NETS 412: Algorithmic Game Theory April 6, 2015

Lecture 19

Lecturer: Aaron Roth *Scribe: Aaron Roth*

Profit Maximization in Online Auctions

In this lecture, we'll bring the class full circle. We'll consider a variant of the problem we considered in the last lecture – profit maximization in digital goods auctions – in which the bidders now arrive one at a time, and we need to decide whether or not to sell to each bidder as he arrives. To solve this problem, we'll revive and invoke our favorite algorithm from earlier in the class – the polynomial weights algorithm.

Recall our solution to revenue maximization from last lecture involves randomly partitioning bidders into two buckets, and then computing the optimal price in one bucket to charge to the other. This in particular means that in order to run this algorithm, it is necessary to have all bidders present before any sales are made, because all of them are necessary to compute the price we will charge. However, in practice, we often start selling things, and then customers arrive slowly over time. We can't wait until we have seen "all" of the bidders before selling things (if this even makes sense – we might have no definite end time in mind), and instead have to make decisions about whether to sell to individual customers immediately after they arrive, without knowing who will come next. We might be able to vary our price with time, however. This is the problem we will consider this lecture, which is called an *online auction*.

Our goal is to find a truthful online auction which approximates the optimal revenue. Recall that our benchmark is $\text{OPT}(v) = \max_{k \in [n]}(k \cdot v_{(k)})$, and that the random sampling auction achieves a 4 approximation[1]. In this class, in addition to working in the more difficult online setting, we will attempt to achieve a 1 approximation in the limit! That is, we want an algorithm that obtains revenue Rev such that under natural conditions, $\lim_{n \to \infty} \frac{Rev}{\text{OPT}} = 1$.

First, lets formally define our setting:

Definition 1 *In an online digital goods auction, we have n bidders with valuations $v_i \in [1, h]$.*

- *At time t, bidder t arrives and reports valuation v'_t.*
- *An item is allocated according to rule $x_t(v'_1, \ldots, v'_t)$, and payment $p_t(v'_1, \ldots, v'_t)$ is collected. Note that the allocation and payment rule is allowed to depend on* previous *bidders, but not* future *bidders.*

It will be helpful for us to think about a particularly simple kind of allocation and payment rule:

Definition 2 *In a* take-it-or-leave-it *(TIOLI) auction:*

- *At time t, a fixed price $s_t = s_t(v'_1, \ldots, v'_{t-1})$ is computed.*
- *The item is sold according to the following allocation and payment rules:*

$$x_t(v'_1, \ldots, v_{t-1}, v'_t) = 1 \Leftrightarrow v'_t \geq s_t \qquad p_t(v'_1, \ldots, v'_{t-1}) = s_t$$

In other words, the item is sold at a fixed price s_t to bidders with valuation above the price, and the price s_t is computed independently *of bidder t's own bid.*

The following is a simple observation:

Theorem 3 *Any take-it-or-leave-it auction is dominant strategy truthful.*

[1] Actually, a 4 approximation to the relaxed benchmark $\text{OPT}^{\geq 2}(v)$

Proof Since the price that bidder t faces is computed independently of his own bid, over/under-reporting does not influence the price – it can only result in agent t winning the item at a price he was not willing to pay, or failing to win the item even when he would have been willing to pay the price. ■

(Actually, its not hard to see that it is without loss of generality to consider TIOLI auctions... In single parameter domains, truthful auctions must be monotone. For deterministic auctions, this means that the allocation rule for each bidder must be determined by a fixed, bid-independent threshold (i.e. the fixed price)).

Our goal will be to *learn* the best fixed price (recall this is the benchmark we are competing against). The idea will be to use the polynomial weights algorithm, using prices as experts.

Lets recall the setting and guarantees of the polynomial weights algorithm: given a collection of N experts, each of whom experience losses $\ell_i^t \in [0,1]$ each day t, the polynomial weights algorithm with update parameter ϵ is able to select experts so as to achieve expected loss after T rounds:

$$L_{PW}^T \leq (1+\epsilon) \min_{k \in [N]} L_k^T + \frac{\ln(N)}{\epsilon}$$

(When we set $\epsilon = \sqrt{\frac{\ln(N)}{T}}$ and observed that $L_k^T \leq T$, we got $L_{PW}^T \leq \min_{k\in[N]} L_k^T + 2\sqrt{T\ln(N)}$) It will be more convenient for us to work in the setting in which the experts experience *gains* rather than losses. Here, the equivalent guarantee is:

$$G_{PW}^T \geq (1-\epsilon) \max_{k \in [N]} G_k^T - \frac{\ln(N)}{\epsilon}$$

Lets fix some collection of N prices $N \subseteq [1,h]$ and treat them as "experts". What should we interpret their gains as?

If we use price s_i on bidder i, we obtain revenue:

$$r_i = \begin{cases} s_i, & \text{if } v_i \geq s_i; \\ 0, & \text{if } v_i < s_i. \end{cases}$$

To run the polynomial weights algorithm, we need to normalize the gains to be in $[0,1]$, and so for price $p \in N$, we define its gain to be:

$$g_p^t = \begin{cases} p/h, & \text{if } v_i \geq p; \\ 0, & \text{if } v_i < s_i. \end{cases}$$

Note that because $p \leq h$, we have $g_p^t \in [0,1]$.

Let Rev_p^T denote the revenue of using fixed price p for the first T bidders:

$$Rev_p^T = p \cdot |\{i \leq T : v_i \geq p\}|$$

Recall that by our normalization, $G_p^T = \frac{Rev_p^T}{h}$. If we use the polynomial weights mechanism to select a price at every round, we get a Take-It-Or-Leave-It mechanism, which we know is dominant strategy truthful. Moreover, the polynomial weights guarantee tells us that:

$$G_{PW}^T \geq (1-\epsilon) \max_{k \in [N]} G_k^T - \frac{\ln(N)}{\epsilon}$$

which (multiplying through by h) is equivalent to:

$$Rev_{PW}^T \geq (1-\epsilon) \max_{p \in N} Rev_p^T - \frac{h\ln(N)}{\epsilon}.$$

So how should we choose our set of prices N? There is a tradeoff – choosing a larger set makes $\max_{p \in N} Rev_p^T$ guaranteed to be closer to $\text{OPT}(v)$, but on the other hand also makes $\ln(N)$ larger...

Consider setting prices as multiples of $(1+\alpha)$ for some $\alpha > 0$ – i.e. let:

$$N = \{1, (1+\alpha), (1+\alpha)^2, (1+\alpha)^3, \ldots, \lfloor h \rfloor_{1+\alpha}\} = \{(1+\alpha)^i : i \in [0, \log_{1+\alpha}(h)]\}$$

Then we have:

$$N \leq \frac{\ln(h)}{\ln(1+\alpha)} \leq \frac{(1+\alpha)\ln(h)}{\alpha}$$

(here we use the inequality $\ln(1+x) \geq \frac{x}{x+1}$ for $x \geq -1$).

We also know that:

$$\max_{p \in N} Rev_p^T \geq (1-\alpha) \max_{k \in N} k \cdot v_{(k)} = (1-\alpha)\text{OPT}(v)$$

(This is because whatever the optimal price $v_{(k)}$ is, we can always pick a smaller price p such that $p \geq (1-\alpha)v_{(k)}$. At least as many sales are made at this price, and we lose only a $(1-\alpha)$ factor of our revenue on each sale.)

Combining these guarantees, we can achieve:

$$\begin{aligned} Rev_{PW}^T &\geq (1-\epsilon)(1-\alpha)\text{OPT}(v) - \frac{h}{\epsilon}\left(\ln\ln(h) + \ln(\frac{1+\alpha}{\alpha})\right) \\ &\geq (1-\epsilon-\alpha)\text{OPT}(v) - \frac{h}{\epsilon}\left(\ln\ln(h) + \ln(\frac{2}{\alpha})\right) \end{aligned}$$

ϵ and α are parameters that we can choose. Choosing them to optimize the expression (using the magic of Mathematica!) we obtain:

$$Rev_{PW}^T \geq \text{OPT}(v) - 2\sqrt{h\text{OPT}\log\left(\frac{200\text{OPT}(v)\log(h)}{h}\right)}$$

How should we interpret this guarantee? It differs from OPT be some additive term, so it is incomparable to the guarantee of the random sampling auction. On the other hand, if we have:

$$\text{OPT} \geq \tilde{\Omega}\left(\frac{h}{\epsilon^2}\right)$$

(where the are ignoring log factors), then we have $Rev_{PW}^T \geq (1-\epsilon)\text{OPT}$, which is much better than a 4 approximation. Moreover, as n grows large, we expect the optimal revenue to grow, and so we expect this condition to hold in the limit for arbitrarily small ϵ.