601.436/636 Algorithmic Game Theory **Lecturer:** Michael Dinitz
Topic: No-Regret and Coarse Correlated Equilibria **Date:** 2/13/20
Scribe: Michael Dinitz

6.1 Introduction

Today we'll talk about computing coarse correlated equilibria, and in particular a "natural" set of dynamics based on certain online learning algorithms known as *no-regret* algorithms. So we're first going to take a detour into online learning theory to define the guarantees of these algorithms, then show how they naturally lead to coarse correlated equilibria when used in games, and then show how to design these algorithms.

6.2 No-Regret in Online Learning

For the rest of this section, forget the game theory setting – we're just going to be doing online learning. In the online learning setting, there is one player (us/the computer) who has an action set A (these are sometimes called "arms", due to the analogy to a "multi-armed bandit" machine (i.e., a slot machine with multiple arms). Often we'll let $n = |A|$. At time $t = 1, 2, \ldots, T$:

- The algorithm picks a mixed strategy p^t (a probability distribution over A)
- The adversary picks a cost vector $c^t : A \to [0, 1]$ (it's crucial that these costs are bounded)
- An action a^t is drawn from p^t, and the algorithm has cost $c^t(a^t)$. The algorithm learns either the entire vector c^t (in the "experts" setting) or just learns $c^t(a^t)$ (in the "bandit") setting.

Informally, think of this process as follows. We're trying to "learn" which action is best, but we can't just observe – we have to actually play. This is "online" because we can't just learn from historical data at some point in the future, but instead need to actually learn as we go, i.e., "online". There are standard stories behind this: combining expert advice of stock market predictors / meteorologists, etc. But since this isn't a class about online learning, I'm not going to motivate the setting too much. Historically, ML people first thought about the case when there was no adversary, and instead each action (arm) had a fixed distribution that it drew from each time, and our goal was to find the arm with lowest expected cost. These are called "stochastic bandits". But it was later generalized to the setting I described, which is sometimes called "nonstochastic bandits".

Recall from 601.433 that the competitive ratio of an online algorithm is the worst case of the cost of the algorithm (which does not know the future) divided by the cost of the optimum (in retrospect, or knowing the future). This measure the "cost" of being online rather than having everything given to you up front.

Let's start with the bad news: there's no way we can actually be competitive with the best thing in retrospect.

Theorem 6.2.1 *No algorithm can be competitive with the best action sequence in hindsight.*

Proof: Let $|A| = 2$. The adversary will work as follows: if $p^t(0) \geq 1/2$ then the adversary sets $c^t(0) = 1$ and $c^t(1) = 0$, and otherwise $p^t(1) \geq 1/2$ and the adversary sets $c^t(1) = 1$ and $c^t(0) = 0$. Then there is some sequence that has total cost 0, but at every time the algorithm has probability at least 1/2 of getting cost 1. Thus the expected cost of the algorithm is at least $T/2$ while $OPT = 0$, for an unbounded competitive ratio. ∎

So we can't compete with the best sequence in hindsight. But let's change our benchmark back to what we were intuitively trying to do: learn the best action. If we're trying to learn the best single action, then we don't need to compete with $\sum_{t=1^T} \min_{a \in A} c^t(a)$ (the best sequence of actions). We just need to compete with $\min_{a \in A} \sum_{t=1}^T c^t(a)$ (the best action).

Definition 6.2.2 *The* regret *of an action sequence $a^1, a^2, \ldots a^T$ with respect to $a \in A$ is*

$$R_T(a) = \frac{1}{T}\left(\sum_{t=1}^{T} c^t(a^t) - \sum_{t=1}^{T} c^t(a)\right)$$

Note that unlike competitive ratio, this is an *additive* notion of approximation. There are both historical and technical reasons for this, but I'm not really going to get into it.

I want to formally define an adversary now, because we're actually going to use two different definitions.

Definition 6.2.3 *An (adaptive) adversary is a function which take as input 1) the algorithm A, 2) the time t, 3) mixed strategies $p^1, \ldots, p^t$ produced by A, and 4) realized actions $a^1, \ldots, a^{t-1}$ from the past, and outputs a cost vector $c^t : A \to [0, 1]$*

There is a more restricted notion of adversary that will be easier for us to analyze.

Definition 6.2.4 *An* oblivious *adversary (or* non-adaptive*) is an adversary that depends only on A and t. Equivalently, an oblivious adversary has to choose the sequence of cost vectors at the beginning of time, knowing only the algorithm A.*

Often the oblivious setting is easier than the adaptive setting, i.e., there are achievable bounds against oblivious adversaries that are not achievable against adaptive adversaries. But in this setting (online learning with nonstochastic bandits) it turns out that they're basically the same. To simplify things, and since this is an AGT class and not a learning class, I'm only going to talk about oblivious adversaries, but we're really going to care more about adaptive adversaries. You can read up on your own (in the book) about how to go from oblivious to adaptive.

Definition 6.2.5 *If A is an online learning algorithm, then its* expected regret *at time t with respect to a is*

$$\mathbf{E}[R_T^A(a)] = \frac{1}{T}\left(\sum_{i=1}^{T} \mathop{\mathbf{E}}_{a^t \sim p^t}[c^t(a^t)] - \sum_{t=1}^{T} c^t(a)\right)$$

Definition 6.2.6 *An algorithm A is a* no-regret algorithm *(or has no-regret) is for every adversary, for every $a \in A$, the expected regret with respect to a is $o(1)$ as $T \to \infty$. Slightly more formally, if $\lim_{T \to \infty} \mathbf{E}[R_T^A(a)] = 0$ or $\mathbf{E}[R_T^A(a)] = o(1)$, where the action sequence is the sequence of actions chosen by A.*

An amazing fact is that no-regret algorithms exist! We can design algorithms that do basically as well as the best action in hindsight, even though an adversary gets to see what we're doing and gets to choose cost vectors specifically to screw us up! I think that this is pretty amazing. And even more amazing is the fact that these algorithms are pretty simple, as we'll see later.

6.3 Connection to AGT

Before we talk about creating no-regret algorithms, let's connect this back to game theory. Why do we care about this crazy online learning setting from an AGT point of view?

6.3.1 Rationality and how to play a game

Suppose that you're a player in a game, and you expect to play this same game many times. One thing that you could do, which might be particularly appealing if you don't think you have a great understanding of the game or of the other players, is to give up on modeling the other players and instead think of the mixed strategies that the other players choose as an "adversary" and your pure strategies as your "actions" in the online learning setting! And then you can use a no-regret algorithm to "learn" which action of yours is best!

Slightly more formally, suppose that we are player i in a game that we play for times $t \in \{1, 2, \ldots, T\}$. At time t:

- Each player $j \neq i$ chooses some mixed strategy p_j^t over their pure strategies S_j (possibly using their own no-regret algorithm, but possibly not). For now, let's assume that these mixed strategies are *private* (unlike when we were talking about equilibria).
- Let σ_{-i} be the product distribution over S_{-i} defined by these mixed strategies.
- Let $c_i^t(a) = \mathbf{E}_{s_{-i} \sim \sigma_{-i}}[c_i(s_{-i}, a)]$ be the expected cost to player i of using action a (this cost vector is not be known to the algorithm since mixed strategies are private).
- Player i then uses a no-regret algorithm to choose a mixed strategy p_i^t and draws a pure strategy from it to play.

Why would player i act this way? By using a no-regret algorithm, player i guarantees that they do at least as well as the best action in hindsight. Since no-regret algorithms exist, why *wouldn't* players want to have this kind of guarantee? Maybe the players actually know more about the game and about the other players and so can do even better, but then they'll still have no-regret![1] In other words: it's easy to get no-regret, so any rational agent should have no-regret. It's a "minimum bar" for rationality. So if we can analyze what happens when players have no-regret, we'll have analyzed what happens with rational players! (Note: this is one interpretation. It is not universally accepted).

[1] This is not exactly true – why not?

6.3.2 Connection to Equilibria

Suppose that we have a cost-minimization game with k players and cost functions $C_i : S \to \mathbb{R}$ for each player $i \in [k]$. Let's start with a few definitions.

- Let p_i^t be the mixed strategy used by player i at time t.
- Let $\sigma^t = \prod_{i=1}^k p_i^t$ be the product distribution over S defined by the individual player distributions.
- Let $\sigma = \frac{1}{T}\sum_{t=1}^T \sigma^t$ be the "average" distribution. Note: σ *is not* a product distribution. Think of it as choosing a uniformly random value of t, and then using the product distribution σ^t. This is not a product distribution since the players are correlated through t.
- For each player i and time t, we define the cost vector $c_i^t(a) = \mathbf{E}_{s\sim\sigma^t}[C_i(s_{-i}, a)]$. Note that $\mathbf{E}_{a\sim p_i^t}[c_i^t(a)] = \mathbf{E}_{s\sim\sigma^t} C_i(s)$.

We can now finally give the main connection to AGT. Suppose that player i uses some algorithm A_i to generate its mixed strategy at each time.

Theorem 6.3.1 *Suppose that $\mathbf{E}[R_T^{A_i}(a)] \leq \epsilon$ for every $i \in [k]$ and $a \in S_i$. Then σ is an ϵ-approximate coarse correlated equilibrium:*

$$\mathop{\mathbf{E}}_{s\sim\sigma}[C_i(s)] \leq \mathop{\mathbf{E}}_{s\sim\sigma}[C_i(s_{-i}, s_i')] + \epsilon \qquad \textit{(for all } i \in [k] \textit{ and } s_i' \in S\textit{)}$$

Proof: We can show this just through some simple algebra using all of our definitions.

$$\begin{aligned}
\mathop{\mathbf{E}}_{s\sim\sigma}[C_i(s)] - \mathop{\mathbf{E}}_{s\sim\sigma}[C_i(s_{-i}, s_i')] &= \frac{1}{T}\sum_{t=1}^T \mathop{\mathbf{E}}_{s\sim\sigma^t}[C_i(s)] - \frac{1}{T}\sum_{t=1}^T \mathop{\mathbf{E}}_{s\sim\sigma^t}[C_i(s_{-i}, s_i')] \\
&= \frac{1}{T}\left(\sum_{t=1}^T \mathop{\mathbf{E}}_{a^t\sim p_i^t}[c_i^t(a^t)] - \sum_{t=1}^T c_i^t(s_i')\right) \\
&= \mathbf{E}[R_T^{A_i}(s_i')] \leq \epsilon.
\end{aligned}$$

■

So if all players use no-regret algorithms, the average distribution converges to a coarse correlated equilibrium! Note that it is important here that the algorithm work against *adaptive* adversaries, since of course the other players will adapt to what we do.