Algorithmic Game Theory **Fall 2019**

Lecture 3: Knapsack Auctions and Welfare Maximization

Scribes: Debayan Bandyopadhyay 10/2/2019

3.1 A Quick Review

Recall that we wanted the following characteristics for any auction, which we called an **Awesome** auction.

1. DISC (defined below)
2. Maximizing the social welfare, $\sum_i v_i a_i$, assuming truthtelling bidders
3. Computationally efficient procedure

We also defined DSIC (Dominant Strategy Incentive Compatible) to mean the following:

1. Every bidder has a dominant strategy
2. Every truthtelling bidder has nonzero utility

The first is important for us to reason about the auction. The second allows bidders to participate without concern that the rules will adversely affect them.

We also discussed **Myerson's Lemma**, which is used in **single parameter environments**, that is, environments where there is only one relevant parameter, the valuation of a particular item. Myerson's Lemma concerns n bidders with private valuations v_i, with some feasible allocation set A, from which allocation vectors $(a_1, a_2, \ldots, a_n)$ are chosen. For example, in a single item auction, the feasible satisfies $\sum_i a_i \leq 1$, whereas in a k item auction, $\sum_i a_i \leq k$. There are three conclusions we made:

1. $\vec{a}$ is implementable if and only if the allocation rule is monotone
2. There is a unique payment rule that is DSIC.
3. $p_i(\vec{b}) = \sum_j z_j \cdot$jump in $a_i(\vec{b})$. The generalization for this in the continuous case is $p_i(\vec{b}) = \int_0^{b_i} z \cdot a_i'(z, \vec{b}) dz$.

3.2 Knapsack Auctions

3.2.1 The Knapsack Auction

Knapsack auctions are single parameter environments that are slightly different from k item auctions. We have two additional concerns with this new type of auction.

First, every bidder has some size, w_i. Second, we have a capacity W, which means $\sum_i w_i a_i \leq W$. The analogy here is Super Bowl commercials. Every commercial has some length, and we need to take those lengths into account given that we can't play all of them in the time that we have. We can actually reconstruct a k item auction in this type of auction by setting $w_i = 1$ for each bidder and $W = k$ for the capacity.

Let's try to design an awesome knapsack auction! Roughgarden provides a two-step dance to addressing this issue. First, we want to assume that every bidder will truthfully provide their valuation, that is, $b_i = v_i$ and then look to maximize welfare. Second, we then devise payment rules to be DSIC.

How do we maximize welfare? Well, noting that welfare is $\sum_i v_i a_i$ and then using our assumption that $b_i = v_i$, we want to find the allocation rule that satisfies $\vec{a}(\vec{b}) = \arg\max_A \sum_i b_i a_i$. This is an instance of the Knapsack problem from CS170! (You're a grave robber trying to make as much money for yourself as you can, but you can only take the jewels that fit in your knapsack. How do you choose the jewels based on their weight/size and value such that you maximize the value of your theft?)

How about payment?

Note that our rule is monotone. If a particular person values the item more, they bid more, and as a result, are MORE likely to be given nonzero allocation. Monotonicity implies that the allocation jumps from 0 to 1 at some value for the valuation, z. Myerson's Lemma says that our payment rule will be $p_i(\vec{b}) = \sum_j z_j \cdot \text{jump in } a_i(\vec{b})$, so factoring in the one jump, we get for bidder i is zero up to the valuation point z after which it is z.

The issue with this is that the problem is in general intractable. Knapsack doesn't run in polynomial time, and as Jonathan pointed out during the lecture, we have to run Knapsack several times for us to run our auction.

Let's revisit our requirements, of which we had three. First, we wanted the auction to be DSIC. Second, we wanted to maximize welfare. Third, we wanted to be poly-time efficient. The two rules that are odds with each other here are the maximization of welfare, and the polynomial runtime used to design the auction. Thus, we shouldn't relax the first constraint.

There are cases in which it makes sense to relax the third constraint, for small or structured enough instances. However, algorithmic mechanism design hopes to relax the maximizing welfare condition and achieve the other two requirements for an awesome auction.

In essence, we want to design algorithms which are polynomial time with monotone allocation rules, which come close to maximizing welfare.

3.2.2 Knapsack, A Second Pass

We're going to consider the following rule:

1. First, sort and reindex the bidders such that we maximize the ratio of their valuation to their weight. In other words, we want to order and reindex such that $\frac{v_1}{w_1} \geq \frac{v_2}{w_2} \geq \ldots \frac{v_n}{w_n}$. Given that we have truthtelling bidders, we just use their bids directly to reorder and index such that $\frac{b_1}{w_1} \geq \frac{b_2}{w_2} \geq \ldots \frac{b_n}{w_n}$.
2. Pick bidders to allocate in order until we can no longer accommodate another bidder.
3. Pick either the result of step (2) or the highest bidder, whichever has a larger total welfare value.

This allocation rule actually provides a half-approximation to the optimal allocation. That is to say, we get at least 50% of the total welfare obtained by using this rule as we do by optimally finding the best allocation.

Sketch: We consider what happens when we allow fractional allocation, wherein we are allowed to allocate some fraction of a full allocation, in exchange for the bidder only contributing that same fraction of their weight to the overall weight. For example, imagine that we allowed commercials to play for only 75% of their total length. Clearly, fractional allocation will always be more optimal than the nonfractional kind, because we can pick and choose fractions to match our capacity exactly. Suppose we ranked bidders in the same way that we did in our scheme, and in doing so, we took the first k bidders, as well as some fraction of the $k+1$st bidder. By an exchange argument (taught in 170!), this greedy scheme is actually optimal for fractional knapsack.

How does our greedy scheme for the nonfractional case do in comparison?

Well, in the first two steps that we take, we take all k of the bidders that the fractional problem can take. In the third step, we choose between those k bidders and the highest bidder, and the highest bidder has valuation at least as much as the $k+1$st bidder, of whom the fractional problem takes some fraction, because the highest bidder has valuation higher than all bidders. The fractional case achieves welfare composed of the total valuation of the first k bidders and the fraction of the $k+1$st bidder that we can accept. With our greedy nonfractional algorithm, we get at least as much as the first k bidders, and at least as much as the $k+1$st bidder (but not both). This means that our greedy algorithm for the nonfractional case achieves a total welfare equivalent to at least half of the welfare of the fractional case. But the fractional case is always going to do at least as well as the case in which we aren't allowed fractional allocations, so our greedy scheme does at least half as well as optimal for nonfractional knapsack problems. In fact, it can be shown that if every bid is only some α fraction of the total capacity, i.e. $w_i \leq \alpha W$, with $\alpha \in (0, \frac{1}{2}]$, then, we can achieve a value of total welfare which is $1 - \alpha$ as much as the optimal with our greedy scheme.

One thing to note is that our allocation scheme is still monotone! If you have a higher valuation, then you bid higher, and if you bid higher, then your order in the list can only go up. This gives you a better chance of being given a spot. Sometimes in polynomial time mechanism design schemes, we don't get monotonicity, but usually we can restructure our problem to get it back.

3.2.3 The Big Question

Is there a single parameter environment for which an approximate polynomial time algorithm beats out an approximation by a polynomial time AND monotone algorithm? Hopefully, we'd want the answer to be no so that all of our mechanism designs can be monotone, but we don't know the answer.

We can also ask if we can accomplish more in nonDSIC mechanisms than in mechanisms that are restricted to be DSIC. For that, we should look back at the assumptions of DSIC mechanisms.

1. Every participant has a dominant strategy.
2. Direct revelation: Truthfully reporting is the dominant strategy.

Can we satisfy the first assumption without satisfying the second? Yes. Suppose we ran a Vickrey auction where we first multiplied all the bids by 2. The winning bid would have to pay twice what they would have had to in a regular Vickrey auction, all bids held constant, so the dominant strategy of every bidder should just be to bid half of their valuation, such that we recover our Vickrey auction.

With this in mind, we can also ask if it's possible to take a scheme which has dominant strategies for every bidder and convert it into a scheme in which each bidder has a dominant strategy, but that dominant strategy is actually to tell the truth. The answer is yes. This is what we call the **Revelation Principle:** For every mechanism M for which every bidder has a dominant strategy, we can construct a mechanism M' for which the dominant strategy is for each bidder to bid their valuation.

The proof is a reduction (170 wooo!). Suppose $b_{(M,i)} = s_i(v_i)$ is the dominant strategy bid for any given bidder. Then, we can construct a scheme M' where we take every bidder's bid, run s_i on it, and then run M on the resulting bids. In other words, $b_{(M',i)} = v_i$, but we run M on $s_i(b_{(M',i)})$.

What this tells us is that truthfulness is not as important as there being an equilibrium of some sort. When we design mechanisms, we might as well design it to also have direct revelation baked in.

3.3 Welfare vs. Revenue Maximization

3.3.1 So, Why Welfare?

So far, we've looked at maximizing this quantity $\sum_i v_i a_i$, which basically says that we want to maximize the value provided to the bidders with the items that we auction off. Why do we consider this?

Here are two reasons why it intuitively makes sense for us to be doing this. First, we see that in many contexts it makes sense for us to maximize welfare. For example, the government doesn't care so much about revenue as it does with providing maximum welfare for its citizens (in theory, anyway). Second, assume you were a seller who didn't look to maximize welfare. Your unsatisfied customers would get swooped up by someone who was doing a better job of maximizing welfare in their mechanism design.

In addition, revenue often depends on inputs of the problem, but welfare does not. For example, let's say you worked with a one item, one bidder auction. In essence, this auction is take it or leave it. Let's say we were to set the price of the item to be r. What revenue would we get? Well, we would get revenue r if $v \geq r$ and 0 if $v < r$.

What would we do if we wanted to maximize welfare? What would we do if we wanted to maximize revenue? To maximize welfare, we might just set the price r to be 0. Then the value added to the bidder would always be v. To maximize revenue, we might do $r = v$, but we don't know what the valuation is before we hold the auction! In addition, this changes for different auction types and different input types (more or less bidders). Maximizing revenue is hard to reason about in a way that welfare is not, and that's why we choose to maximize revenue.

3.3.2 Average Case/Bayesian Analysis

Suppose we are still working in the single parameter environment that we've been working with, but now we want to try to maximize expected revenue. For that, we're gonna assume we have some information about the types of valuations that people are going to have. Otherwise, the problem is just hopeless.

With that in mind, let's suppose every person has some valuation v_i which is actually drawn from a CDF F_i with corresponding PDF f_i. Suppose this probability distribution only has support over the region from $[0, v_{max}]$. This is not unreasonable. People generally get nonnegative value from receiving items from auctions up to some reasonable limit, and each individual bidder will probably value certain types of items more than others. The only additional assumption we make here is that all these distributions F_i are independent from each other. This may be harder to satisfy because the bidders' opinions may sway each other. To approach this problem of maximizing our expected revenue, we say that we don't know the v_i before the bids are presented to us, but we do know the F_is.

What is our expected revenue? If we go back to our one item, one bidder auction and set our price to be r, then it'll be:

$$r \cdot (1 - F(r))$$

where the first term is the revenue of the sale, and the second term is the probability of the bidder. The r that maximizes this expected value is called the **monopoly price** of F. For example, if F is distributed as $U(0, 1)$, then r should be $\frac{1}{2}$. If instead we have two bidders and hold a Vickrey auction, the monopoly price we should set is $r = \frac{1}{3}$. (This, by the way, can be done as a brief exercise to test your understanding of the Vickey design and the monopoly price idea.)

A sidenote is that we could also use a reserve price, wherein we give it to the highest bidder, unless no one bids at or above the reserve price.

So let's get crackin' on this expected revenue. First, let's use the revelation principle to say that we might as well assume that the bidders are going to give us their truthful valuations. With that in mind, we want to calculate the total payment that all of our bidders make given the valuations they make. In other words, we want to find an expression for $\mathbb{E}_{\vec{v}}[\sum_i p_i(\vec{v})]$ such that it helps us maximize it.

Recall from Myerson's Lemma, $p_i(\vec{b}) = \int_0^{b_i} z \cdot a_i'(z, \vec{b}) dz$ if we have our payments dictated by the allocation rule. Let's fix the bidder, i and all other valuations, $\vec{v}_{-i}$, where this notation means the vector containing the valuations for all bidders except i.

$$\begin{aligned}
\mathbb{E}_{v_i}[p_i(\vec{v})] &= \int_0^{v_{max}} p_i(\vec{v})f_i(v_i)dv_i && (3.1)\\
&= \int_0^{v_{max}} \int_0^{v_i} z \cdot a_i'(z,\vec{v})dz f_i(v_i)dv_i && (3.2)\\
&= \int_0^{v_{max}} \Big(\int_0^{v_{max}} f_i(v_i)dv_i\Big) z \cdot a_i'(z,\vec{v})dz && (3.3)\\
&= \int_0^{v_{max}} \big(1-F_i(z)\big) z \cdot a_i'(z,\vec{v})dz && (3.4)\\
&= \Big(\big(1-F_i(z)\big) z \cdot a_i(z,\vec{v})\Big)\Big|_0^{v_{max}} - \int_0^{v_{max}} a_i(z,\vec{v}) \cdot \big(1-F_i(z)-zf_i(z)\big)dz && (3.5)\\
&= \int_0^{v_{max}} \Big(z - \frac{1-F_i(z)}{f_i(z)}\Big) a_i(z,\vec{v})f_i(z)dz && (3.6)\\
&= \int_0^{v_{max}} \phi_i(z)a_i(z,\vec{v})f_i(z)dz && (3.7)
\end{aligned}$$

Note that we used change of bounds from (3.2) to (3.3), and then used integration by parts from (3.4) to (3.5). (Do it as an exercise if that pleases you.)

We defined $\phi_i(v_i) = v_i - \frac{1-F_i(v_i)}{f_i(v_i)}$. The first term is the valuation, what we want to charge them at, and the second term is something we can think of as the price we have to pay for not knowing exactly what the valuation is. Compare and contrast the expression we got in (3.7) with the expected value received from the auction for v_i.

$$\int_0^{v_{max}} v_i(z)a_i(z,\vec{v})f_i(z)dz$$

For this reason, we call $\phi(v_i)$ the **virtual welfare**, and thus, we have found that the expectation of the payment of bidder i is equivalent to the expectation of the virtual welfare for that bidder. In notation, we have found that $\mathbb{E}_{v_i}[p_i(\vec{v})] = \mathbb{E}_[\phi(v_i) \cdot a_i(\vec{v})]$. Employing linearity of expectation, this tells us that our expected revenue is actually equal to the expected virtual welfare, i.e. $\mathbb{E}_{\vec{v}}[\sum_i p_i(\vec{v})] = \mathbb{E}_{\vec{v}}[\sum_i \phi_i(\vec{v})a_i(\vec{v})]$.

Our rule to maximize our expected revenue will then just be one which maximizes our expected virtual welfare. Thus, we should award the item to the person with the highest virtual welfare valuation, UNLESS all of them are negative (because, notably, this value can be negative).

We can ask if it is monotone, i.e. follows DSIC. It will be in the case that F is what we call **regular**. To tease out what "regular" means, let's think about what it should be. Monotonicity means that if we increase our valuation, we are more likely to be allocated the item. Based on our rule, this means that our virtual welfare should also increase. This means that the rule is monotone if $\phi(v) = v - \frac{1-F(v)}{f(v)}$ is increasing in v, i.e. the definition of regular. This makes it so that if we award to the highest virtual valuation, we are also awarding to the highest valuation. In short, if we have regular and independent F_is, we preserve monotonicity.