids:** Offering a price p for any path in the graph that connects a given source and destination. The demand oracle query is answered using a shortest path algorithm.

- **Matching Bids:** In a bipartite graph, offering a price p for a perfect matching between given sets of left vertices (consumers of a virtual good) and right vertices (producers of the virtual good). The demand oracle query is answered using a min-weight matching algorithm.

B Proof of Theorem 1

Proof: We first show that these two conditions are sufficient. Fix v_{-j} and v_j. Now consider an alternative "lie" $v`_j$ for bidder $j$. Let $\lambda$ and $p$ be the mechanism's output for $j$ with input $(v_j, v_{-j})$ and $\lambda`$ and $p`$ be the mechanism's output for $j$ with input $(v`_j, v_{-j})$. If $\lambda = \lambda'$ then the first condition ensures that $p = p' = p_j(\lambda)$, and thus both the allocation and the payments with declaration $v`_j$ are equivalent to those obtained with the truth. If $\lambda' \neq \lambda$, then $p = p_j(\lambda)$, $p` = p_j(\lambda')$, and the second condition ensures that $v_j(\lambda) - p_j(\lambda) \geq v_j(\lambda`) - p_j(\lambda`)$, and thus the utility with declaration $v`_j$ is less than that obtained with the truth.

We now show that they are necessary. Assume to the contrary that the first condition does not hold, i.e. that for some v_{-j}, and two valuations v_j and v'_j, the mechanism yields the same allocation λ to player j, but charges different payments $p > p'$, respectively, from him. Now it is clear that for the case where bidders' valuations are v_{-j} and v_j, for bidder j to declare v'_j instead of v_j will improve his utility (since the allocation remains the same, while the payment decreases), contrary to the definition of incentive compatibility.

Now assume that the first condition holds, but assume to the contrary that the second condition does not hold, i.e. that for some v_{-j}, and valuation v_j, the mechanism allocates λ to j with the property that $v_j(\lambda) - p_j(\lambda) < v_j(\lambda`) - p_j(\lambda`)$, for some λ' that can be allocated to j, e.g. if he bids v'_j. But this exactly says that for the case where bidders' valuations are v_{-j} and v_j, then for bidder j to declare v'_j instead of v_j will improve his utility (since he is now allocated λ' and charged $p_j(\lambda')$), contrary to the definition of incentive compatibility. ▮

C Proof of lemma 3

Let $V(A)$ denote the total sum of valuations for the allocation of algorithm A to the bidders. Our goal is to prove that $V(OPT) \leq C \cdot V(ALG)$ where C is the approximation ratio.

Lemma 7 *$\forall j$ and $\forall \overrightarrow{\lambda}_j$*

$$v_j\left(\overrightarrow{x}_j\right) \geq v_j\left(\overrightarrow{\lambda}_j\right) - \left\langle \overrightarrow{\lambda}_j, \overrightarrow{P}_* \right\rangle,$$

where P_*, is the vector of the goods prices at the end of allocation, $P_* = P_*^{(1)} ... P_*^{(n)}$, and where $P_*^{(i)} = P^{(0)} \cdot r^{l_*^{(i)}}$.

Proof:
$$v_j\left(\overrightarrow{x}_j\right)-\left\langle \overrightarrow{x}_j,\overrightarrow{P}_j\right\rangle \geq v_j\left(\overrightarrow{\lambda}_j\right)-\left\langle \overrightarrow{\lambda}_j,\overrightarrow{P}_j\right\rangle .$$
This inequality derives from the demand oracle definition.
$$v_j\left(\overrightarrow{x}_j\right)-\left\langle \overrightarrow{x}_j,\overrightarrow{P}_j\right\rangle \geq v_j\left(\overrightarrow{\lambda}_j\right)-\left\langle \overrightarrow{\lambda}_j,\overrightarrow{P}_*\right\rangle .$$
The last inequality holds true since $\overrightarrow{P}_* \geq \overrightarrow{P}_j$ for any j. Since $\left\langle \overrightarrow{x}_j,\overrightarrow{P}_j\right\rangle \geq 0$ we get the lemma. ▮

Summing all bidders will give us the next Corollary.

Corollary 1 $V\left(ALG\right) \geq V\left(OPT\right) - \sum_i P_*^{(i)}$.

Since the algorithm is individually rational (i.e each bidder pays no more than the value of the bundle he gets) the total revenue is a lower bound for the total valuation. When bidder j is allocated we have
$$v_j\left(\overrightarrow{x}_j\right) \geq \left\langle \overrightarrow{x}_j,\overrightarrow{P}_j\right\rangle = \sum_i x_j^{(i)} P_0 r^{l_j^{(i)}}.$$
Summing for all bidders we have
$$V\left(ALG\right)=\sum_j v_j\left(\overrightarrow{x}_j\right) \geq \sum_j\sum_i x_j^{(i)} P_0 r^{l_j^{(i)}} = \sum_i\sum_j x_j^{(i)} P_0 r^{l_j^{(i)}}.$$
Thus $V\left(ALG\right) \geq \sum_i R^{(i)}$ where $R^{(i)} = \sum_j x_j^{(i)} P_0 r^{l_j^{(i)}}$ is the total revenue obtained for good i.
Let $\Delta_j R^{(i)} = x_j^{(i)} P_0 r^{l_j^{(i)}}$, then $R^{(i)} = \sum_j \Delta_j R^{(i)}$.
For convenience we denote $h = x_j^{(i)}$, $t = l_j^{(i)}$ and $\Delta R^{(i)} = \Delta_j R^{(i)} = hP_0 r^t$.

To bound the total price change we compare this quantity to the change when the price grows continuously denoted $\Delta\overline{R^{(i)}}$.
$$\frac{\Delta\overline{R^{(i)}}}{P_0} = \int_t^{t+h} r^x dx = \frac{r^x}{\ln r}\Big|_t^{t+h} = \frac{1}{\ln r}\left(r^{t+h}-r^t\right) = \frac{r^t}{\ln r}\left(r^h - 1\right).$$
Since the demand of any good is bounded by Θ we can bound the ratio between $\Delta\overline{R^{(i)}}$ and $\Delta R^{(i)}$:
$$\max_{h\leq\Theta}\left\{\frac{\Delta\overline{R^{(i)}}}{\Delta R^{(i)}}\right\} = \max_{h\leq\Theta}\left\{\frac{\frac{r^t}{\ln r}\left(r^h-1\right)}{hr^t}\right\} = \max_{h\leq\Theta}\left\{\frac{1}{\ln r}\cdot\frac{r^h-1}{h}\right\} = \frac{1}{\ln r}\cdot\frac{r^\Theta - 1}{\Theta}.$$

And so

$$R^{(i)} = \sum_j \Delta_j R^{(i)} \geq \frac{\Theta \ln r}{r^\Theta - 1} \sum_j \Delta_j \overline{R^{(i)}} = \frac{\Theta \ln r}{r^\Theta - 1} \overline{R^{(i)}} = \frac{\Theta \ln r}{r^\Theta - 1} \int_0^{l_*^{(i)}} P_0 r^t dt.$$

$$R^{(i)} \geq \frac{\Theta \ln r}{r^\Theta - 1} \cdot \frac{P_0 r^t}{\ln r} \Big|_0^{l_*^{(i)}} = \frac{\Theta \ln r}{(r^\Theta - 1) \ln r} \left[P_0 r^{l_*^{(i)}} - P_0 \right] = \frac{\left(P_*^{(i)} - P_0 \right) \Theta}{r^\Theta - 1}.$$

Summing this result over all n goods, the following bound is achieved:

Lemma 8 $V(ALG) \geq \sum_i R^{(i)} \geq \frac{\left(\sum_i P_*^{(i)} - nP_0 \right) \Theta}{r^\Theta - 1}$

Lemma 8 gives the inequality

$$V(ALG) \left(\frac{r^\Theta - 1}{\Theta} \right) + nP_0 \geq \sum_i P_*^{(i)}.$$

Together with corollary 1 we obtain Lemma 3.

D Proof of Theorem 4

In order to show the proof of Theorem 4 we will show a reduction from Maximum independent Set (MIS) problem in Hypergarphs.

Definition 8 *A K-Uniform Hypergarph $H_K = (V, E)$ is a hypergarph with all edges of size K.*

Definition 9 *Maximum Independent Set problem for Hypergraphs: Given a hypergarph H_K, find a maximum independent set, i.e. find a set of vertices with maximum size such that no subset of these vertices form an edge in H_K.*

The result is based on the following theorem.

Theorem 5 *Let $K \geq 2$ be a fixed integer and let $\varepsilon > 0$ be fixed. One cannot approximate in polynomial time the size of a MIS in K-uniform hypergraphs on $|V|$ vertices within factor of $O\left(|V|^{1-\varepsilon}\right)$, unless NP=ZPP.*

Theorem 5 was also stated at [11].

Proof: (of theorem 4): Given an instance of an $(K+1)$-uniform hypergraph $H_K = (V, E)$, we define an instance of the K-duplicate multi unit combinatorial auction as follows: For every vertex we have a bidder, and for every edge we have a good which is requested by each of the bidders corresponding to the vertices of that edge. The number of goods is $n = |E| \leq \left(\begin{array}{c} |V| \\ K+1 \end{array} \right) \leq |V|^{K+1}$.

In a solution for the multi unit combinatorial auction any good is allocated to at most K different bidders and therefore the set of vertices corresponding to the accepted bidders is an independent set in the hypergraph H_K and vice versa. This implies it is hard to approximate the multi unit combinatorial auction within $O(n^{\frac{1-\varepsilon}{K+1}})$ unless $NP = ZPP$. ∎

For completeness we present here a simple proof of Theorem 5.

Proof: We reduce the MIS problem for graphs to the MIS for K-uniform hypergraphs. Given a graph $G = (V, E)$ we define the following hypergarph H_K: H_K has the same set of vertices as G, and every subset of vertices with size K that contain an edge in G form an edge in H_K.

Claim 1 *I is a MIS in G with size $\geq K$ if and only if I is a MIS in H_K with size $\geq K$.*

Proof: Let I be a MIS in G. Then I is an IS in H_K because no edge in G is contained in I, so no subset of I form an edge in H_K. I is also MIS: assume to the contrary that there is a MIS in H_K I' such that $\left|I'\right| > |I|$. Since no subset of I' form an edge in H_K, by the definition of H_K no two vertices in I' form an edge in G. Therefore I' is a MIS in G - contradiction the fact that I is a MIS in G and $\left|I'\right| > |I|$. For the same arguments if I is a MIS in H_K, then I is also a MIS in G. ∎

Since MIS problem for graphs is hard to approximate within $O\left(|v|^{1-\varepsilon}\right)$ unless NP=ZPP, MIS problem for hypergraphs is also hard to approximate within $O\left(|v|^{1-\varepsilon}\right)$ unless NP=ZPP. ∎