CSE599s, Spring 2014, Online Learning Lecture 17 - 05/27/2014

Combinatorial Bandits

Lecturer: Ofer Dekel *Scribe: Stephen Joe Jonany*

1 Combinatorial Bandit Game

We first define the combinatorial bandit problem with linear losses. We have a ground set $[k] = \{1, ..., k\}$ and a combinatorial action set $\mathcal{A} \subseteq \{0,1\}^k$. The adversary chooses loss functions $f_1, ..., f_T$, where $f_t : [k] \to [0,1]$. The game then proceeds like so:

Combinatorial Bandit Game

```
for $t = 1, 2, \ldots, T$
    player (randomly) chooses action $A_t$ from the action set $\mathcal{A}$
    player incurs and observes loss $f_t(A_t) = \sum_{i \in A_t} f_t(i)$
```

There are several variations on how informative the losses observed by the players are:

1. Bandit feedback: Player observes only the single number $f_t(A_t)$. This is the case that we are going to discuss.
2. Semi-bandit feedback: Player observes $(f_t(i))_{i \in A_t}$. This loss function is more informative, and one would think that with more information to leverage from the observed losses, one would be able to do better than if only bandit feedback is observed.

2 Online Shortest Paths Problem

2.1 Definition

We are going to look at a specific example of the combinatorial bandit game, the online shortest paths problem. The problem is defined like so. Given $G = (V, E)$, a directed graph, a source vertex $s \in V$ and a target vertex $t \in V$, the adversary defines loss functions $f_1, ..., f_T$, where $f_t : E \to [0,1]$. The player's action set, $\mathcal{A}$, is defined to be the set of all edge indicator vectors in $\{0,1\}^{|E|}$, such that the edges with indicator $= 1$ compose a simple (cycle-free) path.

Online Shortest Paths

```
for $t = 1, 2, \ldots, T$
    player (randomly) chooses an action/path $A_t$ from $s$ to $t$
    player incurs and observes the loss $f_t(A_t) = \sum_{e \in A_t} f_t(e)$
```

In English, at each time step, the adversary puts a loss value on each edge, and the player gets to pick a simple path, and the loss he incurred is equal to the total loss values of the edges on his chosen path.

Our goal is to minimize the regret,

$$\mathbb{E}[\sum_{t=1}^{T} f_t(A_t)] - \min_{a \in \mathcal{A}} \sum_{t=1}^{T} f_t(a),$$

where A_t and a are actions from the action set $\mathcal{A}$, and $f_t(a)$, then cost of an action is shorthand for $\sum_{i \in a} f(i)$.

2.2 First Attempt

A naive solution would be to solve this as a multi-armed bandit problem, which we have previously proven the regret bound for. We perform the transformation by treating each action $a \in \mathcal{A}$ as an arm, and dividing the losses by $\frac{1}{k}$. This is so that at any time step, the loss observed is $\frac{1}{k} f_t(a) \in [0, 1]$. (Note that $f_t(a) \le k$, since k is the number of edges, and each edge cost is at most 1.)

There are two problems with this. First, the regret bound is $O(\sqrt{T|\mathcal{A}|log|\mathcal{A}|})$, which is not that good since $|\mathcal{A}|$, the number of simple paths in the graph, could be exponential with k. Secondly, the computational complexity of each round is also $O(|\mathcal{A}|)$.

2.3 A Better Solution

To arrive at a better solution, we make the observation that the problem has structure that we did not make use of in the naive solution. The adversary assigns loss values to each edge weight, and the player's loss is a linear function of these values. So, the fact that I observed the loss value for the path I chose actually gives me some information about the other actions/paths I could have possibly taken. Furthermore, the ground set, k is small. Maybe there is a way to not depend on $|\mathcal{A}|$, which is superpolynomial in k. We will exploit this problem structure in 3 steps.

Step 1. Formulate the problem as an efficient online integer-linear optimization.
We redefine the problem that so that the action set $\mathcal{A}$ is defined by a polynomial set of constraints, each constraint in $O(|E|)$. Let $in(v)$ denote the incoming edges of the vertex v, and $out(v)$ the outgoing edges. We define $\mathcal{A} \subseteq \{0, 1\}^k$, the set of edge indicator vectors that satisfy the following properties:

1. Unit flow originates at $s : \forall a \in \mathcal{A},\ \sum_{e \in out(s)} a_e = 1$.
2. Unit flow absorbed at $t : \forall a \in \mathcal{A},\ \sum_{e \in in(t)} a_e = 1$.
3. Flow conservation: $\forall a \in \mathcal{A}\ \forall v \in V \setminus \{s, t\}\ \sum_{e \in in(v)} a_e = \sum_{e \in out(v)} a_e$. That is, aside from the source and the sink, for every node, the amount of incoming flow is equal to the amount of outgoing flow.
4. $\forall v \sum_{e \in out(v)} a_e \le 1$. This constraint is needed to ensure that the paths are non cyclic.

The lemma is that the above definition enforces $\mathcal{A}$ to be equivalent to a set of simple paths that originate from s and ends at t. We omit the proof, but it can be found in lectures in algorithm classes related to max-flow algorithms.

Step 2. Relax the combinatorial constraint and solve as a bandit linear optimization problem
Although originally we mentioned that a player can only pick a single path at every time step, we consider the case when the player can choose actions from $\bar{\mathcal{A}}$, where $\bar{\mathcal{A}}$ is the convex hull of $\mathcal{A}$. Or in other words, instead of picking a path with a unit flow, the player gets to pick multiple fractional flows.
If such a move is allowed, then we have a convex set of actions, which means we can use bandit linear optimization algorithms. As described in the previous lecture, when the player chooses actions $W_1, ..., W_T \in \bar{\mathcal{A}}$, the regret is bounded by

$$\mathbb{E}[\sum_{t=1}^{T} W_t \cdot l_t]] - \min_{w \in \bar{\mathcal{A}}} \sum_{t=1}^{T} w \cdot l_t = O(\sqrt{T})$$

We note that the comparator term is equivalent to $\min_{w \in \mathcal{A}} \sum_{t=1}^{T} w \cdot l_t$, that is, if we were to use a pure strategy. We have proven this before. However, we still have to replace the fractional flows W_t with a single discrete path A_t to satisfy the problem definition. The next step helps us satisfy this, while allowing us to keep the same regret bound.

Step 3. Add randomization to recover discrete actions.
We effectively get rid of the fractional flow by treating these fractional weights as probability distributions over unit flows, and randomly choose a path based on this distribution. That is, on each round,

1. Use bandit convex optimization to find a flow W_t. Note that this flow is fractional, and we can't actually use it as a legitimate move.
2. Find paths $a_{t,1}, ..., a_{t,k+1}$ and convex coefficients $\alpha_{t,1}, ..., \alpha_{t,k+1}$ such that $W_t = \sum_{i=1}^{k+1} \alpha_{t,i} a_{t,i}$. This is always possible due to Caratheodory's Theorem (proven below since it's more of a side detail). Note that the runtime complexity of this step is polynomial with respect to k.
3. Independently choose a random path A_t, where $A_t = a_{t,i}$ with probability $\alpha_{t,i}$.

The expected loss, which is $\mathbb{E}[A_t \cdot l_t | W_t]$, using the definition of expectation, is equal to $\sum_{i=1}^{k+1} \alpha_{t,i}(a_{t,i} \cdot l_t)$. Taking the l_t out of the summation, and using the definition of W_t, this is equivalent to $W_t \cdot l_t$. This is exactly the expression we used in the regret bound for bandit linear optimization algorithm!
As a recap, the steps we have showed have a runtime complexity which is polynomial in k, and by adding randomization, we have produced a move that satisfies the problem constraint (non-fractional flow), and whose loss in expectation is equal to $W_t \cdot l_t$, which allows us to benefit from the same regret guarantee of the bandit linear optimization algorithm.

Caratheodory's Theorem

Lemma 1. *Let S be a set of vectors in $\mathbb{R}^k$. For any $z \in$ convex(S) there exists vectors $s_1, ..., s_{k+1} \in S$ such that $z \in convex(s_1, ..., s_{k+1})$.*

Proof. By definition, $z \in$ convex(S) means that $z = \sum_{i=1}^{n} \lambda_i v_i$, where $v_i \in S, \lambda_i > 0$ and $\sum_{i=1}^{n} \lambda_i = 1$.
The easy case is when $n \leq k+1$, we are immediately done.
Otherwise, $n \geq k+2$. Consider the vectors $(v_2 - v_1), ..., (v_n - v_1)$. Since there are at least $k+1$ such vectors, and we are in the k dimension, these vectors are linearly dependent. So, there exists coefficients μ_i, such that $\sum_{i=2}^{n} \mu_i (v_i - v_1) = 0$ and $\exists i \ \mu_i \neq 0$. Define $\mu_1 = -\sum_{i=2}^{n} \mu_i$, which means that
$\sum_{i=1}^{n} \mu_i = \mu_1 + \sum_{i=2}^{n} \mu_i = 0$.
So, $\sum_{i=1}^{n} \mu_i v_i = \sum_{i=1}^{n} \mu_i v_i - 0 \cdot v_1 = \sum_{i=1}^{n} \mu_i v_i - \sum_{i=1}^{n} \mu_i \cdot v_1 = \sum_{i=1}^{n} \mu_i (v_i - v_1) = \sum_{i=2}^{n} \mu_i (v_i - v_1) = 0$.
Therefore,

$$z = \sum_{i=1}^{n} \lambda_i v_i - 0 = \sum_{i=1}^{n} \lambda_i v_i - \alpha \sum_{i=1}^{n} \mu_i v_i = \sum_{i=1}^{n} (\lambda_i - \alpha \mu_i) v_i.$$

We choose $\alpha = \min\{\frac{\lambda_j}{\mu_j} : \mu_j > 0\}$. This is always possible since at least one $\mu_i \neq 0$. When we choose such an α, one of the $\lambda_i - \alpha \mu_i$ coefficients would evaluate to 0, and now we have a convex combination of $n-1$ points from S. We can keep doing this until $n = k+1$. □