TTIC 31260 - Algorithmic Game Theory (Spring 2024)

Lecturer: Avrim Blum Lecture 4: March 27, 2024

Efficient Algorithms for Large Action Spaces

1 Overview

One problem with randomized weighted majority is that it requires keeping explicit track of weights across all experts (or actions). This can be a problem when experts/actions are implicit, like paths in a graph, or different ways of ranking or allocating some set of items. In particular, even though the regret is logarithmic in the number of choices, the running time is (at least) linear. Today we will look at one classic way to get around this problem, and getting a good regret bound efficiently when there is sufficient structure to the action space. If you wish to do a project on this topic, you could look at some more recent results along these lines, such as [BHK$^+$23, DHL$^+$20].

1.1 The general setting - and some structure

We assume we have a set $\mathcal{A}$ of actions, and the world has some class $\mathcal{C}$ of cost functions from $\mathcal{A}$ to $\mathbb{R}_+$. At each time step $t = 1, 2, 3, \ldots, T$, our algorithm chooses some $a_t \in \mathcal{A}$, the world chooses some cost function $c_t \in \mathcal{C}$, and we pay $c_t(a_t)$.

Now, let's add some structure. For today we will assume *linear optimization.* Specifically, we assume $\mathcal{A} \subseteq \mathbb{R}^m_+$ and $\mathcal{C} \subseteq \mathbb{R}^m_+$, with the interpretation that $c(a) = \langle c, a \rangle$. Additionally, we will assume we can efficiently solve the offline optimization problem: given some $c \in \mathbb{R}^m_+$, find $\arg\min_{a \in \mathcal{A}} \langle c, a \rangle$.

1.2 Some concrete examples

Online ranking: Imagine a search engine that given a query q outputs a list a_1 of web pages. The user clicks on one, and we define the loss of a_1 as the depth in the list of the web-page that was clicked. The next time q is queried we give a different ordering a_2 of these web pages, and the user this time clicks on (perhaps) a different page. Our goal is to do nearly as well as best fixed ordering of these pages in hindsight. Suppose there are m webpages in our list, and let's say the cost for the user clicking the top web page is 1, for clicking the second web page on the list is 2, and so on, down to m for the last one (or really it could be any sequence of costs, increasing at whatever rate matches our belief about user annoyance). We could model this as an "experts" problem with $m!$ experts but that would be too much computation. Instead, let A be the set of all $m!$ points in $\mathbb{R}^m$ with coordinates $\{1, 2, \ldots, m\}$ in some order, with the interpretation that each coordinate is one of the web-pages and the value of that coordinate is the depth of that page in the list (or more generally, the cost associated with that depth). If the user at time t clicks on web-page i, define $c_t = e_i$ to be the unit vector in coordinate i. Then $\langle c_t, a_t \rangle$ is the cost of the ordering a_t.

Notice that solving the offline optimization problem (given $c \in \mathbb{R}_+^m$, find $\arg\min_{a \in \mathcal{A}} c(a)$) is easy. Just output the ranking the puts the coordinate of highest loss first, then the coordinate of second-highest loss second, etc.

Online shortest paths: Consider the following adaptive route-choosing problem. Imagine each day you have to drive between two points u and v. You have a map (a graph G) but you don't know what traffic will be like (what the cost of each edge will be that day). Say the costs are only revealed to you after you get to v. We can model this by having one dimension per edge, and each path is represented by the indicator vector listing the edges in the path. Then the cost vector c_t is just the vector with the costs of each edge that day. With this representation, $\langle c_t, a_t \rangle$ is the cost of path a_t. Notice that you *could* represent this as an experts problem also, with one expert per path, but the number of s-t paths can be exponential in the number of edges in the graph (e.g., think of the case of a grid). However, given any set of edge lengths, we can efficiently compute the best path for that cost vector, since that is just a shortest-path problem. You don't have to explicitly list all possible paths.

2 Follow the Perturbed Leader (FTPL)

Today, we will analyze an algorithm for this problem from [KV05] called "Follow the Perturbed Leader" (FTPL).

First, one natural algorithm to try is "Follow the Leader" (FTL), which is to choose the action today that performed best on average in the past. That is, FTL is the algorithm that chooses

$$a_t \quad = \quad \arg\min_{a \in \mathcal{A}} \langle c_1 + ... + c_{t-1}, a \rangle,$$

with ties broken in some deterministic manner. This is like an on-line version of Empirical Risk Minimization. Unfortunately, this is not a great idea in a game-theoretic setting. In particular, it's deterministic, so an adversary could beat you every time even in a simple game like matching-pennies or rock-paper-scissors. For instance, in the case of online shortest paths, imagine there are two routes you can take, one fast and one slow, but they alternate which is fast and which is slow from day to day; this procedure will always choose the wrong one.

To fix this problem, FTPL "hallucinates" a fake day 0 with a random cost vector c_0, chosen from an appropriate distribution. Then it picks the action

$$a_t \quad = \quad \arg\min_{a \in \mathcal{A}} \langle c_0 + c_1 + ... + c_{t-1}, a \rangle.$$

What we will show is that if c_0 is picked from an appropriate probability distribution, then for any sequence of cost vectors, the regret of this algorithm will be low in expectation. For this analysis, we will assume that the series of cost vectors $c_1, c_2, \ldots$ has been determined in advance by the world (but the future is just unknown to us), so that new cost vectors do not depend on our algorithm's previous actions. This is often called an *oblivious adversary* model. One can also prove bounds for an *adaptive adversary* model in which new cost vectors *can* depend on the algorithm's previous actions, but then one needs to re-randomize at each time step.

Specifically, we will prove the following theorem about the FTPL algorithm.

Theorem 1 *Assume the maximum L_1 length of any cost vector $c \in \mathcal{C}$ is 1, and the L_1 diameter of the action set $\mathcal{A}$ is D. If $c_0 \sim \mathrm{Unif}([0, 2/\epsilon]^m)$ then the expected regret of FTPL in T steps satisfies:*

$$\mathbf{E}[Regret] \quad \leq \quad D(\epsilon T/2 + 1/\epsilon).$$

Setting $\epsilon = \sqrt{2/T}$ gives an expected regret at most $D\sqrt{2T}$.

2.1 Proof Intuition

The first step of the proof is to show that the non-implementable "Be the Leader" (BTL) algorithm of picking

$$a_t \quad = \quad \arg\min_{a \in \mathcal{A}} \langle c_1 + ... + c_t, a\rangle$$

has zero (or negative) regret. I.e., you find the best action in hindsight tomorrow and use that today. This is maybe not surprising but also not obvious. Note this is *not* the same as the action a that minimizes $\langle c_t, a\rangle$, which obviously would be super-optimal.

Now, FTL and BTL both go to a bar and start drinking. As they drink, their objective functions get noisier and noiser, adding in their randomized c_0's, creating FTPL and BTPL. The second step of the proof is to show that at an appropriate noise level, FTPL is very similar to BTPL (this is much like "randomized smoothing"), and BTPL is not too much worse than BTL.

2.2 A Simple Preliminary Fact

Fact 1 *For any $c \in \mathcal{C}$ and any $a, a' \in \mathcal{A}$ we have $\langle c, a\rangle - \langle c, a'\rangle \leq D$.*

This follows from the fact that the L_∞ length of c is at most 1 (so the requirement that the L_1 length of c is at most 1 is overkill for this property, though we will use that later) and the fact that the L_1 diameter of $\mathcal{A}$ is D.

2.3 Proof Part 1: Be the Leader

We begin by proving that Be the Leader has zero or negative regret.

Theorem 2 *Let $a_t = \arg\min_{a \in \mathcal{A}} \langle c_1 + ... + c_t, a\rangle$. Then*

$$\langle c_1, a_1\rangle + \langle c_2, a_2\rangle + ... + \langle c_T, a_T\rangle \quad \leq \quad \langle c_1, a_T\rangle + \langle c_2, a_T\rangle + ... + \langle c_T, a_T\rangle.$$

Proof: We prove this by induction on T.

Base Case: For $T = 1$, the LHS and RHS are equal.

General Case: By induction, we can assume:

$$\langle c_1, a_1\rangle + ... + \langle c_{T-1}, a_{T-1}\rangle \quad \leq \quad \langle c_1 + ... + c_{T-1}, a_{T-1}\rangle.$$

We also know, by definition of a_{T-1}, that:

$$\langle c_1 + ... + c_{T-1}, a_{T-1}\rangle \quad \leq \quad \langle c_1 + ... + c_{T-1}, a_T\rangle.$$

Putting these together, and adding $\langle c_T, a_T\rangle$ to both sides, yields the theorem. ■

2.4 Proof Part 2: Analyzing the Randomization

We now use Theorem 2 to prove Theorem 1.

Proof of Theorem 1: We begin by showing that the expected cost paid by FPTL at some time t is close to the expected cost paid by BPTL at time t.

Specifically, FTPL is choosing its objective function $c_0 + c_1 + ... + c_{t-1}$ uniformly at random from a box of side-length $2/\epsilon$ with corner at $c_1 + ... + c_{t-1}$. BTPL is choosing its objective function uniformly at random from a box of side-length $2/\epsilon$ with corner at $c_1 + ... + c_t$. For convenience of notation, let's call the boxes Box_t^{FTPL} and Box_t^{BTPL}.

These boxes each have volume $V = (2/\epsilon)^m$. Since their bottom corners differ by a vector c_t of L_1 length at most 1, the intersection of the boxes has volume at least $(2/\epsilon)^m - (2/\epsilon)^{m-1} \geq V(1-\epsilon/2)$. Note that this is where the L_1 length of the cost vectors comes in.

The above analysis implies that for some $p \leq \epsilon/2$ we can view FTPL as with probability $1-p$ choosing its objective function uniformly at random from $I = Box_t^{FTPL} \cap Box_t^{BTPL}$, and with probability p choosing its objective function uniformly at random from $F = Box_t^{FTPL} \setminus Box_t^{BTPL}$. Similarly, we can view BTPL as with probability $1-p$ choosing its objective function uniformly at random from I, and with probability p choosing its objective function uniformly at random from $B = Box_t^{BTPL} \setminus Box_t^{FTPL}$. Thus, if $\alpha_I = \mathbf{E}_{c\sim I}[\langle c_t, a_t\rangle \mid a_t = \arg\min_{a\in\mathcal{A}}\langle c, a\rangle]$, $\alpha_F = \mathbf{E}_{c\sim F}[\langle c_t, a_t\rangle \mid a_t = \arg\min_{a\in\mathcal{A}}\langle c, a\rangle]$, and $\alpha_B = \mathbf{E}_{c\sim B}[\langle c_t, a_t\rangle \mid a_t = \arg\min_{a\in\mathcal{A}}\langle c, a\rangle]$, then the expected cost of FTPL at time t is $(1-p)\alpha_I + p\alpha_F$ and the expected cost of BTPL at time t is $(1-p)\alpha_I + p\alpha_B$. By Fact 1, we know that $|\alpha_F - \alpha_B| \leq D$, so the difference in expected cost between the two algorithms at time t is at most $pD \leq \epsilon D/2$. Summing over all times t we get a difference at most $\epsilon DT/2$.

We complete the proof by showing that the expected regret of BTPL is at most D/ϵ. To do this, define

$$a_t^{BTPL} \quad = \quad \arg\min_{a\in\mathcal{A}}\langle c_0 + ... + c_t, a\rangle,$$

and define

$$a_t^{BTL} \quad = \quad \arg\min_{a\in\mathcal{A}}\langle c_1 + ... + c_t, a\rangle.$$

Note that a_T^{BTL} is the optimal action in hindsight. For any vector c_0, by Theorem 2 (starting from index 0 instead of index 1) we have

$$\begin{aligned}\langle c_0, a_0^{BTPL}\rangle + ... + \langle c_T, a_T^{BTPL}\rangle &\leq \langle (c_0, +... + c_T), a_T^{BTPL}\rangle \\ &\leq \langle (c_0, +... + c_T), a_T^{BTL}\rangle.\end{aligned}$$

So,

$$\langle c_1, a_1^{BTPL}\rangle + ... + \langle c_T, a_T^{BTPL}\rangle \leq \langle (c_1, +... + c_T), a_T^{BTL}\rangle + \langle c_0, a_T^{BTL} - a_0^{BTPL}\rangle. \qquad (1)$$

The left-hand-side of Equation (1) is the cost of BTPL. The first term on the right-hand-side of Equation (1) is the cost of the optimal action in hindsight. So, the expected regret of BTPL is at most $\mathbf{E}[\langle c_0, a_T^{BTL} - a_0^{BTPL}\rangle]$. Since $\mathcal{A}$ has L_1 diameter at most D, and each coordinate of c_0 has expected value $1/\epsilon$, we get $\mathbf{E}[\langle c_0, a_T^{BTL} - a_0^{BTPL}\rangle] \leq D/\epsilon$, completing the proof. ■

More information: For more information on this and related methods, see [KV05, SS12, Zin03].

References

[BHK+23] Daniel Beaglehole, Max Hopkins, Daniel Kane, Sihan Liu, and Shachar Lovett. Sampling equilibria: Fast no-regret learning in structured games. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3817–3855. SIAM, 2023.

[DHL+20] Miroslav Dudík, Nika Haghtalab, Haipeng Luo, Robert E Schapire, Vasilis Syrgkanis, and Jennifer Wortman Vaughan. Oracle-efficient online learning and auction design. *Journal of the ACM (JACM)*, 67(5):1–57, 2020.

[KV05] Adam Kalai and Santosh Vempala. Efficient algorithms for online decision problems. *Journal of Computer and System Sciences*, 71(3):291 – 307, 2005.

[SS12] Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2), 2012. http://www.cs.huji.ac.il/~shais/papers/OLsurvey.pdf.

[Zin03] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *ICML*, pages 928–936, 2003.