Algorithmic Game Theory December 23, 2013

Lecture 10

Lecturer: Michal Feldman *Scribe: Baruch Weizman, Barak Arkis, Lior Shultz*

1 Multi-unit auction polynomial time approximation algorithm

Reminder

In the multi-unit auction problem the valuation of an allocation is defined as:

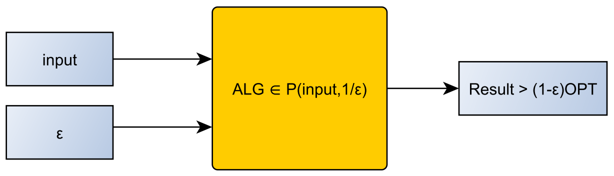
$$\sum_i v_i(k_i)$$

Where k_i is the number of units allocated to bidder i and v_i is the bidder i's value function.

Reminder

We saw an algorithm for finding OPT in the multi-unit auction problem. The algorithm runs in $O(nm^2)$ using dynamic programming. However, the size of the input is described as $n, \log m$. This means that we have an exponential time algorithm.

We now want to find a polynomial algorithm that will give us an approximation of OPT.



Theorem 1 *The algorithm presented here is a FPTAS (Fully Polynomial Time Approximation Scheme) based on dynamic programming.*

Lets us define $V = \max_i v_i([m])$ *and* $\delta = \epsilon \frac{V}{n}$

We round down every value of $v_i(k)$ to an integer multiple of δ and run the algorithm on the modified v_i evaluation functions.

Proof: This means that every value is of the form $w\delta$, with w being an integer. We also know that:

$$w\delta \leq V$$
$$w\epsilon\frac{V}{n} \leq V$$
$$w \leq \frac{n}{\epsilon}$$

Since the largest possible evaluation for n players is the sum of n such values we get that that OPT is bounded by:

$$\frac{n \cdot n}{\epsilon} = \frac{n^2}{\epsilon}$$

On the other hand the maximal error for a single player is smaller than δ. Since the error is additive we conclude that the maximal error is $n\delta$ and we get:

$$n\delta = \frac{n\epsilon V}{n} = \epsilon V$$

This means that our maximal deviation from OPT is:

$$\frac{\epsilon V}{OPT} \leq \epsilon$$

We use the table $K(i,w)$ in our dynamic algorithm. The cell in (i,w) holds the minimal number of distributed products needed to achieve a value of at least $w\delta$ when distributed between the first i players. We initialize the table with $K(i,0) = 0$ for every i. We define $val(j)$ as the minimal number distributed products needed in order to achieve a value of at least $w\delta$ with player i receiving a value of at least $j\delta$. This means that the rest of table can be filled using the following method:

$$K(i,w) = \min_{0 \leq i \leq j} val(j),$$

$$val(j) = k + K(i-1, w-j),$$

k is the minimal number of products that need to be allocated to player i to achieve a value of $j\delta$. This way by adding $K(i-1,w-j)$ we ensure we receive a total value of at least $w\delta$, as needed. The output of the algorithm is $\max_{w} K(n,w) \leq m$. This is the largest value we can achieve while distributing no more than m items.

The table has $n \cdot \frac{n^2}{\epsilon}$ cells. To fill each cell we need to do a binary search over m for each j we try out. There are $\frac{n}{\epsilon}$ options for j. This gives us a total running time of $O(\frac{n^4 \log m}{\epsilon^2})$ which is polynomial as required. ■

1.1 Truthful approximation algorithms

We now ask an interesting question: Let us assume that we have an $NP - hard$ problem and a polynomial algorithm that gives us an α approximation for the problem. Can we get the same approximation using a truthful algorithm? (Can we receive the truthful property for "free"?)

We try to do this with VCG. We will now try to run the approximated algorithm for VCG with the valuation functions rounded down to integers and the payments calculated using the original valuation function. We see that in this case we lose VCG's truthfulness.

Example: We have two players and two products. The valuation function for both of them is:

$$v(1) = 1.9, v(2) = 3.1$$

In the optimal solution each player receives one product and pays $3.1 - 1.2 = 1.9$.

Now we look at the approximation when we cut the values down to integers:

$$v^*(1) = 1, v^*(2) = 3$$

Using this approximation one of the players will receive both products and pay 3.1 giving him a utility of 0. However, if a player lies and says that his value function really is:

$$v(1) = 3, v(2) = 3$$

Each player will receive one product and the lying player will pay 1.2 giving him a utility of $1.9 - 1.2 = 0.7 > 0$. Meaning that this approximation is *not* truthful!

2 Maximum In Range

In this family of approximation algorithms, we find the optimal solution amongst a limited subset of allocations. On the one hand, this subset should be rather small so we can find an optimal solution efficiently. On the other hand, it should be large enough so that the approximation factor will be satisfying.

We will present a Maximum-In-Range algorithm which yields 2-approximation.

2-approximan truthful MIR algorithm: for the sake of simplicity, we will assume that n^2 divides m. The algorithm works in the general case as well.

We will divide the m products to n^2 packages, of size m/n^2 each.Now, we will apply VCG on these packages. In this case, we can find the allocation which maximizes the social welfare efficiently, by using the dynamic-programming algorithm we saw earlier, as its running time is polynomial in the number of players and the number of products. By dividing the products to n^2 packages, the number of products is also polynomial in the number of players, and therefore the total running time of the algorithm will be polynomial in the number of players,n.

Claim 2 *This algorithm yields a truthful 2-approximation to the multi-unit auction problem.*

Proof: Let the optimal allocation be $(m_1^*, m_2^*, ..., m_n^*)$ where m_j^* is the number of products allocated to the $j-$th player. Let i be the player who got the largest amount products. Obviously, $m_i^* \geq m/n$. We will look in two complementary cases.
Case 1:
$v_i(m_i^*) \geq \sum_{j \neq i} v_j(m_j^*)$. That is, at least 1/2 of the total SW comes from i. The allocation which gives the $i$the player all of the products is naturally in the range. Hence, we will get at least 2-approximation in this case.
Case 2:
$v_i(m_i^*) \leq \sum_{j \neq i} v_j(m_j^*)$. Namely, most of the SW comes from the other $n-1$ players. In this case, we can take all of the products given to the i-th player, and divide them between the other $n-1$ players, so that each of them will have at least $\lceil m_j^*/(m/n^2) \rceil * (m/n^2)$. We need at most $n * m/n^2 = m/n$ products for this.v_i has ,indeed, (at least), this amount of products. This allocation is in the range and hence a 2-approximation. ■

Theorem 3 *There is no MIR algorithm with approximation ratio less than 2 that runs in time complexity which is polynomial in $logm, n$.*

Proof:

Let ALG be a MIR polynomial-time algorithm which has approximation ratio < 2. If the range is full that is, it contains all pairs $(m_, m - m_1)$ we will get OPT in polynomial time. This, however, contradicts that fact we have proven in class that there is no polynomial time algorithm which finds the allocation which maximizes the social welfare, even for two players. If the range is not full, there is a pair $(m_1, m - m_1)$ that is not consisted in the range. We will look at the following input:

$$v_1(k) = \begin{cases} 1 & \text{if } k \geq m_1 \\ 0 & \text{else} \end{cases}$$

and let $v_2(k)$ be defined as follows:

$$v_2(k) = \begin{cases} 1 & \text{if } k \geq m - m_1 \\ 0 & \text{else} \end{cases}$$

The optimal allocation is $(m_1, m - m_1)$, yielding a SW of 2. This allocation is not, however, in the range.Therefore, this allocation is not a possible outcome of ALG. It means that the maximal SW induced by a possible outcome of ALG is 1, in contradiction to the fact that ALG has approximation ratio smaller than 2. ■

Now let's have a look again at single-minded bidders. In this case, every bidder i has a valuation function of the following form:

$$v_i(k) = \begin{cases} v_i & \text{if } k \geq k_i \\ 0 & \text{else} \end{cases}$$

We showed the following characterization of truthful mechanisms in single-parameter environments, regarding the allocation and the payments:

1. Monotonic allocation: If (k_i, v_i) wins, then (k_i', v_i') also wins for $(k_i' \leq k_i)$ and $(v_i' \geq v_i)$
2. Critical value: a player who won with value (k, v) pays the lowest v' such that (k, v') also wins

Recall that there exists a dynamic-programming algorithm which gives a $1 + \epsilon$ approximation, for every ϵ. We should ask ourselves whether the allocation induced by this algorithm is monotonic. It seems that the answer is yes, and it is indeed monotonic for fixed values of δ and ϵ. The problem is, however, that δ was defined by $\delta = \epsilon V/n$, where $V = max_i v_i(m)$. δ depends on V, and by increasing her bid, player i can increase δ and by that, get less products. It turns that there is an algorithm that promises truthfulness, but this is beyond the scope of this lecture.

3 Non quasi-linear valuations

So far, we have assumed that the utility function of each player is: $u_i = v_i - p_i$ we will look at a few settings where this assumption does not take place.

1. Players with budgetconstraints
2. Mechanism design without money

In this lecture, we will focus on players with budget constraints. In this setting, the utility of each player is

$$u_i = \begin{cases} v_i - p_i & \text{if } p_i \leq B_i \\ -\infty & \text{else} \end{cases}$$

SW cannot be maximized in this setting; in second-price auction, for instance, the payments imposed on a player might exceed his budget. We will look in multi-unit auctions. In this setting, each player has a private value v_i where $v_i(k) = v_i k$ and a non-private budget, B_i. We will define the demand of a player i given a price p:

$$D_i(p) = \begin{cases} min(\lfloor B_i/p_i \rfloor, m) & \text{if } p < v_i \\ 0 & \text{if } p > v_i \\ \{0, ..., \lfloor B_i/p \rfloor\} & \text{else} \end{cases}$$

As the price increases, the demand of every player decreases. Let p^* be the price which cleans the market. p^* suffices :

$$\lim_{p \downarrow p^*} \sum D_i(p) \leq m \leq \lim_{p \uparrow p^*} \sum D_i(p)$$

(In other words, p^* is the price below it the demand will be greater than m, and above it the demand will be less than m). An idea for an allocation and payments rule could be the following: Every player i gets $D_i(p^*)$ products at the price of p^* each. It turns out , however, that this mechanism is not truthful.

Example 1 *We will look at the following setup:*

$$\begin{gathered} m = n = 2 \\ B_1 = \infty \ v_1 = 6 \\ B_2 = 5 \ v_2 = 5 \end{gathered}$$

When the first player bids his real value, he will get two products at the price of 5 each. His utility will be $12 - 10 = 2$. On the other hand, if player 1 bids the value of 3, he will get one product and player 2 will get one product. The utility of player 1 will be $6 - 3 > 2$. We will present a mechanism which solves this issue:

3.1 Clinching Auction

Starting at the price $p = 0$ we will keep $S = m$ (S stands for supply) , $\overline{B_i} = B_i$ ($\overline{B_i}$ will keep the current budget of player i)
As long as $s > 0$:

1. Raise p until there exists a player i so that $k = S - \sum_(j \neq i) D_j(p) > 0$
2. Allocate k units to player i at the price of p.
3. Update $S \leftarrow S - k$, $\overline{B_i} \leftarrow \overline{B_i} - pk$

Theorem 4 *The Clinching Auction is a truthful mechanism*

Proof: This theorem could have been proved using Myerson's lemma, but we will present an easier, direct proof. Let v_i be the real valuation of player i, and b_i be the actual bid given by the player. What could player i gain from underbidding? The outcome of the auction will be the same until the price reaches b_i. However, in the prices interval $[b_i, v_i]$ every additional item the player gets, the utility of the player is increased by a certain positive value. Therefore, underbidding can just reduce the player's utility. Similarly, overbidding reduces the player's utility. The outcome of the auction when bidding $b_i > v_i$ is identical until p reaches v_i. Above v_i, every additional item given to the player, the utility of the player is reduced since $v_i - p < 0$ for $p \in [v_i, b_i]$. As overbidding as well as underbidding does not pay, Clinching Auction is indeed a truthful mechanism. ■

Definition 5 *An outcome A is* **Pareto-better** *than outcome B if the utility of at least on player is increased, and the utility of the rest does not decrease.*

Definition 6 *An outcome A is Pareto-optimal if there is no other possible outcome which is Pareto-better than A.*

Claim 7 *(without proof) Clinching Auction is Pareto-optimal*