15.1 Introduction

Today is the first part of the last section of the class, on algorithmic mechanism design. Like we did with computation of equilibria and with inefficiency of equilibria, today will serve as a general overview of what we're going to cover, without doing too much technical work. But we will give some of the most important definitions, since they're not all that technical anyway, which will let us dive into more technical stuff on Thursday.

Mechanism design is in some ways a weird fit for "algorithmic game theory", since we're rarely going to be talking about games in the sense that we've been talking about them. But for a variety of reasons it is usually considered part of AGT, even though in the economics community 'mechanism design" and "game theory" are usually thought of as somewhat separate things. These days, mechanism design is probably the largest subarea of AGT.

At a very high level, mechanism design is the question of how to design systems (algorithms) with strategic participants with good performance guarantees. In some sense, it's the "design" version of inefficiency of equilibria: instead of analyzing games, we want to *design* games so that rational agents will result in good global behavior. However, we don't necessarily think of the setup for mechanism design as a "game" in the way that we have been (even though it is!). Instead, the simplest setting to think about are *auctions*. We'll start with the simplest version possible, and later will move to more complicated versions.

15.2 Single-Item Sealed Bid Auction

Instead of describing a game, I'm going to describe what should be a pretty familiar setup for an auction.

- Setup: We, the auctioneer, are selling a single item. There are n players/bidders who are interested in it. Each player i has a private valuation v_i for the item (unknown to us or the other players)
- Auction: each player sends a private bid b_i to us (the auctioneer). We look at all the bids, and decide two things:
  1. Who gets the item, and
  2. How much to charge them for it (price p).
- After these decisions have been made the utility of player i is 0 if i does not get the item, and $v_i - p$ if i does get the item. This is called the *quasilinear utility* model.

So our main question is: what should the auctioneer do? Given bids, who should we give the item to and at what price? Note that this is indeed a game like we've been talking about: the strategies

are the possible bids, and our method of choosing who to give the item to and at what price gives some utility function to each player. But our goal isn't to analyze the equilibria, but instead to figure out the rules of the game.

We'll formally talk about our goal later, but informally let's suppose that we're benevolent – we want to give the item to the player who values it the most (i.e., we're not necessarily trying to maximize our revenue).

Option 1: Since we're being benevolent, why not just give it to the highest bidder at price 0? This would work fine if players were not strategic, but since they are, this is a very silly mechanism. It boils down to a game of "who can name the largest number", since what we bid doesn't need to have anything to do with our actual valuation. So we can't reason about what will happen, and in particular there's no reason to think that the person with highest valuation will also be the person who gets the item (has the highest bid).

Option 2: The obvious way to fix this is to change the price, and in particular to set $p = \max_{i \in [n]} b_i$ and give the item to $\arg\max_{i \in [n]} b_i$. After all, if you put in a bid of b_i, then you should be willing to pay what you bid! This is called a "first-price auction".

What happens in the first-price auction? Suppose that bidder i sets $b_i = v_i$. Then no matter what, bidder i has utility 0, whether they win the auction or not. So bidder i will not bid v_i, but will instead try to "underbid" some $b_i < v_i$. How much will they try to underbid? If they believe that they are the bidder with highest valuation, then they will try to bid as little as possible while still having the maximum bid. So they will try to "guess" what the second highest valuation is, guess how much that player will underbid, and then bid just a little bit more. That's a lot of guesswork, and if the player is wrong about any of it, there's the risk that they won't actually be the highest bidder, so we as the auctioneer will have given the item to the "wrong" bidder.. Are we willing to take that risk? What do we believe about the beliefs that the bidders have about each other? There are ways of talking about this, but it's very hard and delicate. Is there any way we can change our auction so that we can be confident the person with the highest valuation will actually get the item?

Option 3: Let's use the same assignment, so the player who gets the item is $\arg\max_{i \in [n]} b_i$, but let's change the price. Instead of setting the price to the highest bit, let's set the price to the *second* highest bid. This is known as a *second-price* auction, or a *Vickrey* auction.

Theorem 15.2.1. *For all $i \in [n]$, bidding $b_i = v_i$ is a dominant strategy for player i.*

Proof. We want to show that for any bid vector b_{-i} which does not include the bid of player i, then player i will maximize their utility by setting $b_i = v_i$. Thus no matter what other players bid, the best thing for player i to do is bid its true valuation, so that is a dominant strategy.

Let $B = \max_{j \neq i} b_j$. Let's split into two cases.

- Case 1: $v_i < B$. Then setting $b_i = v_i$ will result in utility 0, since i won't get the item. Setting $b_i \leq B$ will also result in utility 0, while setting $b_i > B$ will result in utility $v_i - B < 0$. Thus the best thing for player i to do is bid v_i.

- Case 2: $v_i \geq B$. If $b_i \geq B$ then player i will get utility $v_i - B \geq 0$, and if $b_i < B$ then player i will get utility 0. Thus bidding v_i results in the maximum possible utility.

Thus no matter what, the best thing for player i to do is bid v_i, so bidding v_i is a dominant strategy. □

So in a second-price auction everyone will bid truthfully, so we will actually give the item to the player who values it the most!

There's one more subtlety that's easy to take care of: it's also important that bidders actually want to participate in the auction, since we can't force anyone to bid! So we also want the following easy theorem.

Theorem 15.2.2. *In a second-price auction, every truthtelling bidder is guaranteed nonnegative utility.*

Proof. The losers have utility 0, while the winner i has utility $v_i - p \geq 0$, since player i bid $b_i = v_i$ and the price is $p \leq b_i$. □

We will combine these two properties into possibly the single most important definition.

Definition 15.2.3. *A mechanism is* incentive compatible *(or* DSIC *for "dominant strategy incentive compatible", or* truthful*) if:*

1. *Truthful bidding is a dominant strategy for all players, and*
2. *No player regrets participation (no player ever has negative utility).*

Now we can note three particularly nice properties of second-price auctions:

Theorem 15.2.4. *Second-price auctions have the following properties:*

1. *Incentive compatible*
2. *If players bid truthfully, it maximizes* social surplus $\sum_{i=1}^{n} v_i x_i$*, where x is the set of possible outcome vectors (one entry is* 1 *and the others are all* 0*).*
3. *It can be implemented in polynomial time (in fact, linear time).*

As a side note, this notion of social surplus is equal to the sum of utilities if we also think of the auctioneer as a player, since the winner i gets utility $v_i - p$ and the auctioneer gets utility (revenue) p. This is similar to what we talked about for the facility location game.

All three of these properties are important.

- Incentive compatibility is important because it makes it easy for the bidders to play (they can just tell the truth) and easy for the auctioneer to reason about what will happen.

- Maximizing social surplus is important since, at the end of the day, that's our actual goal. If we just cared about incentive compatibility, we could give away the item for free to either a fixed or a random bidder.
- Computational efficiency is obviously important – we're computer scientists!

Much of algorithmic mechanism design is building auctions with these three properties (what Roughgarden's book calls "awesome auctions") for more complicated settings. The "algorithmic" part of algorithmic mechanism design means that computational efficiency is something that we actually really care about. For classical economic mechanism design, computational efficiency is not particularly crucial, and in fact some of the most foundational results in mechanism design are *not* computationally efficient. So we're going to study what happens when we're not willing to compromise on computational efficiency.

15.3 Sponsored Search Auctions

Like we did with games, we're going to particularly focus on settings that arise from large computational systems. For example, one of the most important settings from 10-20 years ago was *sponsored search auctions*. Search engines needed a way of auctioning off ad spots in searches. We can somewhat formalize this as follows:

- There are k items $[k]$: these are slots where ads can be placed.
- The click through rate (CTR) of slot j is α_j. We'll assume that $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_k$ (slots that are higher up are better than slots that are further down), and will also assume that the click through rate is independent of the actual ad, and just depends on the slot.
- The n bidders are advertisers who have bid on some keyword
- Bidder i gets benefit of v_i *per click*, which means that its utility if its ad gets placed in slot j is $\alpha_j v_i$.

This is clearly more complicated than the single-item auction: there are multiple items which are not identical, rather than just one item. But there is an important similarity: in both of them, each bidder has just a single private parameter v_i.

We want to design an awesome auction (incentive compatible, surplus maximizing, computationally efficient) for sponsored search. What does this actually mean for sponsored search?

- Incentive compatible is basically the same as single-item, since there's still only one private parameter. For each bidder, we want it to be the case that telling v_i to the auctioneer is a dominant strategy.
- Surplus maximizing means that the auctioneer chooses the best assignment: out of all injective functions $f : [k] \to [n]$, we choose the one that maximizes $\sum_{j=1}^{k} \alpha_j v_{f(j)}$ (note that it is crucial that f is injective, since the same ad can't be placed in two different slots).

- Computationally efficient will mean polynomial running time.

Our approach will be to separate the two things we need to decide: the assignment and the prices. We'll first figure out the right assignment (to get surplus maximization), and then figure out the prices to get incentive compatibility. Then we'll check that both of these computations can be done efficiently.

So we have two questions:

Question 1: *Assuming* truthful bids, what should the assignment be?

Question 2: Given the answer to question 1, how do we set prices to make the mechanism incentive-compatible?

Question 1 is actually pretty easy: a greedy algorithm maximizes the surplus: assign slot 1 to the highest bidder, slot 2 to the second highest bidder, etc. It's straightforward to check that this does maximize the surplus (assuming all bids are the true valuations).

Question 2 is much more complicated, and is what we'll focus on next class. In fact, we'll generalize beyond sponsored-search auctions to something called "single-parameter environments", which are the setting in which each bidder has only a single private number. We'll see that we can in fact find a payment rule to make the surplus-maximizing assignment incentive-compatible, but that it's much more complicated than the simple second-price rule we used in the single item setting.