CS 6840 Algorithmic Game Theory April 20, 2020

Lecture 29: Lerning in games

Instructor: Eva Tardos *Scribes: Zhen Zhang*

In the following lectures, we will discuss learning in games. Today's main focus is on repeated auctions:

- using smoothness framework
- extending PoA bound to agents who learn

The setup of the game is the following:

- auction repeats at time $t = 1, 2, ..., T$
- n players labeled as $1, ..., n$
- m items
- Auctions at time t is defined as allocating a subset of items A_i^t to player i, with payment p_i^t
- players' values are fixed at all time. i.e. $v_i(A)$ is independent of t
- utility of player i is defined as $u_i = \sum_{t=1}^{T}[v_i(A_i^t) - p_i^t] = \sum_{t=1}^{T} u_i^t$
- assume learning occurs for all players

The payment scheme can be first-price, second-price, all pay, or anything we've seen from the homework. The proof we will see requires smoothness, so it will work for any payment scheme where smoothness holds.

Multiplicative Weight Update Learning

Notation: Let $\mathbf{s}^t$ be the strategy vector at time t, s_i be any strategy of player i not necessarily from $\mathbf{s}^t$, $(s_i, \mathbf{s}_{-i}^t)$ be the strategy that player i plays s_i while everyone else plays according to $\mathbf{s}^t$

The following inequality was proved for multiplicative weight learning:

$$\sum_{t=1}^{T} c_i(\mathbf{s}^t) \leq \frac{1}{1-\epsilon} \min_{s_i^*} \sum_{t=1}^{T} c_i(s_i^*, \mathbf{s}_{-i}^t) + \frac{\log k}{\epsilon} \tag{1}$$

where k is the number of available strategies.

We use, without proof, the analogous inequality of utility version:

$$\sum_{t=1}^{T} u_i(\mathbf{b}^t) \geq (1-\epsilon) \max_{b_i^*} \sum_{t=1}^{T} u_i(b_i^*, \mathbf{b}_{-i}^t) - \frac{\log k}{\epsilon} \tag{2}$$

Assuming our auction is (λ, μ) smooth, we have the following theorem:

Theorem 1. For T sufficient large:

$$SW = \sum_{t=1}^{T}[Rev(\mathbf{b}^t) + \sum_i u_i(\mathbf{b}^t)] \approx\geq \frac{\lambda}{\max(1,\mu)} T \times OPT \tag{3}$$

where $Rev(\mathbf{b}^t)$ is the revenue with strategy $\mathbf{b}^t$, and OPT is the optimal social welfare in one iteration. Here $\approx\geq$ means the inequality only holds approximately.

Proof. First observe this inequality looks like the smoothness result we had before. Starting from the inequality (2), we need a b_i^* so that we can translate our smoothness result here.

Recall: from smoothness result previously, there exists a b_i^* for all i such that for all bid $\mathbf{b}$

$$\sum_i u_i(b_i^*, \mathbf{b_{-i}}) \geq \lambda OPT - \mu Rev(\mathbf{b}) \tag{4}$$

This is the b_i^* we are going to use for this proof.

Now we add up all players to get the following:

$$\sum_i \sum_{t=1}^{T} u_i(\mathbf{b}^t) \geq (1-\epsilon) \sum_{t=1}^{T} \sum_i u_i(b_i^*, \mathbf{b}_{-i}) - n\frac{\log k}{\epsilon} \tag{5}$$

Applying smoothness gives

$$\sum_i \sum_{t=1}^{T} u_i(\mathbf{b}^t) \geq (1-\epsilon) \sum_{t=1}^{T} (\lambda OPT - \mu Rev(\mathbf{b}^t)) - n\frac{\log k}{\epsilon} \tag{6}$$

Rearranging terms gives

$$\sum_i \sum_{t=1}^{T} u_i(\mathbf{b}^t) + (1-\epsilon)\mu \sum_{t=1}^{T} Rev(\mathbf{b}^t) \geq (1-\epsilon) \sum_{t=1}^{T} (\lambda OPT) - n\frac{\log k}{\epsilon} \tag{7}$$

Finally we have

$$\sum_{t=1}^{T} SW(\mathbf{b}^t) \geq \frac{(1-\epsilon)T(\lambda OPT)}{\max(1, \mu(1-\epsilon))} - n\frac{\log k}{\epsilon \max(1, \mu(1-\epsilon))} \tag{8}$$

Taking T large and ϵ properly so that the theorem approximately holds. Notice we never proved the strict inequality: if ϵ is large, then $(1-\epsilon)$ becomes small, and if ϵ is small, then the subtraction term is large. ■

Potential problems: Recall k is the number of strategies, and there are uncountably many of them if we define our bids on $\mathbb{R}^{\geq 0}$. One way of solving this problem is discretizing the values.

Suppose each item are unit valued and we discretize it with intervals of length δ, then there are $k = (\frac{1}{\delta})^m$ viable strategies. Notice that the k is inside a log function in the inequality, so the error may not be too bad.

However, throughout multiplicative weight update process, we need to maintain the probability of all strategies, which means there are at least $k = (\frac{1}{\delta})^m$ numbers for each t.