Online Learning in Games

DRAFT

Prof. Volkan Cevher
volkan.cevher@epfl.ch

Lecture 8: Computational Efficient Online Learning

Laboratory for Information and Inference Systems (LIONS)
École Polytechnique Fédérale de Lausanne (EPFL)

EE-735 (Spring 2024)

License Information for Online Learning in Games Slides

Acknowledgements

These slides were originally prepared by Lukas Vogl and Weronika Wrzos-Kaminska.

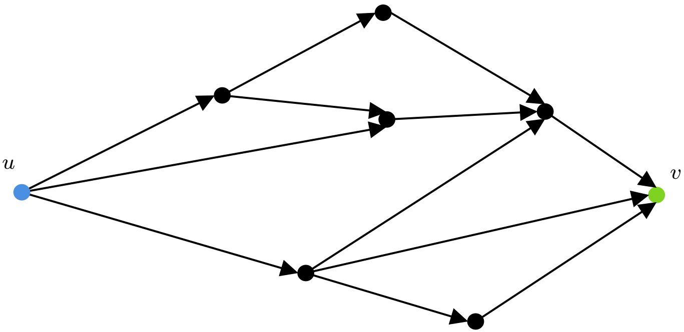
Online shortest paths

Online shortest path

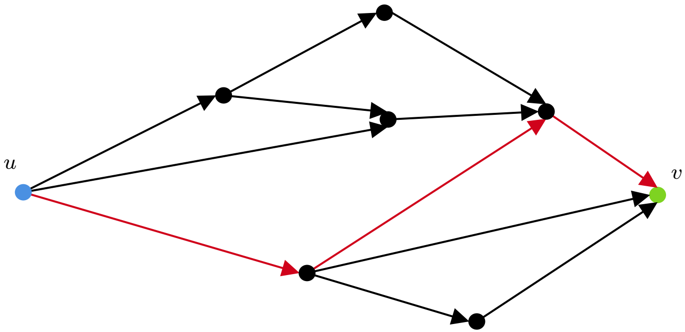
We are given a directed graph $G = (V, E)$ and a fixed pair of vertices u, v. At each round $t = 1, \dots, T$,

- The *learner* selects a path $p^t \in \{0,1\}^E$ from u to v.
- An *adversary* selects lengths for each edge $c_t \in [0,1]^E$ depending on $p^1, \dots, p^{t-1}$.
- The *learner* incurs a cost of $\langle c_t, p^t \rangle$ and receives c_t as feedback.

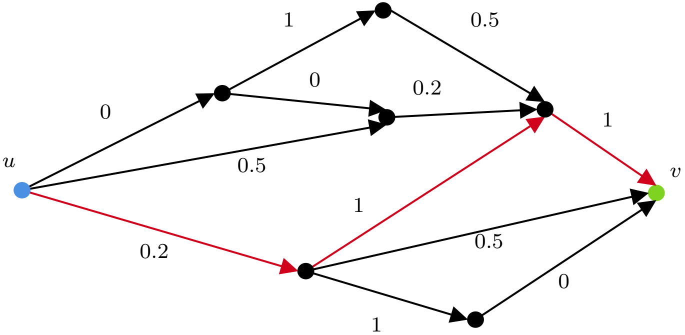
Online shortest path



Online shortest path



Online shortest path



Regret for online shortest paths

Regret (Online shortest path)

Let $\mathcal{A}$ be a (randomized) algorithm and $c_1, \ldots, c_T \in [0,1]^E$ a sequence of edge lengths. Then, the regret of $\mathcal{A}$ is defined as

$$\mathcal{R}_{\mathcal{A}}(T) \coloneqq \sum_{t=1}^{T} \langle c_t, p^t \rangle - \min_{u\text{-}v \text{ path } p} \sum_{t=1}^{T} \langle c_t, p \rangle$$

where p^t is the path that algorithm $\mathcal{A}$ selects in round $t = 1, \ldots, T$.

Remark:

- We are interested in bounding the expected regret $\mathbb{E}[\mathcal{R}_{\mathcal{A}}(T)]$.

Hedge algorithm for online shortest paths

One can solve online shortest path with the Hedge Algorithm by introducing an expert for each possible path $\mathcal{S} = \{p_1, ..., p_k\}$.

Hedge algorithm for online shortest paths

- Let $\mathcal{S} = \{p_1, ..., p_k\}$ be the set of all paths from u to v and set $\gamma = \sqrt{\log k/T}$.
- Let $w_p^1 = 1$ for all paths $p \in \mathcal{S}$.
- For each round $t = 1, ..., T$:
  - The *learner* selects $x^t \in \Delta(\mathcal{S})$ as follows,
$$x_p^t = \frac{w_p^t}{\sum_{p' \in \mathcal{S}} w_{p'}^t} \quad \text{for each path } p \in \mathcal{S}.$$
  - The *adversary* selects a cost $c_p^t \in [0, n]$ for each path $p \in \mathcal{S}$.
  - The *learner* suffers expected cost $\langle c^t, x^t \rangle$ and receives c^t as feedback.
  - Update the weights as follows,
$$w_p^{t+1} = w_p^t e^{-\gamma c_p^t} \quad \text{for each path } p \in \mathcal{S}.$$

.

Goal for online shortest path

- One can solve online shortest path with the Hedge Algorithm by introducing an expert for each possible path $\mathcal{S} = \{p_1, ..., p_k\}$.
- Number of paths is exponential, $k = 2^{\Omega(n)}$.
- $\mathcal{R}_{\textit{Hedge}}(T) = O\left(\sqrt{T \log k}\right) = O\left(\sqrt{T \cdot n \log n}\right)$.
- Regret is polynomial but **runtime** is exponential!

Goal (Online shortest path)

We want to find a (randomized) algorithm $\mathcal{A}$ that runs both in polynomial time and has polynomial regret in expectation, $\mathbb{E}[\mathcal{R}_{\mathcal{A}}(T)] = O\left(poly(n)\sqrt{T}\right)$.

Generalized setting: Online decision problems

Linear optimization oracle

A linear optimization oracle $M : \mathbb{R}^d \to \mathcal{S}$ optimizes a linear function over a (possible infinite) strategy space $\mathcal{S} \subseteq \mathbb{R}^d$ in polynomial time, i.e. M computes the best strategy given a cost vector c, $M(c) = \text{argmin}_{s \in \mathcal{S}} \langle c, s \rangle$.

Online decision problem

We are given a (possible infinite) strategy space $\mathcal{S} \subseteq \mathbb{R}^d$ and a linear optimization oracle $M : \mathbb{R}^d \to \mathcal{S}$. At each round $t = 1, \ldots, T$,

- The *learner* has to select a strategy $s_t \in \mathcal{S}$.
- An *adversary* selects a cost vector $c_t \in \mathcal{C} \subseteq \mathbb{R}^d$ depending on $s_1, \ldots, s_{t-1}$.
- The *learner* incurs a cost of $\langle c_t, s_t \rangle$ and observes the cost vector c_t.

Remark:

- We can model online shortest path by choosing $\mathcal{S} = \{s \in \{0,1\}^E \mid s \text{ is the characteristic vector of a path from } u \text{ to } v\}$.
- The linear optimization oracle M can be implemented by the Bellman-Ford algorithm.

Further examples: MST

Example (Minimum spanning tree)

- Given a graph $G = (V, E)$ want to find a spanning tree minimizing the weight.
- Strategy space: $\mathcal{S} \subseteq \mathbb{R}^E$ is the discrete set $\mathcal{S} \subseteq \{0,1\}^E$ representing all spanning trees of G (each spanning tree can be represented as a vector in $\{0,1\}^E$).
- The linear optimization oracle M can be implemented via Prim's or Kruskal's algorithm.
- At each time step $t = 1, 2, ...T$,
  - Select a spanning tree $s_t \in \mathcal{S}$.
  - Adversary selects a cost for each edge, which yields a cost vector $c_t \in \mathbb{R}^E$.
  - At time t, the cost is $\langle c_t, s_t \rangle$ and the feedback is c_t

Further example: Bipartite matching

Example (Maximum bipartite matching)

- Given a bipartite graph $G = (V, E)$ want to find a matching $M \subseteq E$ maximizing the weight of the matching $\sum_{e \in M} w_e$.
- Strategy space: $\mathcal{S} \subseteq \mathbb{R}^E$ is the discrete set $\mathcal{S} \subseteq \{0,1\}^E$ representing matchings of G (each matching can be represented as a vector in $\{0,1\}^E$).
- The linear optimization oracle M can be implemented via linear programming.
- At each time step $t = 1, 2, ...T$,
  - Select a matching $s_t \in \mathcal{S}$.
  - Adversary selects a weight w_e for each edge. The cost vector $c_t \in \mathbb{R}^E$ can be represented as $c_t(e) = -w_e$.
  - At time t, the cost is $\langle c_t, s_t \rangle$ and the feedback is c_t

Generalized setting: Online decision problems

Online decision problem

We are given a (possible infinite) strategy space $\mathcal{S} \subseteq \mathbb{R}^d$ and a linear optimization oracle $M : \mathbb{R}^d \to \mathcal{C}$. At each round $t = 1, \ldots, T$,

- The *learner* has to select a strategy $s_t \in \mathcal{S}$.
- An *adversary* selects a cost vector $c_t \in \mathcal{C} \subseteq \mathbb{R}^d$ depending on $s_1, \ldots, s_{t-1}$.
- The *learner* incurs a cost of $\langle c_t, s_t \rangle$ and observes the cost vector c_t.

Regret

Let $\mathcal{A}$ be a (randomized) algorithm and let $c_1, \ldots, c_T$ be a sequence of cost vectors. Then, the regret of $\mathcal{A}$ is defined as

$$\mathcal{R}_{\mathcal{A}}(T) := \sum_{t=1}^{T} \langle c_t, s_t \rangle - \min_{s \in \mathcal{S}} \sum_{t=1}^{T} \langle c_t, s \rangle$$

where s_t is the output of algorithm $\mathcal{A}$ in round $t = 1, \ldots, T$.

Follow the leader: A simple approach that does not work

Follow the leader algorithm

For each round $t = 1, \ldots, T$,

- Let $c_{1:t-1} = \sum_{k=1}^{t-1} c_k$.
- The *learner* selects the strategy $M(c_{1:t-1}) = \text{argmin}_{s \in \mathcal{S}} \langle c_{1:t-1}, s \rangle$, i.e. the strategy with the minimum aggregated cost so far.
- The *learner* suffers cost $\langle c_t, M(c_{1:t-1}) \rangle$ and receives feedback c_t.

Example

Consider $\mathcal{S} = \{(1,0),(0,1)\}, c_1 = (0.5, 0)$ and

$$c_t = \begin{cases} (0,1), & \text{if } t \text{ even} \\ (1,0), & \text{if } t \text{ odd.} \end{cases}$$

FTL has cost at least T while the best fixed strategy has cost $T/2$.

Be the leader

What if we had access to c_t in round t?

Be the leader lemma

Choosing strategy $M(c_{1:t})$ at time $t = 1, \ldots, T$ has non-positive regret,

$$\sum_{t=1}^{T} \langle c_t, M(c_{1:t}) \rangle \leq \langle c_{1:T}, M(c_{1:T}) \rangle = \min_{s \in \mathcal{S}} \sum_{t=1}^{T} \langle c_t, s \rangle.$$

Proof.

Assume for induction on T that $M(c_{1:t}) \leq \langle c_{1:T}, M(c_{1:T}) \rangle$. Then, we have for $T+1$ that

$$\begin{aligned}
\sum_{t=1}^{T+1} \langle c_t, M(c_{1:t}) \rangle &= \sum_{t=1}^{T} \langle c_t, M(c_{1:t}) \rangle + \langle c_{T+1}, M(c_{1:T+1}) \rangle \\
&\leq \langle c_{1:T}, M(c_{1:T}) \rangle + \langle c_{T+1}, M(c_{1:T+1}) \rangle \\
&\leq \langle c_{1:T}, M(c_{1:T+1}) \rangle + \langle c_{T+1}, M(c_{1:T+1}) \rangle \\
&= \langle c_{1:T}, M(c_{1:T}) \rangle.
\end{aligned}$$

□

Follow the *regularized* leader

Follow the regularized leader

For each round $t = 1, \dots, T$,

- The *learner* selects $s_t \in \mathcal{S}$ as follows,

$$s_t = \text{argmin}_{s \in \mathcal{S}} \langle c_{1:t-1}, s \rangle + \frac{1}{\gamma} h(x).$$

- The *adversary* selects a cost vector $c_t \in \mathcal{C}$.
- The *learner* suffers cost $\langle c_t, s_t \rangle$ and receives feedback c_t.

- $h : \mathcal{S} \to \mathbb{R}$ is a strongly-convex regularizer in some norm $||\cdot||$.
- $\gamma > 0$ is the learning rate.

Follow the *perturbed* leader: Additive version

Follow the perturbed leader: Additive version

Let $\epsilon > 0$ be the learning parameter. For each round $t = 1, \ldots, T$,

- Let $c_{1:t-1} = \sum_{k=1}^{t-1} c_k$.
- The *learner* samples $p_t \sim [0, 1/\epsilon]^d$.
- The *learner* choose strategy $s_t := M(c_{1:t-1} + p_t) = \text{argmin}_{s \in \mathcal{S}} \langle c_{1:t-1} + p_t, s \rangle$.
- The *learner* suffers cost $\langle c_t, s_t \rangle$ and receives feedback c_t.

Follow the *perturbed* leader: Multiplicative version

Multidimensional exponential distribution

Sampling a vector $x \in \mathbb{R}^d$ from an exponential distribution $\text{Exp}(\epsilon)$ means that we sample each coordinate independently according to the density $\mu(x) = \epsilon e^{-\epsilon x}$.

Follow the perturbed leader: Multiplicative version

Let $\epsilon > 0$ be the learning parameter. For each round $t = 1, \ldots, T$,

- Let $c_{1:t-1} = \sum_{k=1}^{t-1} c_k$.
- The *learner* samples $p_t \sim \text{Exp}(\epsilon)$
- The *learner* choose strategy $s_t \coloneqq M(c_{1:t-1} + p_t) = \text{argmin}_{s \in \mathcal{S}} \langle c_{1:t-1} + p_t, s \rangle$.
- The *learner* suffers cost $\langle c_t, s_t \rangle$ and receives feedback c_t.

Adversary

- The adaptive adversary maximizes the algorithm's expected regret

$$\max_{c_1,\ldots,c_T} \mathbb{E}\left[\mathcal{R}_{\mathcal{A}}(T)\right] = \max_{c_1,\ldots,c_T} \left(\mathbb{E} \sum_{t=1}^{T} \langle c_t, M(c_{1:t-1} + p_t) \rangle - \min_{s \in \mathcal{S}} \sum_{t=1}^{T} \langle c_t, s \rangle \right).$$

Adversary

- The adaptive adversary maximizes the algorithm's expected regret

$$\max_{c_1,\ldots,c_T} \mathbb{E}\left[\mathcal{R}_{\mathcal{A}}(T)\right] = \max_{c_1,\ldots,c_T} \left(\mathbb{E} \sum_{t=1}^{T} \langle c_t, M(c_{1:t-1} + p_t)\rangle - \min_{s\in\mathcal{S}} \sum_{t=1}^{T} \langle c_t, s\rangle \right).$$

- At round t, c_t is independent of p_t,

$$\mathbb{E}\langle c_t, M(c_{1:t-1} + p_t)\rangle = \langle c_t, \mathbb{E}[M(c_{1:t-1} + p_t)]\rangle.$$

Adversary

- The adaptive adversary maximizes the algorithm's expected regret

$$\max_{c_1,\ldots,c_T} \mathbb{E}\left[\mathcal{R}_{\mathcal{A}}(T)\right] = \max_{c_1,\ldots,c_T} \left(\mathbb{E} \sum_{t=1}^{T} \langle c_t, M(c_{1:t-1} + p_t) \rangle - \min_{s \in \mathcal{S}} \sum_{t=1}^{T} \langle c_t, s \rangle \right).$$

- At round t, c_t is independent of p_t,

$$\mathbb{E}\langle c_t, M(c_{1:t-1} + p_t)\rangle = \langle c_t, \mathbb{E}[M(c_{1:t-1} + p_t)]\rangle.$$

- The adaptive adversary maximizes

$$\max_{c_1,\ldots,c_T} \left(\sum_{t=1}^{T} \langle c_t, \mathbb{E}[M(c_{1:t-1} + p_t)] \rangle - \min_{s \in \mathcal{S}} \sum_{t=1}^{T} \langle c_t, s \rangle \right).$$

- This quantity is independent of the algorithms choices.

Adversary

- The adaptive adversary maximizes the algorithm's expected regret

$$\max_{c_1,\ldots,c_T} \mathbb{E}\left[\mathcal{R}_{\mathcal{A}}(T)\right] = \max_{c_1,\ldots,c_T} \left(\mathbb{E} \sum_{t=1}^{T} \langle c_t, M(c_{1:t-1} + p_t) \rangle - \min_{s \in \mathcal{S}} \sum_{t=1}^{T} \langle c_t, s \rangle \right).$$

- At round t, c_t is independent of p_t,

$$\mathbb{E}\langle c_t, M(c_{1:t-1} + p_t)\rangle = \langle c_t, \mathbb{E}[M(c_{1:t-1} + p_t)]\rangle.$$

- The adaptive adversary maximizes

$$\max_{c_1,\ldots,c_T} \left(\sum_{t=1}^{T} \langle c_t, \mathbb{E}[M(c_{1:t-1} + p_t)] \rangle - \min_{s \in \mathcal{S}} \sum_{t=1}^{T} \langle c_t, s \rangle \right).$$

- This quantity is independent of the algorithms choices.
- We can assume that the cost vectors are fixed in advance.
- We can assume that the algorithm only samples one p at the beginning,

$$\mathbb{E}_{p_1,\ldots,p_T} \sum_{t=1}^{T} \langle c_t, M(c_{1:t-1} + p_t) \rangle = \mathbb{E}_p \sum_{t=1}^{T} \langle c_t, M(c_{1:t-1} + p) \rangle.$$

Follow the perturbed leader: Additive version

Follow the perturbed leader: Additive version

Let $\epsilon > 0$ be the learning parameter and let $p \sim [0, 1/\epsilon]^d$. For each round $t = 1, \ldots, T$,

- Let $c_{1:t-1} = \sum_{k=1}^{t-1} c_k$.
- The *learner* choose strategy $s_t \coloneqq M(c_{1:t-1} + p) = \text{argmin}_{s \in \mathcal{S}} \langle c_{1:t-1} + p, s \rangle$.
- The *learner* suffers cost $\langle c_t, s_t \rangle$ and receives feedback c_t.

Follow the perturbed leader: Multiplicative version

Follow the perturbed leader: Multiplicative version

Let $\epsilon > 0$ be the learning parameter and Let $p \sim \text{Exp}(\epsilon)$. For each round $t = 1, \ldots, T$,

- Let $c_{1:t-1} = \sum_{k=1}^{t-1} c_k$.
- The *learner* chooses strategy $s_t \coloneqq M(c_{1:t-1} + p) = \text{argmin}_{s \in \mathcal{S}} \langle c_{1:t-1} + p, s \rangle$.
- The *learner* suffers cost $\langle c_t, s_t \rangle$ and receives feedback c_t.

Regret bounds: Additive guarantee

Parameters

Let D be the diameter of the strategy space, R be an upper bound on the costs and A be an upper bound on the norm of the costs,

- $||s - s'||_1 \leq D$ for all $s, s' \in \mathcal{S}$
- $|\langle c, s\rangle| \leq R$ for all $c \in \mathcal{C}, s \in \mathcal{S}$
- $||c||_1 \leq A$ for all $c \in \mathcal{C}$

Theorem (Kalai, Vempala [3])

Follow the perturbed leader FPL is polynomial time online learning algorithm such that for any sequence [1] *of cost vectors $c_1, ..., c_T \in \mathcal{C}$ and any learning parameter $\epsilon > 0$,*

$$\mathbb{E}[\mathcal{R}_{FPL}(T)] = \mathbb{E}\left[\sum_{t=1}^{T}\langle c_t, s_t\rangle\right] - \min_{s\in\mathcal{S}}\sum_{t=1}^{T}\langle c_t, s\rangle \leq \epsilon RAT + D/\epsilon$$

where $s_t \in \mathcal{S}$ is the (random) output of FPL at round $t = 1, \ldots, T$ when choosing p uniformly at random from $[0, 1/\epsilon]^d$.

[1] We assume that this sequence is fixed in advance.

Regret bounds: Additive guarantee

Corollary

Optimizing the bound in the previous theorem gives for $\epsilon = \sqrt{D/(RAT)}$,

$$\mathbb{E}[\mathcal{R}_{FPL}(T)] \leq 2\sqrt{DRAT}.$$

Solving online shortest path

Goal (Online shortest path)

We want to find a (randomized) algorithm $\mathcal{A}$ that runs both in polynomial time and has polynomial regret in expectation, $\mathbb{E}[\mathcal{R}_{\mathcal{A}}(T)] = O\left(poly(n)\sqrt{T}\right)$.

- We choose $\mathcal{S} = \{s \in \{0,1\}^E \mid s$ is the characteristic vector of a path from u to $v\}$
- M can be implemented by the Bellman-Ford algorithm.

Solving online shortest path

Parameters for online shortest path

Remember that D is the diameter of the strategy space, R an upper bound on the costs and A an upper bound on the norm of the costs,

- $||s - s'||_1 \leq ||s||_1 + ||s'||_1 \leq 2n = D$ for all $s, s' \in \mathcal{S}$
- $|\langle c, s \rangle| \leq ||c||_\infty \cdot ||s||_1 \leq n = R$ for all $c \in [0,1]^E, s \in \mathcal{S}$
- $||c||_1 \leq m = A$ for all $c \in [0,1]^E$

FPL for online shortest path

Follow the perturbed leader runs in polynomial time and has expected regret

$$\mathbb{E}[\mathcal{R}_{FPL}(T)] \leq O\left(\sqrt{n^2 m T}\right) \leq O\left(n^2 \sqrt{T}\right).$$

Regret bounds: Multiplicative guarantee

Parameters

Let D be the diameter of the strategy space, R be an upper bound on the costs and A be an upper bound on the norm of the strategies,

- $||s - s'||_1 \leq D$ for all $s, s' \in \mathcal{S}$
- $|\langle c, s\rangle| \leq R$ for all $c \in \mathcal{C}, s \in \mathcal{S}$
- $||s||_1 \leq A$ for all $s \in \mathcal{S}$

Theorem (Kalai, Vempala [3])

Follow the perturbed leader FPL is polynomial time online learning algorithm such that for any fixed sequence of cost vectors $c_1, ..., c_T \in \mathcal{C}$ and any learning parameter $\epsilon > 0$,

$$\mathbb{E}[\mathcal{R}_{FPL}(T)] = \mathbb{E}\left[\sum_{t=1}^{T}\langle c_t, s_t\rangle\right] - \min_{s\in\mathcal{S}}\sum_{t=1}^{T}\langle c_t, s\rangle \leq \epsilon \min_{s\in\mathcal{S}}\sum_{t=1}^{T}\langle c_t, s\rangle + 4AD(\ln d + 1)/\epsilon$$

where $s_t \in \mathcal{S}$ is the (random) output of FPL at round $t = 1, \ldots, T$ when choosing p from an exponential distribution Exp$(\epsilon/2A)$.

Regret bounds: Multiplicative guarantee

Corollary

Optimizing ϵ in the bound of the previous theorem gives

$$\mathbb{E}[\mathcal{R}_{FPL}(T)] \leq 4(AD(1+\ln d))\left(\min_{s\in\mathcal{S}}\sum_{t=1}^{T}\langle c_t, s\rangle\right) + 4AD(1+\ln d)$$

for

$$\epsilon = \min\left\{1, 2\sqrt{AD(1+\ln d)/\left(\min_{s\in\mathcal{S}}\sum_{t=1}^{T}\langle c_t, s\rangle\right)}\right\}.$$

Be the perturbed leader

- Assume we still know c_t in round t.
- First step: bound the error introduced by adding perturbations.

Lemma 1

Remember that $D \geq ||s - s'||_1$ for all $s, s' \in \mathcal{S}$. Then, for any fixed sequence of cost vectors $c_1, \dots, c_T$ and any perturbation vector p,

$$\sum_{t=1}^{T} \langle c_t, M(c_{1:t} + p) \rangle \leq \langle c_{1:T}, M(c_{1:T}) \rangle + ||p||_\infty \cdot D.$$

Proof.

Pretend that at time $t = 1$ the cost is $c_1 + p$. Then, by the BTL lemma,

$$\begin{aligned} \langle c_1 + p, M(c_1 + p) \rangle + \sum_{t=2}^{T} \langle c_t, M(c_{1:t} + p) \rangle &\leq \langle c_{1:T} + p, M(c_{1:T} + p) \rangle \\ &\leq \langle c_{1:T} + p, M(c_{1:T}) \rangle \end{aligned}$$

□

Be the perturbed leader

Proof (Cont.).

Rearranging,

$$\begin{aligned}
\sum_{t=1}^{T} \langle c_t, M(c_{1:t}+p)\rangle &\leq \langle c_{1:T}, M(c_{1:T})\rangle + \langle p, M(c_{1:T}) - M(c_1+p)\rangle \\
&\leq \langle c_{1:T}, M(c_{1:T})\rangle + ||p||_\infty \cdot |M(c_{1:T}) - M(c_{1:1}+p)|_1 \\
&\leq \langle c_{1:T}, M(c_{1:T})\rangle + ||p||_\infty \cdot D.
\end{aligned}$$

Follow the perturbed leader: Additive guarantee

- How much do "be the perturbed leader" and "follow the perturbed leader" differ?
- Perturbing the cost vectors creates an 'overlap' between the two cases.
- Observe that the distributions $c_{1:t-1} + p$ and $c_{1:t} + p$ are both uniform distributions over cubes.

Lemma 2

Let $C_1 = [0, 1/\epsilon]^d$ and let $C_2 = v + C_1$ be the first cube shifted by a vector v. Then,

$$Pr_{x \sim C_1}[x \notin C_2] \leq \epsilon ||v||_1.$$

Proof.

- Take $x \in [0, 1/\epsilon]^d$ uniformly at random.
- If $x \notin v + [0, 1/\epsilon]^d$, then $x_i \notin v_i + [0, 1/\epsilon]$ for some i.
- This happens with probability at most $\epsilon |v_i|$.
- By a union bound, we have that $x \notin v + [0, 1/\epsilon]^d$ with probability at most $\epsilon ||v||_1$.

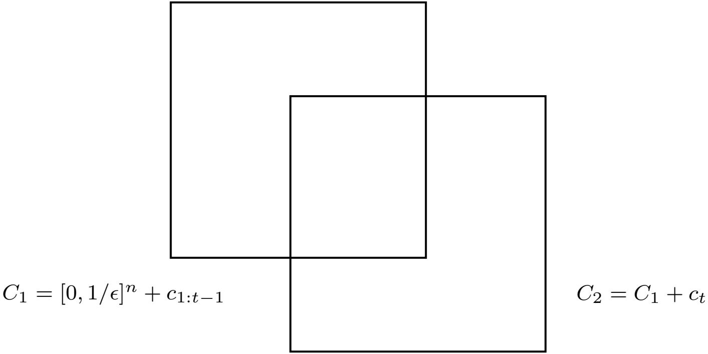
□

Follow the perturbed leader: Additive guarantee

Lemma 3

Remember that $\|c\|_1 \leq A$ for all $c \in \mathcal{C}$ and $|\langle c, s\rangle| \leq R$ for all $c \in \mathcal{C}, s \in \mathcal{S}$. Let p be drawn uniformly at random from $[0, 1/\epsilon]^d$. Then, for all $t = 1, \ldots, T$ and any cost vector $c_t \in \mathcal{C}$,

$$\mathbb{E}[\langle c_t, M(c_{1:t-1} + p)\rangle] \leq \mathbb{E}[\langle c_t, M(c_{1:t} + p)\rangle] + \epsilon AR.$$



Follow the perturbed leader: Additive guarantee

Lemma 3

Remember that $\|c\|_1 \leq A$ for all $c \in \mathcal{C}$ and $|\langle c, s\rangle| \leq R$ for all $c \in \mathcal{C}, s \in \mathcal{S}$. Let p be drawn uniformly at random from $[0, 1/\epsilon]^d$. Then, for all $t = 1, \dots, T$ and any cost vector $c_t \in \mathcal{C}$,

$$\mathbb{E}[\langle c_t, M(c_{1:t-1} + p)\rangle] \leq \mathbb{E}[\langle c_t, M(c_{1:t} + p)\rangle] + \epsilon AR.$$

Proof.

Let $C_1 = c_{1:t-1} + [0, 1/\epsilon]^d$ and $C_2 = c_t + C_1 = c_{1:t} + [0, 1/\epsilon]^d$. Then,

$$\begin{aligned}
\mathbb{E}_p[\langle c_t, M(c_{1:t-1} + p)\rangle] &= \mathbb{E}_{x\sim C_1}[\langle c_t, M(x)\rangle] \\
&= \mathbb{E}_{x\sim C_1}[\langle c_t, M(x)\rangle \mid x \in C_1 \cap C_2] \cdot Pr[x \in C_1 \cap C_2] \\
&+ \mathbb{E}_{x\sim C_1}[\langle c_t, M(x)\rangle \mid x \notin C_1 \cap C_2] \cdot Pr[x \notin C_1 \cap C_2] \\
&\leq \mathbb{E}_{x\sim C_1}[\langle c_t, M(x)\rangle \mid x \in C_1 \cap C_2] \cdot Pr[x \in C_1 \cap C_2] + R\epsilon||c_t||_1 \\
&= \mathbb{E}_{x\sim C_2}[\langle c_t, M(x)\rangle \mid x \in C_1 \cap C_2] \cdot Pr[x \in C_1 \cap C_2]] + R\epsilon||c_t||_1 \\
&\leq \mathbb{E}_{x\sim C_2}[\langle c_t, M(x)\rangle] + R\epsilon||c_t||_1 \\
&= \mathbb{E}_p[\langle c_t, M(c_{1:t} + p\rangle)] + R\epsilon||c_t||_1 .
\end{aligned}$$

□

Follow the perturbed leader: Additive guarantee

Theorem (Kalai, Vempala [3])

Follow the perturbed leader FPL is polynomial time online learning algorithm such that for any fixed sequence of cost vectors $c_1, ..., c_T \in \mathcal{C}$ and any learning parameter $\epsilon > 0$,

$$\mathbb{E}[\mathcal{R}_{FPL}(T)] = \mathbb{E}\left[\sum_{t=1}^{T} \langle c_t, s_t \rangle\right] - \min_{s \in \mathcal{S}} \sum_{t=1}^{T} \langle c_t, s \rangle \leq \epsilon RAT + D/\epsilon$$

where $s_t \in \mathcal{S}$ is the (random) output of FPL at round $t = 1, \ldots, T$ when choosing p uniformly at random from $[0, 1/\epsilon]^d$.

Proof.

Let $c_1, \ldots, c_T \in \mathcal{C}$ be a sequence of cost vectors and $s_t = M(c_{1:t-1} + p)])$ the output of FPL at round $t = 1, \ldots, T$ when choosing p uniformly at random from $[0, 1/\epsilon]^d$. Then, by Lemma 3,

$$\mathbb{E}\left[\sum_{t=1}^{T} \langle c_t, s_t \rangle\right] = \sum_{t=1}^{T} \mathbb{E}[\langle c_t, M(c_{1:t-1} + p)\rangle] \leq \sum_{t=1}^{T} \mathbb{E}[\langle c_t, M(c_{1:t} + p)\rangle] + \epsilon ART$$

□

Follow the perturbed leader: Additive guarantee

Proof (cont.)

Let $c_1, \ldots, c_T \in \mathcal{C}$ be a sequence of cost vectors and $s_t = M(c_{1:t-1} + p)])$ the output of FPL at round $t = 1, \ldots, T$ when choosing p uniformly at random from $[0, 1/\epsilon]^d$. Then, by Lemma 3,

$$\mathbb{E}\left[\sum_{t=1}^{T} \langle c_t, s_t \rangle\right] = \sum_{t=1}^{T} \mathbb{E}[\langle c_t, M(c_{1:t-1} + p)\rangle] \leq \sum_{t=1}^{T} \mathbb{E}[\langle c_t, M(c_{1:t} + p)\rangle] + \epsilon ART$$

By Lemma 1, we have that,

$$\sum_{t=1}^{T} \langle c_t, M(c_{1:t} + p)\rangle \leq \langle c_{1:T}, M(c_{1:T})\rangle + |p|_\infty \cdot D \leq \langle c_{1:T}, M(c_{1:T})\rangle + D/\epsilon.$$

We can conclude by BTL lemma,

$$\mathbb{E}\left[\sum_{t=1}^{T} \langle c_t, s_t \rangle\right] \leq \langle c_{1:T}, M(c_{1:T})\rangle + D/\epsilon + \epsilon ART.$$

Follow the perturbed leader: Multiplicative guarantee

Fact

Let $X_1, X_2, ..., X_d$ be independent random variables drawn from Exp(ϵ). Then,

$$\mathbb{E}\left[\max_{i\in[d]} X_i\right] \leq (1 + \ln d)/\epsilon.$$

Hence, we have that $||p||_\infty \leq (1 + \ln d)/\epsilon$.

Follow the perturbed leader: Multiplicative guarantee

As before, the distributions $c_{1:t-1}+p$ and $c_{1:t}+p$ overlap.

Lemma 4

Remember that $\|c\|_1 \leq A$ for all $c \in \mathcal{C}$. Let p be drawn from Exp(ϵ). Then, for all $t = 1, \ldots, T$ and any cost vector $c_t \in \mathcal{C}$,

$$\mathbb{E}[\langle c_t, M(c_{1:t-1}+p\rangle] \leq (1+2\epsilon A) \cdot \mathbb{E}[\langle c_t, M(c_{1:t}+p)\rangle].$$

Proof.

$$\begin{aligned}
\mathbb{E}[\langle c_t, M(c_{1:t-1}+p)\rangle] &= \int_x \langle c_t, M(c_{1:t-1}+x)\rangle d\mu(x) \\
&= \int_y \langle c_t, M(c_{1:t}+y)\rangle d\mu(y+c_t)) \\
&= \int_y \langle c_t, M(c_{1:t}+y)\rangle \cdot e^{\epsilon(||y-c_t||_1 - ||y||_1)} d\mu(y) \\
&= e^{\epsilon||c_t||_1} \cdot \mathbb{E}[\langle c_t, M(c_{1:t}\rangle + p_t)] \leq (1+2\epsilon A) \cdot \mathbb{E}[\langle c_t, M(c_{1:t}+p_t)\rangle]
\end{aligned}$$

Follow the perturbed leader: Multiplicative guarantee

Theorem (Kalai, Vempala [3])

Follow the perturbed leader FPL is polynomial time online learning algorithm such that for any fixed sequence of cost vectors $c_1, ..., c_T \in \mathcal{C}$ and any learning parameter $\epsilon > 0$,

$$\mathbb{E}[\mathcal{R}_{FPL}(T)] = \mathbb{E}\left[\sum_{t=1}^{T} \langle c_t, s_t \rangle\right] - \min_{s\in\mathcal{S}} \sum_{t=1}^{T} \langle c_t, s \rangle \leq \epsilon \min_{s\in\mathcal{S}} \sum_{t=1}^{T} \langle c_t, s \rangle + 4AD(\ln d + 1)/\epsilon$$

where $s_t \in \mathcal{S}$ is the (random) output of FPL at round $t = 1, \ldots, T$ when choosing p from an exponential distribution Exp$(\epsilon/2A)$.

Proof.

Let $c_1, \ldots, c_T \in \mathcal{C}$ be a sequence of cost vectors and $s_t = M(c_{1:t-1} + p)$ the output of FPL at round $t = 1, \ldots, T$ when choosing p from an exponential distribution Exp(ϵ). Then, by Lemma 4,

$$\mathbb{E}\left[\sum_{t=1}^{T} \langle c_t, s_t \rangle\right] = \sum_{t=1}^{T} \mathbb{E}[\langle c_t, M(c_{1:t-1} + p)\rangle] \leq \sum_{t=1}^{T} (1 + 2\epsilon A) \cdot \mathbb{E}[\langle c_t, M(c_{1:t}\rangle + p)]$$

$\square$

Follow the perturbed leader: Multiplicative guarantee

Proof (cont.)

Let $c_1, \ldots, c_T \in \mathcal{C}$ be a sequence of cost vectors and $s_t = M(c_{1:t-1} + p)$ the output of FPL at round $t = 1, \ldots, T$ when choosing p from an exponential distribution Exp(ϵ). Then, by Lemma 4,

$$\mathbb{E}\left[\sum_{t=1}^{T} \langle c_t, s_t \rangle\right] = \sum_{t=1}^{T} \mathbb{E}[\langle c_t, M(c_{1:t-1} + p)\rangle] \leq \sum_{t=1}^{T} (1 + 2\epsilon A) \cdot \mathbb{E}[\langle c_t, M(c_{1:t}\rangle + p)]$$

By Lemma 1, we have that,

$$\sum_{t=1}^{T} \mathbb{E}[\langle c_t, M(c_{1:t} + p)\rangle] \leq \langle c_{1:T}, M(c_{1:T})\rangle + \mathbb{E}[||p||_\infty] \cdot D \leq \langle c_{1:T}, M(c_{1:T})\rangle + D \cdot (1 + \ln d)/\epsilon.$$

We can conclude by BTL lemma,

$$\mathbb{E}\left[\sum_{t=1}^{T} \langle c_t, s_t \rangle\right] \leq (1 + 2\epsilon A) \left(\langle c_{1:T}, M(c_{1:T})\rangle + D \cdot (1 + \ln d)/\epsilon\right).$$

The theorem follows by choosing $\epsilon' = \epsilon/(2A)$.

A motivating example for the multi-armed bandit version of the problem

Consider a real-life application of the online shortest path problem:

Example (Routing services)

- Every day $t \in \{1, 2, ..., T\}$ our company needs to send goods from a point u to a point v.
- There are multiple routes. The travel time of each stretch of the route depends on unpredictable (weather, congestion) factors.
- At the end of each day, we learn the travel times of all the individual stretches of all the routes ("full feedback").
- As we have seen, we can use the follow the perturbed leader algorithm to solve this problem.

A motivating example for the multi-armed bandit version of the problem - part II

Now consider a more realistic setting:

Example (Routing services)

- Every day $t \in \{1, 2, ..., T\}$ our company needs to send goods from a point u to a point v.
- There are multiple routes. The travel time of each stretch of the route depends on unpredictable (weather, congestion) factors.
- Now, we only learn the travel time of *the route which we chose* (“multi-armed bandit feedback"), and we only receive the *total travel time* from the start vertex u to the end vertex v (“end-to-end feedback) as feedback.
- How do we select our route every day in this setting?

Using EXP3 for Online Path Selection with Bandit Feedback

Could try to apply Exp3 to Online shortest paths:

Exp3 algorithm for shortest paths

- Let $\mathcal{S}$ be the set of all paths from u to v and $\gamma = \sqrt{\log|S|/|S|T}$.
- Let $w_{1,s} = 1$ for each path $s \in \mathcal{S}$.
- for all $t = 1, ..., T$:
  - Set $x_s^t = \frac{w_s^t}{\sum_{s' \in \mathcal{S}} w_{s'}^t}$ for each path $s \in \mathcal{S}$.
  - Play path $s_t \in \mathcal{S} \sim x^t \in \Delta(S)$ and receive cost c_{t,s_t}
  - Set $\hat{c}_t = \frac{c_{t,s_t}}{x_{s_t}^t} \mathbf{e}_{s_t}$
  - Update $w^{t+1} = w^t e^{-\gamma \hat{c}_t}$

Theorem (Guarantee of EXP3, Auer et al.[1])

Assume that (EXP3) is run with a step-size $\gamma = \sqrt{\log|S|/|S|T}$. *Then , the following guarantee holds:*

$$\mathcal{R}_{EXP3}(T) = \mathbb{E}\left[\sum_{t=1}^{T}\langle c_t, s_t\rangle\right] - \min_{s\in\mathcal{S}}\sum_{t=1}^{T}\langle c_t, s\rangle \leq 2\sqrt{|S|\log|S|T},$$

where $s_t \in \mathcal{S}$ *is the (random) output of* $EXP3$ *at round* $t = 1, \ldots, T$.

- Issue: $|S|$ exponential in the number of edges.
- The regret $R_{\text{Exp3}} = O(\sqrt{|S|\log|S|T}) = O(2^{m/2}\sqrt{mT})$ is exponential in the number of edges m.
- Running time and space complexity is also exponential in the number of edges!

Goal (MAB setting)

We want an algorithm that runs both in polynomial time and space. Moreover, the regret of the algorithm should be polynomial in the number of edges m, and sublinear in the time horizon T.

Linear optimization with an efficiently computable optimizer and MAB feedback

To solve problem in the previous example, let us first formalize and generalize it.

Online Decision Problem with Multi-armed Bandit (MAB) Feedback

We are given a set of strategies $\mathcal{S} \subseteq \mathbb{R}^d$. At each round $t = 1, ..., T$,

- The algorithm picks a strategy $s_t \in \mathcal{S}$
- The adversary selects a cost vector $c_t \in \mathcal{C} \subseteq \mathbb{R}^d$
- The algorithm incurs cost $\langle c_t, s_t \rangle$, and this is the only feedback.

Oblivious adversary

We will assume that the adversary is oblivious, meaning that the cost vectors $c_1, \ldots c_T$ are fixed in advance.

Efficiently computable optimizer

We will assume that we have access to a function $M : \mathbb{R}^d \to \mathcal{S}$ that computes the best strategy given a cost vector in polynomial time,

$$M(c) = \text{argmin}_{s \in \mathcal{S}} \langle c, s \rangle.$$

Application: Online shortest paths revisited

How to interpret the routing example in terms of linear optimization with an efficiently computable optimizer and MAB feedback:

Online shortest paths problem as linear optimization

- Given a directed graph $G = (V, E)$ and a fixed pair of vertices u, v, want to find shortest path from u to v using at most H edges.
- Strategy space $S \subseteq \mathbb{R}^E$: Represent each path from u to v as a vector in $\{0,1\}^E$ by setting value 1 on each of its edges and value 0 on all other edges. Take $\mathcal{S}$ to be the discrete set representing all paths from u to v of length at most H.

MAB feedback for online shortest path

At each time step $t = 1, .., T$

- Pick a path $s_t \in \mathcal{S}$.
- Adversary selects a cost for each edge, represented as a cost vector $c_t \in [0,1]^E$.
- The incurred cost is then $\langle s_t, c_t \rangle$, i.e. the total length of the chosen path.
- *The only feedback is the total length of the selected path*, encoded as $\langle s_t, c_t \rangle$.

Remark: Given a cost vector $c \in \mathbb{R}^E$, we may compute the shortest path of length $\leq H$ efficiently by running the Bellman-Ford algorithm for H steps.

Further examples: MST

The setting of linear optimization with an efficiently computable optimizer and MAB also captures other important combinatorial problems:

Example (Minimum spanning tree)

- Given a graph $G = (V, E)$ want to find a spanning tree minimizing the weight.
- Strategy space: $\mathcal{S} \subseteq \mathbb{R}^E$ is the discrete set $\mathcal{S} \subseteq \{0, 1\}^E$ representing all spanning trees of G (each spanning tree can be represented as a vector in $\{0, 1\}^E$).
- The linear optimization oracle M can be implemented via Prim's or Kruskal's algorithm.
- At each time step $t = 1, 2, ...T$,
  - Select a spanning tree $s_t \in \mathcal{S}$.
  - Adversary selects a cost for each edge, which yields a cost vector $c_t \in \mathbb{R}^E$.
  - Cost and feedback at time t: Total cost of chosen spanning tree, encoded as $\langle c_t, s_t \rangle$.

Further example: Bipartite matching

Example (Maximum bipartite matching)

- Given a bipartite graph $G = (V, E)$ want to find a matching $M \subseteq E$ maximizing the weight of the matching $\sum_{e \in M} w_e$.
- Strategy space: $\mathcal{S} \subseteq \mathbb{R}^E$ is the discrete set $\mathcal{S} \subseteq \{0,1\}^E$ representing matchings of G (each matching can be represented as a vector in $\{0,1\}^E$).
- The linear optimization oracle M can be implemented via linear programming.
- At each time step $t = 1, 2, \ldots T$,
  - Select a matching $s_t \in \mathcal{S}$.
  - Adversary selects a weight w_e for each edge. The cost vector $c_t \in \mathbb{R}^E$ can be represented as $c_t(e) = -w_e$.
  - Cost and feedback at time t: Total weight of the matching, encoded as $\langle c_t, s_t \rangle$.

An efficient algorithm for MAB feedback

In the next slides, we will see an algorithm that achieves the goal against an oblivious adversary:

Theorem (Awerbuch, Kleinberg [2])

There exists a polynomial-time and space online learning algorithm $\mathcal{A}$ for MAB such that for any fixed sequence $c_1, \ldots, c_T \in \mathbb{R}^d$ with $|\langle c, s\rangle| \leq R$ for all $s \in \mathcal{S}$,

$$\mathbb{E}\left[\sum_{t=1}^{T}\langle c_t, s_t\rangle\right] - \min_{s\in\mathcal{S}}\sum_{t=1}^{T}\langle c_t, s\rangle \leq \mathcal{O}\left(T^{2/3}Rd^{5/3}\right)$$

where $s_t \in \mathcal{S}$ is the (random) output of $\mathcal{A}$ at round $t \in [T]$.

MAB: Idea behind the efficient algorithm

Recall the follow the perturbed leader algorithm FPL from the previous section:

Follow the Perturbed Leader (FPL) (for full feedback version)

For each round $t = 1, \ldots, T$,

- Let $c_{1:t-1} = \sum_{k=1}^{t-1} c_k$.
- Choose p_t uniformly at random from the hypercube $[0, 1/\epsilon]^d$.
- Choose strategy $s_t := M(c_{1:t-1} + p_t) = \text{argmin}_{s \in \mathcal{S}} \langle c_{1:t-1} + p_t, s \rangle$.
- Suffer cost $\langle c_t, s_t \rangle$ and receive feedback c_t.

MAB: Idea behind the efficient algorithm

Recall the regret guarantee for the follow the perturbed leader algorithm

Parameters

Let D be the diameter of the strategy space, R be an upper bound on the costs and A be an upper bound on the norm of the strategies,

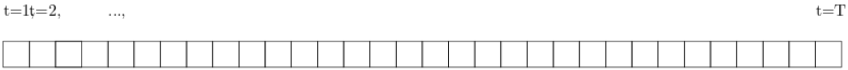
- $\|s - s'\|_1 \leq D$ for all $s, s' \in \mathcal{S}$
- $|\langle c, s\rangle| \leq R$ for all $c \in \mathcal{C}, s \in \mathcal{S}$
- $\|c\|_1 \leq A$ for all $c \in \mathcal{C}$

Theorem (Guarantee of FPL, Kalai, Vempala [3])

Let $c_1, \ldots, c_T \in \mathcal{C}$ be a sequence of cost vectors and let and $s_1, \ldots, s_T$ be the (random) sequence produced by the follow the perturbed leader algorithm FPL. Then

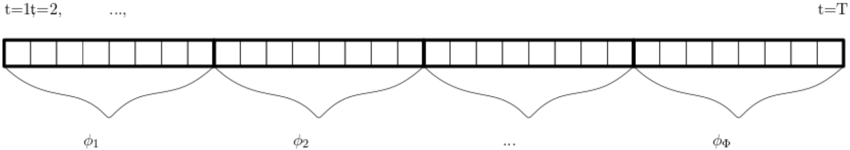
$$\mathcal{R}_{FPL}(T) = \mathbb{E}\left[\sum_{t=1}^{T} \langle c_t, s_t\rangle\right] - \min_{s\in\mathcal{S}} \sum_{t=1}^{T} \langle c_t, s\rangle \leq \epsilon RAT + D/\epsilon,$$

MAB: Illustration of the efficient algorithm



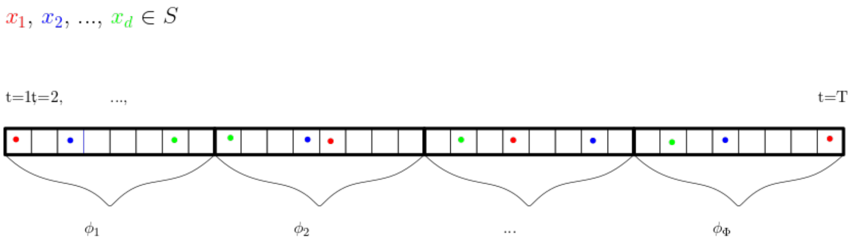
- We will leverage the FPL algorithm in the MAB setting as follows:

MAB: Illustration of the efficient algorithm



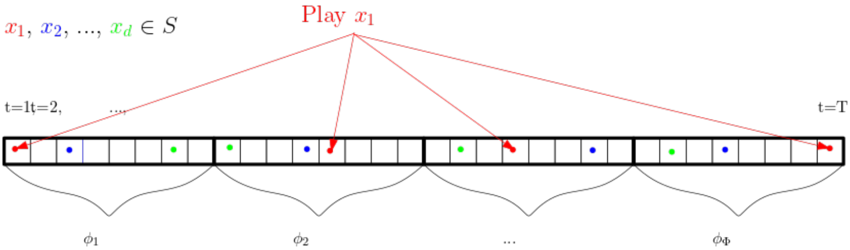
- Divide timeline into Φ phases of a fixed length τ.
- Each phase ϕ_i, $i = 1, \ldots, \Phi$ will simulate one time-step in the full feedback model.

MAB: Illustration of the efficient algorithm



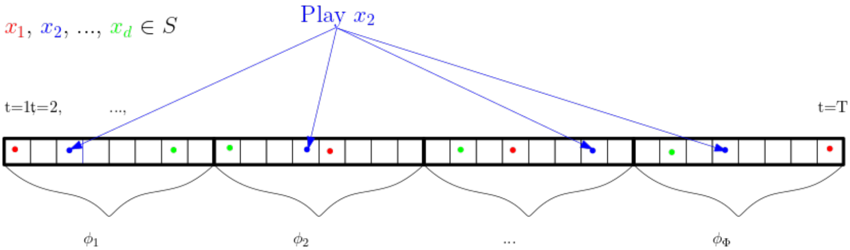
- Fix $x_1, \ldots x_d \in S$ (to be explained later)
- In each phase: Randomly subdivide time steps into “estimation steps" and “exploitation steps".
- In estimation steps: Play the x_i
- In exploitation steps: Play according the the FTL algorithm

MAB: Illustration of the efficient algorithm



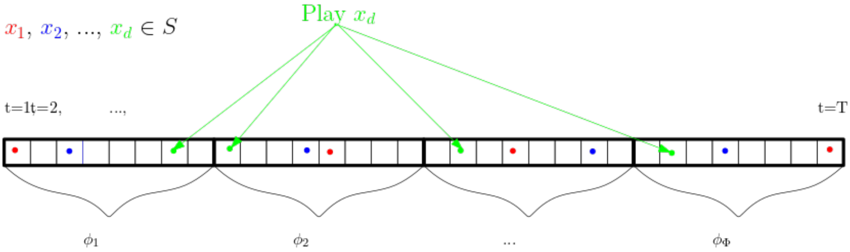
- In estimation steps: Play the x_i

MAB: Illustration of the efficient algorithm



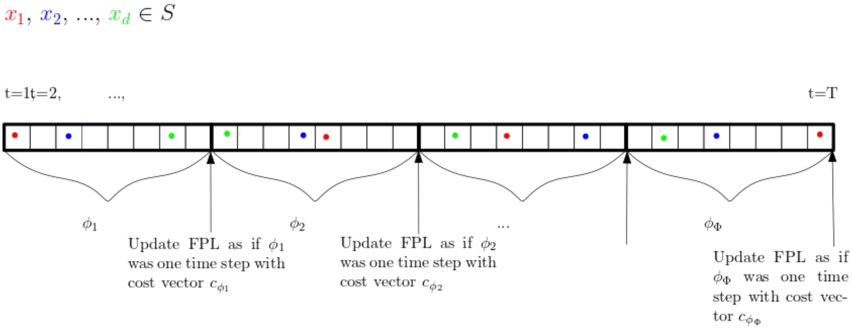
- In estimation steps: Play the x_i

MAB: Illustration of the efficient algorithm



- In estimation steps: Play the x_i

MAB: Illustration of the efficient algorithm



- During a phase ϕ: Use the feedback from the estimation steps to create an estimated cost vector c_ϕ.
- At the end of a phase ϕ: Update FPL as if ϕ was a single time step with cost vector c_ϕ.
- In the exploitation steps: Play according to the FPL algorithm.

MAB: How do we select the estimation steps

Definition

A set X of vectors in $\mathbb{R}^d$ is a basis if every element of $\mathbb{R}^d$ can be written in a unique way as a linear combination of elements of X.

- Want to select a basis $X = \{x_1, \ldots, x_d\} \subseteq \mathcal{S}$ of $\mathbb{R}^d$.
- We may assume that $\mathcal{S}$ is not contained in any proper linear subspace of $\mathbb{R}^d$ (otherwise we may decrease d).
- Therefore we know that there exists a basis $X = \{x_1, \ldots, x_d\} \subseteq \mathcal{S}$ of $\mathbb{R}^d$.
- It is possible to find such a basis in polynomial time, assuming that we have access to a an oracle for optimizing linear functions over $\mathcal{S}$.
- Why this is useful: Consider a fixed phase ϕ and let c_{x_i} denote be the observed cost of x_i in the estimation time step of x_i. If X is a basis, then there exists a unique vector c_ϕ such that $\langle c_\phi, x_j \rangle = c_{x_j}$ for all j.
- Can find this c_ϕ in polynomial time by linear interpolation.

MAB: Formal algorithm description

Let $FPL(c_1, ..., c_t)$ be the distribution over $\mathcal{S}$ given by the FPL algorithm after observing cost vectors $c_1, ...c_t$.

MAB algorithm

Parameters: ϵ, δ, $\tau := \lceil \frac{d}{\delta} \rceil$.
Select a basis $X = \{x_1, ..., x_d\} \subseteq \mathcal{S}$ of $\mathbb{R}^d$.
Partition T into $\Phi := \lceil \frac{T}{\tau} \rceil$ phases $\phi_1, \phi_2, ..., \phi_\Phi$ of length τ.
For each phase ϕ_i, $i = 1, ..., \Phi$:

- Select d time steps uniformly at random ("estimation steps"), and select a random correspondence to elements $x_1, ...x_d$ of X.
- In each estimation step: Play the corresponding x_i.
- In all other steps ("exploitation steps): Play $s \sim FPL(c_{\phi_1}, ..., c_{\phi_{i-1}})$
- At the end of the phase: Let c_{ϕ_i} be the unique vector such that $\langle c_{\phi_i}, x_j \rangle$ is the cost observed in the estimation step of x_j.

Analysis of the algorithm: Regret of the exploitation steps for a single phase

Claim

Let FPL_ϕ be the probability distribution over $\mathcal{S}$ specified by the FPL algorithm in phase ϕ, and let $x_\phi \sim FPL_\phi$ be a random sample from FPL_ϕ. Then

$$\mathbb{E}[\text{Cost of exploitation in phase } \phi] \leq \tau \mathbb{E}[\langle c_\phi, x_\phi \rangle].$$

Proof.

Let $\bar{c}_\phi := \frac{1}{\tau} \sum_{t \in \mathcal{T}_\phi} c_t$ be the average cost in phase ϕ, where $\mathcal{T}_\phi := \{\tau(\phi - 1) + 1, \tau(\phi - 1) + 2, ..., \tau\phi\}$ denotes the set of steps in phase ϕ.
Recall that c_ϕ is the vector such that $\langle c_\phi, x_j \rangle$ is the cost observed in estimation step of x_j for $j = 1, ...d$, and note that

$$\mathbb{E}[c_\phi] = \bar{c}_\phi,$$

since the adversary is oblivious and the time step at which we play x_j is chosen uniformly at random. □

Analysis of the algorithm: Regret of the exploitation steps for a single phase continued

Proof (cont.)

Moreover, note that c_ϕ and x_ϕ are independent in the oblivious adversary setting, since FPL_ϕ depends only on data *before* phase ϕ. Thus,

$$\begin{aligned}
\mathbb{E}[\langle c_\phi, x_\phi\rangle] &= \langle \mathbb{E}[c_\phi], \mathbb{E}[x_\phi]\rangle, && \text{since } c_\phi \text{ and } x_\phi \text{ are independent}\\
&= \langle \bar{c}_\phi, \mathbb{E}[x_\phi]\rangle && \text{since } \mathbb{E}[c_\phi] = \bar{c}_\phi,\\
&= \mathbb{E}[\langle \bar{c}_\phi, x_\phi\rangle], && \text{by linearity of expectation.}
\end{aligned}$$

So

$$\begin{aligned}
\mathbb{E}[\text{Cost of exploitation in phase } \phi] &\leq \mathbb{E}[\sum_{t\in\mathcal{T}_\phi} \langle x_\phi, c_t\rangle]\\
&= \tau\mathbb{E}[\langle \bar{c}_\phi, x_\phi\rangle], && \text{by linearity of expectation}\\
&= \tau\mathbb{E}[\langle c_\phi, x_\phi\rangle].
\end{aligned}$$

Analysis of the algorithm: Total regret of the exploitation

Parameters

Let D be the diameter of the strategy space, R be an upper bound on the costs and A be an upper bound on the norm of the strategies,

- $\|s - s'\|_1 \leq D$ for all $s, s' \in \mathcal{S}$
- $|\langle c, s \rangle| \leq R$ for all $c \in \mathcal{C}, s \in \mathcal{S}$
- $\|c\|_1 \leq A$ for all $c \in \mathcal{C}$

Claim

$\mathbb{E}$[Total cost from exploitation steps] $\leq \epsilon RAT + \frac{Dd}{\delta\epsilon} + \min_{s \in \mathcal{S}} \sum_{t=1}^{T} \langle c_t, s \rangle$.

Proof.

Let $\Phi = \frac{T}{\tau} = \frac{Td}{\delta}$ denote the number of phases.
Recall regret bound of FPL: Given cost vector $c_1, \ldots c_\phi$,

$$\mathbb{E}_{FPL}\left[\sum_{\phi=1}^{\Phi} \langle c_\phi, s_\phi \rangle\right] \leq \epsilon RA\Phi + \frac{D}{\epsilon} + \sum_{\phi=1}^{\Phi} \langle c_\phi, s \rangle, \text{ for all } s \in \mathcal{S}.$$

Analysis of the algorithm: Total regret of the exploitation

Proof cont.

Thus,

$$
\begin{aligned}
\mathbb{E}_{c_\phi, FPL}[\text{Total cost of exploitation}] &\leq \sum_{\phi=1}^{\Phi} \tau \mathbb{E}_{c_\phi, FPL}[\langle c_\phi, x_\phi \rangle], && \text{by the previous claim} \\
&\leq \tau(\epsilon R A \Phi + \frac{D}{\epsilon}) + \tau \min_{s \in \mathcal{S}} \sum_{\phi=1}^{\Phi} \mathbb{E}_{c_\phi}[\langle c_\phi, s \rangle], && \text{by } FPL \text{ guarantee} \\
&\leq \tau(\epsilon R A \Phi + \frac{D}{\epsilon}) + \min_{s \in \mathcal{S}} \sum_{t=1}^{T} \langle c_t, s \rangle, && \text{since } \mathbb{E}[c_\phi] = \bar{c}_\phi \\
&\leq \epsilon R A T + \frac{Dd}{\delta \epsilon} + \min_{s \in \mathcal{S}} \sum_{t=1}^{T} \langle c_t, s \rangle, && \text{Since } \Phi = T/\tau \text{ and } \tau = d/\delta.
\end{aligned}
$$

$\square$

Analysis of the algorithm: Putting it together

Theorem

There exists a polynomial-time and space online learning algorithm $\mathcal{A}$ for MAB such that for any fixed sequence $c_1, \ldots, c_T$,

$$\mathbb{E}\left[\sum_{t=1}^{T}\langle c_t, s_t\rangle\right] - \min_{s\in\mathcal{S}}\sum_{t=1}^{T}\langle c_t, s\rangle \le T^{2/3}R^{2/3}A^{1/2}D^{1/3}d^{1/3},$$

where $s_t \in \mathcal{S} \subseteq \mathbb{R}^d$ is the (random) output of $\mathcal{A}$ at round $t \in [T]$, and R, D, A are parameters such that $\|s - s'\|_1 \le D$ for all $s, s' \in \mathcal{S}$; $|\langle c, s\rangle| \le R$ for all $c \in \mathcal{C}, s \in \mathcal{S}$; $\|c\|_1 \le A$ for all $c \in \mathcal{C}$.

Proof.

The cost of the estimation steps is at most $Rd\Phi = RT\delta$, so by the previous claim, we have

$$\mathbb{E}\left[\sum_{t=1}^{T}\langle c_t, s_t\rangle\right] - \min_{s\in\mathcal{S}}\sum_{t=1}^{T}\langle c_t, s\rangle \le \epsilon RAT + \frac{DTd}{\delta\epsilon} + RT\delta = RT(\epsilon A + \delta) + \frac{DTd}{\delta\epsilon}$$

Setting $\epsilon = R^{-1/3}A^{-1/2}T^{-1/3}D^{1/3}d^{1/3}$ and $\delta = R^{-1/3}T^{-1/3}D^{1/3}d^{1/3}$ yields the result. □

Analysis of the algorithm: Getting rid of the parameter dependence

- What if D and A are really large? Can we get rid of the dependence on them in the final guarantee?
- Note that A, D are defined in terms of the l_1-norm. But the l_1-norm implicitly depends on the choice of basis!

Definition: l_1-norm

Given a basis $x_1, \dots x_d$ of $\mathbb{R}^d$, the l_1-norm of a vector $v = \sum_{i=1}^d \lambda_i x_i$ is given by

$$\|v\|_1 := \sum_{i=1}^d |\lambda_i|.$$

Analysis of the algorithm: Getting rid of the parameter dependence

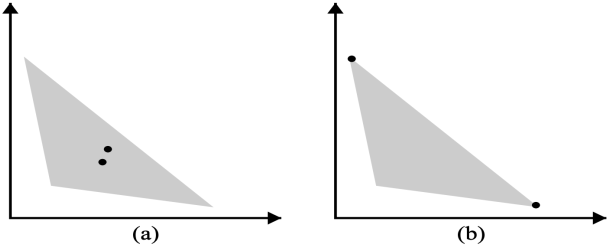


Figure: (a): A "bad" basis (b) A "well-spaced" basis

- To avoid large coefficients: Change the coordinate system by choosing a “well-spaced" basis.
- Then change the cost vectors so as to preserve $\langle c, s\rangle$ for all $c \in \mathcal{C}$, $s \in \mathcal{S}$.

How to choose a “well-spaced" basis: Barycentric spanners

Definition: Approximate barycentric spanner

Let $\mathcal{S} \subseteq \mathbb{R}^d$ be a subset which is not contained in any lower-dimensional linear subspace of $\mathbb{R}^d$. A set $X = \{x_1, \ldots x_d\}$ is a *C-approximate barycentric spanner* if every $s \in \mathcal{S}$ can be expressed as a linear combination of elements of X using coefficients in $[-C, C]$.

Theorem (Awerbuch, Kleinberg [2])

Suppose that $\mathcal{S} \subseteq \mathbb{R}^d$ is a compact set not contained in any proper subspace. Then for any $C \geq 1$, $\mathcal{S}$ has a C-approximate barycentric spanner.

Moreover, given an oracle for optimizing linear functions over $\mathcal{S}$, for any $C > 1$ we may compute a C-approximate barycentric spanner for $\mathcal{S}$ in polynomial time.

MAB: Analysis of algorithm

Theorem (Awerbuch, Kleinberg [2])

There exists a polynomial-time and space online learning algorithm $\mathcal{A}$ for MAB such that for any fixed sequence $c_1, \ldots, c_T \in \mathbb{R}^d$ with $|\langle c, s\rangle| \leq R$ for all $s \in \mathcal{S}$,

$$\mathbb{E}\left[\sum_{t=1}^{T} \langle c_t, s_t\rangle\right] - \min_{s\in\mathcal{S}} \sum_{t=1}^{T} \langle c_t, s\rangle \leq \mathcal{O}\left(T^{2/3} R d\right)$$

where $s_t \in \mathcal{S}$ is the (random) output of $\mathcal{A}$ at round $t \in [T]$.

Proof

Given $c \in \mathcal{C}, s \in \mathcal{S}$, let $c^s := \langle c, s\rangle$ denote the cost of a strategy $\mathcal{S}$ if the adversary chooses cost vector c. We will transform the coordinate system so that c^s is preserved:

- Let $X := \{x_1, \ldots x_d\} \subseteq \mathcal{S}$ be a 2-approximate barycentric spanner.
- Transform the coordinate system by mapping x_i to $Rd\mathbf{e}_i$.
- Transform the cost vectors $c \in \mathcal{C}$ into new cost vectors $\tilde{c}$ so that the cost of each strategy is preserved, i.e. so that $\langle \tilde{c}, s\rangle = c^s$ for all $c \in \mathcal{C}, s \in \mathcal{S}$ in the new coordinate system.

MAB: Analysis of algorithm

Proof cont.

Now we have that:

- $D = \max_{s,s' \in \mathcal{S}} \|s - s'\|_1 \leq 4Rd^2$ in this new coordinate system.
- The new cost vectors $\tilde{c}$ have no coordinate greater than $1/d$.
- So $A := \max_{\tilde{c}} \|\tilde{c}\|_1 \leq 1$ in the new coordinate system.

We have already seen that there exists a polynomial online learning algorithm $\mathcal{A}$ for MAB such that for any fixed sequence $c_1, \ldots, c_T$,

$$\mathbb{E}\left[\sum_{t=1}^{T} \langle c_t, s_t \rangle\right] - \min_{s \in \mathcal{S}} \sum_{t=1}^{T} \langle c_t, s \rangle \leq T^{2/3} R^{2/3} A^{1/2} D^{1/3} d^{1/3},$$

where D, A are parameters such that $\|s - s'\|_1 \leq D$ for all $s, s' \in \mathcal{S}$ and $\|c\|_1 \leq A$ for all $c \in \mathcal{C}$.
Setting $D = 4Rd^2$ and $A = 1$ yields the result. □

Remark: We can apply a similar argument to the FPL algorithm in the full feedback setting to remove the dependence on A and D in the guarantee.

References I

[1] P. Auer, N. Cesa-Bianchi, Y. Freund, and R.E. Schapire.
Gambling in a rigged casino: The adversarial multi-armed bandit problem.
In *Proceedings of IEEE 36th Annual Foundations of Computer Science*, pages 322–331, 1995.
(Cited on page 46.)

[2] Baruch Awerbuch and Robert D. Kleinberg.
Adaptive routing with end-to-end feedback: distributed learning and geometric approaches.
In László Babai, editor, *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pages 45–53. ACM, 2004.
(Cited on pages 51, 70, and 71.)

[3] Adam Kalai and Santosh Vempala.
Efficient algorithms for online decision problems.
Journal of Computer and System Sciences, 71(3):291–307, 2005.
(Cited on pages 26, 30, 37, 41, and 53.)