601.436/636 Algorithmic Game Theory **Lecturer:** Michael Dinitz
Topic: Online Auctions **Date:** 4/28/20
Scribe: Michael Dinitz

24.1 Introduction

So far all of our mechanism have been "one-shot": we run an auction by soliciting bids and then deciding on an outcome and prices. This is intuitively a pretty good model of what we think of as an "auction". But let's think of another setting: we're selling items in a store. Then players will enter our store, make us an offer, and leave the store. Clearly if we sell an item to a player then we cannot later sell the same item to someone else, and if a player leaves the store then we cannot afterwards sell them the item. So our decisions are *irrevocable*. This is a classical setting for online algorithms, so today we'll be considering online mechanisms which combine online algorithms with incentives.

24.2 Example

Suppose that we're selling two identical items. Each bidder wants one item (they don't care which) and has a valuation for how much they value an item (the same value for both, since they're identical), but also has an arrival and departure time. Suppose that there are three bidders. Bidder 1 has a valuation of 100, arrival time of noon (12pm), and departure time of 2pm. Bidder 2 has valuation 80, arrival time of noon, and departure time of 2pm. Bidder 3 has valuation of 60, arrival time of 1pm, and departure time of 2pm.

If we were in the old setting, without arrival and departure times, it's pretty easy to use VCG to figure out what to do. The social welfare-maximizing allocation is to give one item to each of the first two bidders, and charge each of them 60 (the valuation of the third bidder). This is because the externality caused by each of the two players who get an item is the damage they do to the third bidder, which is 60.

What's the natural thing to do in the online setting for this example? One idea would be to basically sell one item each hour, say at the end of each hour. If we do that, then at 12:59pm we'll hold an auction with just the first two bidders, and then at 1:59pm we'll hold an auction with all three bidders. So we'll sell one item to bidder 1 for 80 (Vickrey auction), and then sell one item to bidder 2 for 60 (second Vickrey auction, since bidder 1 will no long be bidding or will equivalently have valuation 0 since it already received an item). This looks great – our revenue went up from 120 to 140!

Unfortunately, this is not incentive compatible. Depending on the exact format of the bids, there are a few ways the bidders could lie to get an advantage. For example:

- Bidder 1 could like about their valuation, by bidding (for example) 65. Then it won't win the first auction, but will still win the second auction and so will still get an item. And now it only has to pay 60 instead of 80.

- Bidder 1 could delay their arrival to 1pm. Then in the first auction bidder 2 will win for price 0, and in the second auction bidder 1 will win for price 60. So bidder 1 is better off.

Can we design mechanisms that take time into account that are still incentive compatible and do well on social welfare and/or revenue? That's what we're going to focus on today.

24.3 Setup

- Each agent $i \in [n]$ has an arrival time a_i, departure time d_i, and value v_i (all of which are private and are nonnegative real numbers).
- We have $k \geq 1$ identical goods to sell (for most of today we'll just have $k = 1$).
- A bid is a triple (a'_i, d'_i, b_i) which is supposed to be the three hidden parameters of the agent. Note that the agent may lie about all three of these. And since we're in an online environment, this bid is not revealed to the mechanism (auctioneer) until time a'_i.
- An online allocation rule maps bids to *schedules*: who gets an item and at what time.
- An online payment rule gives prices for each agent which are collected when they receive an item.
- Incentive compatibility here means that bidding truthfully is a dominant strategy (and that bidding truthfully never results in negative utility).
- The utility of an agent i is v_i- price it pays if i gets the item in $[a_i, d_i]$, and is 0 otherwise.

We're going to make a few additional assumptions as well.

- $a'_i \geq a_i$: you cannot claim to arrive before you're actually there, since you'd have to be there to make such a claim.
- The valuations are drawn i.i.d. from some unknown distribution (i.e., this is prior-free).

There are two natural goals that we can try to achieve.

1. Welfare maximization. If we're not online and we're selling one item, then we can actually achieve the maximum possible surplus $\max_{i \in [n]} v_i$ by using a Vickrey (second-price) auction. Can we get close to this in the online setting?
2. Revenue maximization. Since we want to be prior-free, we can't actually set a reserve price appropriately for a true revenue-maximizing auction, even offline. But Bulow-Klemperer tells us that the Vickrey auction still has interesting guarantees on its revenue. So again we'll try to compete with the Vickrey auction, which has revenue equal to the second highest valuation. Can we get revenue close to this online?

24.4 Simple setting

Let's start with a super simplified setting: all arrival-departure intervals are disjoint, and we're not going to care about incentives. So this is just a pure algorithmic question, and there is never more than one agent at a time. We just see the agents one at a time, when we see them they tell us their value, and we have to decide whether to give them the item. Our goal is to give the item to whoever has the highest value.

This should look familiar to you – it's *almost* the setting of the prophet inequality from Lecture 20. However, for the prophet inequality we allow the bidders to have valuations from *different* distributions that we *know*. Here all valuations are from the *same* distribution (making it easier for us) but we do not know this distribution (making it harder). We instead of using ideas from the prophet inequality, we're going to use ideas from the *secretary problem*.

The secretary problem is defined as follows. We interview n job applicants in *random order*. When we interview a candidate we find out their value and then need to immediately make an irrevocable decision whether or not to hire them. So here we're assuming that the valuations are worst-case (adversarial), but then the order they appear in is random. This is different from our auction setup, where the values themselves are random but the ordering is fixed, but it's not too hard to show that the random order, adversarial value setting is at least as hard as the random value, arbitrary order setting. So if we can design algorithms for the secretary problem, they'll also work for our auction problem.

Theorem 24.4.1 (Dynkin '62)**.** *There is an algorithm which chooses the maximum value applicant with probability at least* $1/e$ *(where the probability is over the random ordering).*

Proof Sketch. We're not going to prove the $1/e$ bound today, since it's a little involved. We'll do something much easier and give a bound of $1/4$.

Our algorithm is the following. We do not hire any of the first $n/2$ applicants we see, no matter what their value is. We let p equal the largest value we've seen so far (the largest of the first $n/2$). We then hire the next applicant who has value at least p.

This algorithm will definitely return the most valuable element if *second* most valuable element is in the first $n/2$ elements and the most valuable element is in the last $n/2$ elements, since then we will set the threshold p to the second largest value and the best applicant is still available. This happens with probability at least $1/4$ (it would be exactly $1/4$ if the two events were independent, but it's actually slightly better than $1/4$ because if we condition on the second most valuable element being in the first half we have that the probability that the most valuable element is in the second half is $(n/2)/(n-1) > 1/2$). □

To get the true $1/e$ bound we have to change the sample from the first $n/2$ to the first n/e and do some additional analysis to give ourselves credit for other cases when we still select the best (e.g., if the third best is in the sample but the top two are not, but the best comes before the second best in the ordering).

24.5 Online Auction for One Item

Let's go back to our actual setting where intervals can overlap and there are incentives, but where we're still selling just one item. Our analysis of the secretary problem suggests an obvious auction: wait until we've received $n/2$ bids (at some time t), set p to be the largest bid seen so far, and sell the item to the next bidder who bids above p. Unfortunately, this isn't incentive compatible. To see this, let i be the $n/2$'nd player to arrive (if everyone tells the truth), so $t = a_i$, and suppose that i has the largest valuation and that there is some i' such that $a_i < a_{i'} < d_i$. Then if player i tells the truth it will be in the sample, and so will always get utility 0. But if it lies about its arrival time and arrives a bit late, in particular at time just after $a_{i'}$, then it will not be in the sample and so will get positive utility (since it will definitely get the item since it is the high valuation and is the first bid after the sample).

Fortunately, this isn't too hard to fix. We just need some way of dealing with this bad case, where the bidders overlap right around time t. We'll use the following mechanism.

- Let t be the time when we receive the $n/2$ bid.
- Let $p \geq q$ be the *two* largest bids so far.
- If at time t there is an active bidder (a bidder whose interval contains t, i.e., is currently available) with bid p, then sell to that bidder at price q.
- Otherwise, sell to the next agent with bid at least p at price p.

24.5.1 Incentive Compatibility

Fix a player i and all other bids. Since a bid consist of a trip (a'_i, d'_i, b_i), we need to show that bidding (a_i, d_i, v_i) is a dominant strategy. Let's handle these three parts one at a time.

Departure Times. This is obvious. If i sets $d'_i < d_i$ then all they're doing is leaving earlier than the could, which clearly cannot help them. And if $d'_i > d_i$ then this is only different from $d'_i = d_i$ if they get the item at some time after d_i, in which case they get 0 utility. So it's always best to set $d'_i = d_i$.

Values. This is also pretty simple. The key property we'll use is that if i wins the item, then the price they pay is *independent* of their bid b_i. In other words, any bid which wins results in the exact same price. So if $b_i < v_i$, this does not affect the price and either player i still wins (in which case it has the same utility as if it bid v_i) or it loses (in which case it will have negative utility). Similarly, if $b_i > v_i$ then player i will still win but will be charged the exact same price, so will have the same utility as it would if it had bid v_i. Thus bidding v_i is a dominant strategy.

Arrival Times. This is the trickiest part: we have to show that it does not help i to lie about its arrival time. Note that $a'_i \geq a_i$, since we assumed that in order to claim you've arrived you have to actually be there.

Out of all the bids *other* than i, let r be the $n/2 - 1$ arrival time and let s be the $n/2$ arrival time. This means that t is either r (if $a'_i < r$) or a'_i (if $r < a'_i < s$) or s (if $s < a'_i$). As always, we will assume that bidding a_i' results in winning the auction, since otherwise telling the truth is clearly at least as good.

If $a_i > s$, then clearly lying cannot help: it only can push i later in the ordering when it already was after t, so there it cannot affect the price and there is no way that it can win the auction if it wouldn't have won at a_i. So in this case player i has no incentive to lie.

If $r < a_i < s$, then if i tells the truth $t = a_i$ and the auction will function in the exact same way for any $a_i < a_i' < s$, so we only have to worry about a lie with $a_i' > s$. But if i wins with $a_i' > s$ then it would also win with $a_i' = a_i$ (since it must be the highest bid of the sample and be active at time s) and the price it is charged can only increase. Hence again telling the truth is dominant.

If $a_i < r$, then any $a_i' < r$ makes no difference. If $r < a_i' < s$, then t changes from r to a_i', but in either case the values of p and q are the same at t and so i will also win if it tells the truth and will have the same payment. If $a_i' > s$ then again player i will win if it tells the truth (since it must be the high bid) and now it is charged a higher price (p rather than q), and so it is worse off. Hence telling the truth is a dominant strategy.

24.5.2 Objective Values (Competitive Ratios)

Let's analyze both social welfare and revenue.

Theorem 24.5.1. *The agent with maximum valuation wins the item with probability at least* $1/4$.

Proof. We divide into two cases.

Case 1: if the item is sold at time t, then it is sold to the highest bidder of the first half, which is the highest value over all with probability $1/2 > 1/4$.

Case 2: If the item is sold after time t, then the analysis is just like the secretary problem! With probability at least $1/4$ the second-highest bidder appeared by time t and the highest appeared at time after t, in which case we sell the item to the highest bidder. □

Theorem 24.5.2. $\mathbf{E}[\textit{Revenue}] \geq \frac{1}{4}\mathbf{E}[\textit{Revenue of Vickrey}]$.

Proof. Let's do the same thing as before and break into the same two cases.

Case 1: if the item is sold at time t, then it was sold to the bidder i with the largest value in the first half. With probability $1/4$, this bidder i is the largest bidder overall *and* the second-largest bidder overall is also in the first half. If this happens, then the revenue is exactly the same as Vickrey, so our expected revenue is at least $\frac{1}{4}$ of the expected revenue of Vickrey.

Case 2: if the item is sold after time t, then again it's just like the secretary problem: with probability $1/4$ the largest value is in the second half and the second-largest value is in the first half, so with probability $1/4$ we have the same revenue as Vickrey. □

24.6 Generalization to k Items

Generalizing what we did from selling 1 item to selling k (identical) items is technically quite difficult, but conceptually it's not so bad. The mechanism has three phases.

1. "Learning" phase: do nothing until time t, the arrival of the $n/2$ agent.

2. "Transition" phase: At time t, sell up to $s = k/3$ items to agents that are active at time t and who have bid above the $(k/3 + 1)$-highest bid so far.

3. "Accepting" phase: After time t, set p to be the s'th highest bid in the first half, and sell items at price p while supplies last.

We're not going to analyze this, because it's a bit too difficult for this course, but the intuition should be clear.

24.7 Extensions

There are a lot of extensions to this basic framework, some obvious and some not. What if the items are not identical? What if we're in a combinatorial auctions setting? Less obviously (but important in many practical settings), what if we have "time discounts", e.g., it's better to sell earlier than later? For many of these settings we have decent *algorithms*, but there are still many open questions about how to turn those online algorithms into incentive compatible mechanisms.