1 Maximising Seller’s Profits

Last week we discussed the revenue equivalence theorem, that shows that in every two games, in which the choice function is the same, the the expected outcome is the same, up to normalisation.

Supposedly from the point of view of a seller who want to make profit, their is not much to do - no matter what he will try the outcome will be the same. But, that is incorrect. what we actually showed is that if we want to maximise social welfare then we cannot achieve any better profit by changing the mechanism. But, if we want to maximise seller’s profit - it can be done.

1.1 Second Price Auction With Threshold

A second price auction with threshold is an auction, in which the seller will sell the items only if he gets a proposal greater than some threshold, and the buyer pays the greater between the threshold and the second highest bid.

Let’s examine in the following case: 1 seller, 1 item and two buyers. The $V_i$ of the buyers distribute uniformly in $[0, 1]$, and the threshold is $\frac{1}{2}$, meaning that only if at least one of the offers is greater than $\frac{1}{2}$ the item will be sold. If no offer is grater than $\frac{1}{2}$ the item will be destroyed.

First note that this action is incentive compatible - that is since the price the winner pays does not depend on his offer - $P(\text{winner}) = \max\{\frac{1}{2}, \text{loser's bid}\}$. What will be the seller’s profit in this case? Since the sellers bid is uniformly distributed, with probability $\frac{1}{4}$ both players will bid less than $\frac{1}{2}$, and the item will be destroyed, leading to no profit. With probability $\frac{1}{4}$ one player will bid more than $\frac{1}{2}$ and the other less than $\frac{1}{2}$, so the price, and seller’s profit will be $\frac{1}{2}$. In probability of $\frac{1}{4}$ both will bid more than $\frac{1}{2}$, and in that case the expected value of the lower bid will be $\frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2} = \frac{3}{4}$. In all, the expected price is $(\frac{1}{2})^2 + \frac{1}{4} \cdot \frac{2}{3} = \frac{5}{12}$ which is grater than the expected price in normal second bid auction, which we showed last lesson equals $\frac{1}{3}$.

To conclude - a Mechanism that does not seek to maximise social profit, can get grater profit for the seller.
2 Combinatorial Auction

Up to this point, we handled only auction in which each item was sold separately, and had it’s own value. We will now try to expand our discussion to a more complicated case, in which groups of items are sold, and each group of items has a value for each bidder. Let us start with some formal definitions:

Let \( \{1..m\} \) be a group of items being sold.

Let \( V_i : 2^m \rightarrow \mathbb{R} \) be the valuation function for player \( i \), where \( V_i(S) \) is the value for player \( i \) from getting group of items \( S \subseteq \{1..m\} \)

We shall make two assumptions about the valuation functions:

- Normalisation, i.e. \( V_i(\emptyset) = 0 \).
- Monotonicity (“free disposal”), i.e. \( S \subseteq T \Rightarrow V(T) \geq V(S) \)

Now, what should be the relation between \( V(S \cup T) \) and \( V(S) + V(T) \)?

The most natural is for the two to be equal, but in that case it’s not fun - we get a normal auction of multiple items. The more interesting cases are:

- \( V(S \cup T) \geq V(S) + V(T) \) - the whole is greater than the sum of its parts - in that case we call the items complements.
- \( V(S \cup T) \leq V(S) + V(T) \) - the whole is less than the sum of its parts - in that case we call the items substitutes.

Generally, we can formalise the problem of combinatorial auction as \( k \) players, group \( \{1,.., m\} \) of items, and each player has a function \( V_i : 2^m \rightarrow \mathbb{R} \) that donates for him a value for each subgroup of items. Our objective is to maximise the social welfare, given by \( \sum_{i} V_i(S_i) \), where \( (S_1, S_2, .., S_n) \) is a division of the group \( \{1,.., m\} \).

2.1 Example 1 - Landing and Takeoff slots in Airports

In big airports, landing and takeoff slots are valuable. The airport has limited capacity, and in any given time only a small number of airplanes can land or takeoff. The problem is how to distribute the slots between the various companies using the airport. At first, the value of the slots was not recognised and they where divided between the airliners, but not only profit was lost, it was also very hard to distribute the slots - a company cannot use a landing slot in airport A if it does not have a takeoff slot in airport B in an earlier time. When the value of the slots was recognised, the airports started to sell them, each slot individually, but that made a really big mess, because of the reason mentioned. The solution was to use combinatorial auction - each company does not have a value for a slot but of each group of slots.
2.2 Example 2 - Radio frequencies in USA

This is an example we already mentioned a few times in this course - the FCC sells licenses to use given radio frequencies in given areas. The pager and cellular companies buying the frequencies do not have value for one area only - they need many areas in order to give adequate coverage. Again - this is combinatorial action, although in reality only a few times a real combinatorial auction was held in that case.

2.3 Example 3 - Procurement auction

A company has many suppliers. Each supplier can supply the company with a group of items, that the company needs. For example - if the company needs to transport goods from point A to point D, it needs a transporter that can move the goods between each pair of points - A to B, B to C, etc. The company must achieve all the line, but has no use for two transports from A to B. Each transporter company can supply some of the lines. In reality, there are only a few companies that make their business by doing combinatorial auctions in such cases, and the results reported are extraordinary - a cut of 10%-20% in buyers’ costs while rising the suppliers profit!

2.4 Example 4 - Generalisation of network traffic problem

Let’s say we have three people who want to send a message over a computer network. The network can be described as a graph, where the minimal cut of the graph is the limit on the number of messages that can be sent at a given time. Each person has some positive value for sending the message from his position to his destination, and zero for every other route in the network. The group of edges in the graph are the goods to be sold, and we want to maximise the value of messages that pass in the network.

2.5 What is hard in combinatorial auctions?

- Exponential Input: The input of the valuation functions $V_i$ is exponential in $m$ (the number of items). This means that the input is actually unknown - the players themselves cannot know their own private values, and even if they somehow knew, they have no way of telling it to us. Even if there was some magic black box that can give the valuation for a given situation, it would still take an exponential number of questions for us to achieve the information.

- The problem is computationally hard - It is an NP complete problem, as we will show soon, and what is more, it is even hard to get a decent approximation.

- Strategy: Beyond the previous two “new” problems we have the “normal” auction problem of designing an incentive compatible mechanism.
3 Single minded combinatorial Auction

The problems presented above are indeed hard problems, but for now we will try to solve them the easy way - an simple instance that is easier to solve. Such is the case of “Single minded combinatorial auction”. Here, each player is interested only in one subgroup of items, for which she has positive value, and for any other item group (that does not contain the wanted subgroup) she has zero value.

Definition 1 A player is called “single minded” if:

There exists \( V^* \) and \( S^* \) such that \( V_i(S) = \begin{cases} V^* & S^* \subseteq S \\ 0 & \text{o.w.} \end{cases} \)

With this definition we got rid of the exponential input problem - we need only to describe \( O(m) \) bits of input for each player. Note that this is a pure case of complement items - only the whole group of items has a positive value. The problem we now have is:

Input: \( V_1, \ldots, V_n, S_1, \ldots, S_n \) (w.l.o.g we will dispose items ourselves)

Output: Find a group \( W \subseteq \{1, \ldots, n\} \) such that \( i \neq j \in W \Rightarrow S_i \cap S_j = \emptyset \), that maximises the value of \( \sum_{i \in W} V_i \)

Lemma 2 The problem is NP Complete

Proof: Reduction from Clique. We will show how to build a single minded combinatorial auction from a Clique problem. Each node in the graph will represent a player, and each edge will represent an item sold. We define (given a graph \((V,E)\)):

\( V_i = 1 \)

\( S_i = \{j \mid (i,j) \in E\} \)

Note that we have reduced the problem to the case where only 2 players want every item. When we will solve the combinatorial auction, we will get: | Max independent subgroup | = Maximum social welfare. The solution of the auction will give us the clique, where the winners in the auction will be the vertices in the clique.

Theorem 3 For each \( \epsilon \geq 0 \) it is NP Hard to approximate Clique problem up to a factor of \( N^{1-\epsilon} \).

Corollary 4 Our problem is NP Hard to approximate up to a factor of \( O(m^{\frac{1}{2}-\epsilon}) \).
3.1 Strategies

What can we do now? we can try to solve it anyway, and surprisingly - in almost all real world cases we can actually do so, whether by proofed algorithms for specific cases, or by heuristics.

Theoretically we can ask if there is an $O(m^{\frac{1}{2}-\epsilon})$ approximating algorithm. If we can find such an approximation, it will probably work much better in reality, and with some heuristics we can even improve it more.

After we have “Solved” the problem of complexity, we shall now approach the mechanism design problem. How can we design an incentive compatible mechanism? VCG will not work in this case, because we can’t even calculate the cost each player causes to society. We note, that by using approximations to VCG we will get a mechanism that is not an honest one (the VCG proof was based on exact calculations of the cost).

3.2 LOS Mechanism

That leaves us with the need for a new mechanism. Such mechanism is LOS (Lehman, O’Callaghan, Shoham), and the algorithm it uses is the following:

1. Order the buyers by descending order of $\frac{V_i}{\sqrt{|S_j|}}$
2. Greedily, insert to $W$ every player possible ($i$ such that $S_i \cap S_j = \phi$ for every $j \in W$).
3. The payment that player $i$ must pay is $\frac{V_i\sqrt{|S_i|}}{\sqrt{|S_j|}}$, where $j$ is the first player who did not win only because of $i$.

Before we treat this algorithm formally let’s have some intuition about it: Obviously, we would prefer a player with a large $V_i$ to win. However, there might be a situation the large $V_i$ will not be worthwhile because of a very large group of items. Thus, we will take into consideration both value and group size.

Now formally:

**Lemma 5** LOS is incentive compatible

We will leave the proof for later on.

**Lemma 6** LOS is $\sqrt{m}$ approximation

**Proof:** Let us compare the optimum result (OPT) with LOS result (ALG). Each algorithm gives us a group of winners. What is the connection between the groups? Each player
who won in OPT and did not win in ALG has another player in ALG that prevented him from winning.

Formally, \( \forall i \in (OPT - ALG) \exists \text{ (first) } j \in ALG \text{ s.t because of } j \ intervenes \).

What do we know about that \( j \)?

- \( S_i \cap S_j \neq \phi \).
- \( \frac{V_i}{\sqrt{|S_i|}} \leq \frac{V_j}{\sqrt{|S_j|}} \)

**Claim 7** \( \forall j \in ALG \sum_j \text{ is responsible for } i \) \( V_i \leq \sqrt{m} V_j \)

**Proof:**

\[
\sum_j \text{ is responsible for } i \ V_i \leq \sum_j \frac{V_j}{\sqrt{|S_j|}} \sqrt{|S_i|} = \frac{V_j}{\sqrt{|S_j|}} \sum \sqrt{|S_i|} \\
= \frac{V_j}{\sqrt{|S_j|}} (\sqrt{|S_1|}, \sqrt{|S_2|}, ..., \sqrt{|S_t|})(1,1,...,1)
\]

We know that:

- \( \sqrt{(1,1,...,1)} \leq \sqrt{t} \leq \sqrt{|S_j|} \)
- \( |(\sqrt{|S_1|}, \sqrt{|S_2|}, ..., \sqrt{|S_t|})| \leq \sqrt{\sum |S_i|} \leq \sqrt{m} \)

End so we achieve:

\[
\sum_j \text{ is responsible for } i \ V_i \leq \frac{V_j}{\sqrt{|S_j|}} \sqrt{m} \sqrt{|S_j|} = \sqrt{m} V_j \]

From here we easily derive that LOS is indeed a \( \sqrt{m} \) approximation algorithm!

8-6