IEOR 8100-001: Learning and optimization for sequential decision making

Instructor: Shipra Agrawal

Industrial Engineering and Operations Research
Columbia University

...

Simple regret in $\mathcal{X}$-armed bandits

Space $\mathcal{X}$ of arms. On playing $x_t \in \mathcal{X}$, observe reward $r_t = f(x_t) + \epsilon$, ϵ is IID 0-mean bounded noise. **Unknown** f, can only evaluate.
Minimize "simple regret"

$$r_T = \sup_{x \in \mathcal{X}} f(x) - f(x_{T+1})$$

Regret measures how good is the final recommendation, compared to best.

How to explore optimally under budget of T? Also, called budgeted learning.

Assumptions

$\mathcal{X}$ is equipped with **known** metric $\ell(x, y)$, and f is Lipschitz with respect to $\ell(\cdot, \cdot)$.

That is, there is a known $\ell(\cdot, \cdot)$ such that for all $x, y, z \in \mathcal{X}$,

$$\ell(x, y) = \ell(y, x), \ell(x, x) = 0, \ell(x, z) \leq \ell(x, y) + \ell(y, z),$$

$$|f(x) - f(y)| \leq \ell(x, y)$$

Can be relaxed to $\mathcal{X}$ being equipped with semi-metric ℓ under which f is locally Lipschitz around optimal point x^*.

$$\ell(x, y) = \ell(y, x), \ell(x, x) = 0$$

$$|f(x^*) - f(y)| \leq \ell(x^*, y)$$

Example

- Euclidean space $\mathbb{R}^d$ is a metric space with $\ell(x, y) = \|x - y\|$ (Euclidean norm). Any 1-Lipschitz function f is Lipschitz with respect to ℓ.
- $\mathbb{R}^d$ with $\ell(x, y) = \|x - y\|^\alpha$ is metric space when $0 \leq \alpha \leq 1$.
- (for $\alpha > 1$ it is a semi-metric space.)

MDPs with large state space

- Solving a *known* MDP
  - K actions, Large state space S
  - Playing Tetris game: there are few actions for rotations of the new block, but many possible states.
  - many other game playing examples
- Large state space size challenges:
  - Usual MDP Algorithms like Value iteration algorithm are not implementable
  - Value iteration maintains $J^t(s)$ for every state s
  - In iteration $t+1$, it uses the complete vector J^t for update,

$$J^{t+1}(s) = \arg\max_a r_a(s) + \gamma \sum_{s'} p_a(s, s') J^t(s')$$

Monte Carlo Search approach

Simulation based approach, very popular in applications

- The MDP is known but large
- we have access to a generative model:
  - for any s, a we can simulate the action a in state s, observe the reward and next state s'.
- Overall structure of this algorithm: In each iteration t,
  - given current state s_t, **search** over all policies starting from state s_t, find best action a_t
  - execute the action a_t, generating transition to next state s_{t+1}
  - repeat again and again
- The **search** is called Monte Carlo tree search (MTCS)

Monte Carlo Tree Search (MTCS)

Searching for best action from state s_t: consider deterministic MDP with large state space.

- tree formed with root as s_t, K children as K possible next states.
- Edge weights are $\gamma^{d-1} r_a(s)$ for edge from node s at depth d to a^{th} child of s, where $a = 1 \ldots, K$.
- The value of taking a sequence $\mathbf{a}$ of actions starting in state s is given by weight of the corresponding path starting from s.
- Best action in s is the first action of a best path from s.

How to search over all paths in this large tree, with maximum number of T state-action evaluations allowed?

MTCS as $\mathcal{X}$-armed bandit

- Consider all infinite sequences of actions $\mathcal{A}^\infty$.
- Space $\mathcal{X}$ of arms is $\mathcal{A}^\infty$.
- Value of an arm $\mathbf{a} \in \mathcal{A}^\infty$, $f(\mathbf{a}) = \sum_{i=1}^{\infty} \gamma^{i-1} r_{a_i}(s_i)$.($s_{t+1}$ is determined by s_t, a_t)
- $|f(\mathbf{a}) - f(\mathbf{a}')| \leq \ell(\mathbf{a}, \mathbf{a}')$, where $\ell(\mathbf{a}, \mathbf{a}') = \frac{\gamma^{h(\mathbf{a},\mathbf{a}')}}{(1-\gamma)}$.
- $h(\mathbf{a}, \mathbf{a}')$ is the largest common prefix of $\mathbf{a}$ and $\mathbf{a}'$.

Even if we can observe r_{a_i} and hence $f(\mathbf{a})$ without any noise, the problem is difficult because of large number of arms (all sequences in $\mathcal{A}^\infty$).

Deterministic $\mathcal{X}$-armed bandits

Space $\mathcal{X}$ of arms. On playing $x_t \in \mathcal{X}$, observe reward $r_t = f(x_t)$.
Minimize "simple regret"

$$r_T = \sup_{x \in \mathcal{X}} f(x) - f(x_{T+1})$$

Challenge: Space $\mathcal{X}$ of arms is large, can evaluate f at only T points.
Utilize similarity structure given by metric space and Lipschitz assumption!

Optimistic approach

By Lipschitz condition: for any x, x_t, we have

$$f(x) \leq f(x_t) + \ell(x, x_t)$$

On evaluating f at x_t, we obtain an upper bound on $f(x)$ for all x. Optimistic search for the maximizer:

- At time t, pick $x_t = \max_{x \in \mathcal{X}} B_t(x)$, where

$$B_t(x) := \min_{k=1,\ldots,t-1} f(x_k) + \ell(x, x_t)$$

Upper bound improves for all x when ever we evaluate f at another point.

Illustration

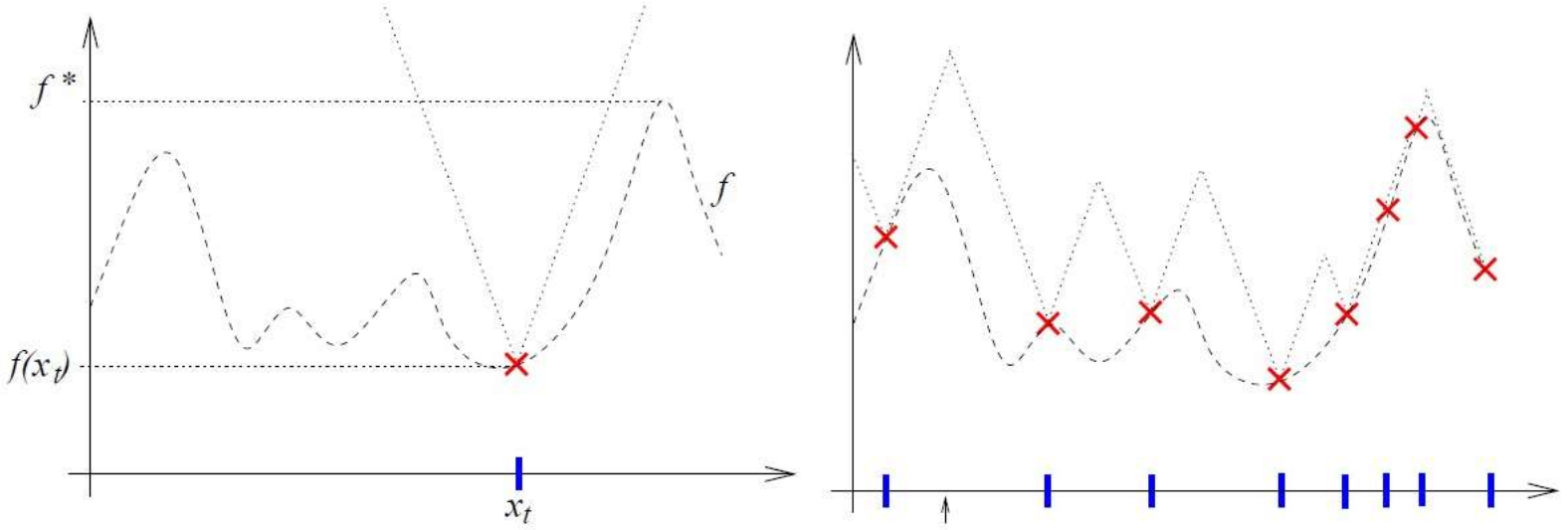


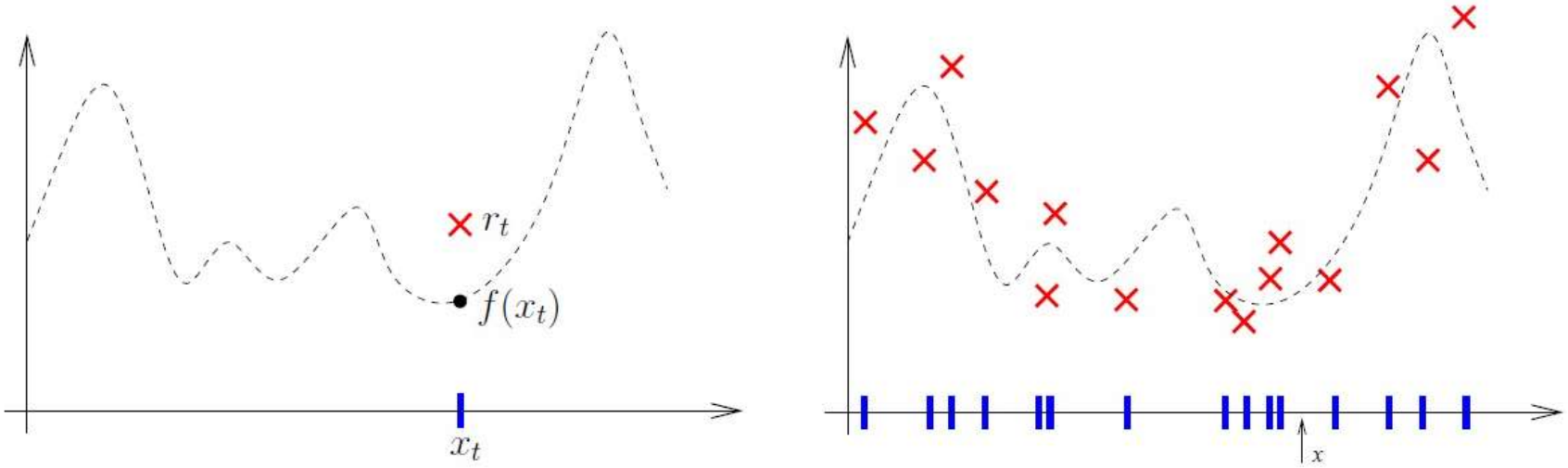
Figure 3.1: Left: The function f (dotted line) is evaluated at a point x_t, which provides a first upper bound on f (given the Lipschitz assumption). Right: several evaluations of f enable the refinement of its upper-bound. The optimistic strategy samples the function at the point with highest upper-bound.

Questions

- Is this a good way to choose x_t, does x_t provides good improvement in our knowledge of f?
- How to solve the maximization problem efficiently?
- How to extend it to Stochastic $\mathcal{X}$-armed bandit?

Stochastic $\mathcal{X}$-armed bandits

Perturbed evaluation, how to get an upper bound: UCB approach?



Use group of points

Average to reduce noise: use average over group of points X_i

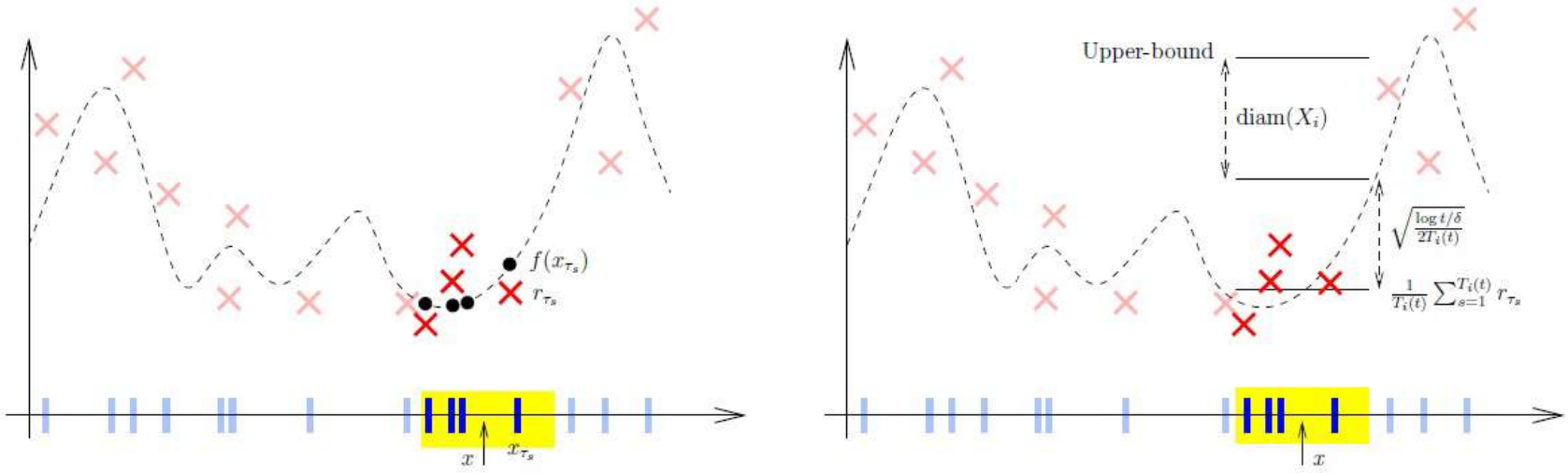


Figure 3.3: A possible way to define a high-probability bound on f at any $x \in \mathcal{X}$ is to consider a subset $X_i \ni x$ and average the $T_i(t)$ rewards obtained in this subset $\sum_{s=1}^{T_i(t)} r_{\tau_s}$, then add a confidence interval term $\sqrt{\frac{\log(t/\eta)}{2T_i(t)}}$, and add the diameter $\mathrm{diam}(X_i)$. This defines an UCB (with probability $1-\eta$) on f at any $x \in X_i$.

UCB

Given any group of points $X_i \subseteq \mathcal{X}$, with probability $1 - \delta$

$$|\frac{1}{n_{i,t-1}} \sum_{t:x_t \in X_i} r_t - \frac{1}{n_{i,t-1}} \sum_{t:x_t \in X_i} f(x_t)| \leq \sqrt{\frac{\log(t/\delta)}{2n_{i,t-1}}}$$

For $x, x' \in X_i$,

$$f(x) \leq f(x') + \text{diam}(X_i)$$

where $\text{diam}(X_i) = \max_{x,x' \in X_i} \ell(x, x')$.

Upper confidence Bound for all $x \in X_i$,

$$f(x) \leq B_t(X_i) := \frac{1}{n_{i,t-1}} \sum_{t:x_t \in X_i} r_t + \text{diam}(X_i) + \sqrt{\frac{\log(t/\delta)}{2n_{i,t-1}}}$$

Optimistic algorithm

At time t, pull arm

$$\arg\max_{x \in \mathcal{X}} B_t(X_{i(x)})$$

equivalently find

$$\arg\max_{i} B_t(X_i)$$

and play any point in X_i.

Tradeoffs in choice of group X_is

- Computational Efficiency: don't want large number of groups, or arbitrary shaped groups
- Accuracy: UCB is poor if either (a) diameter is large, or (b) X_i contains very few samples. Need not too large, not too small.

How to partition? Uniformly? Adaptively?
Hierarchical partition! Adaptively partition, zoom-in different parts of the space, based on which part needs more exploration.

Hierarchical partition for Euclidean 1-dim space

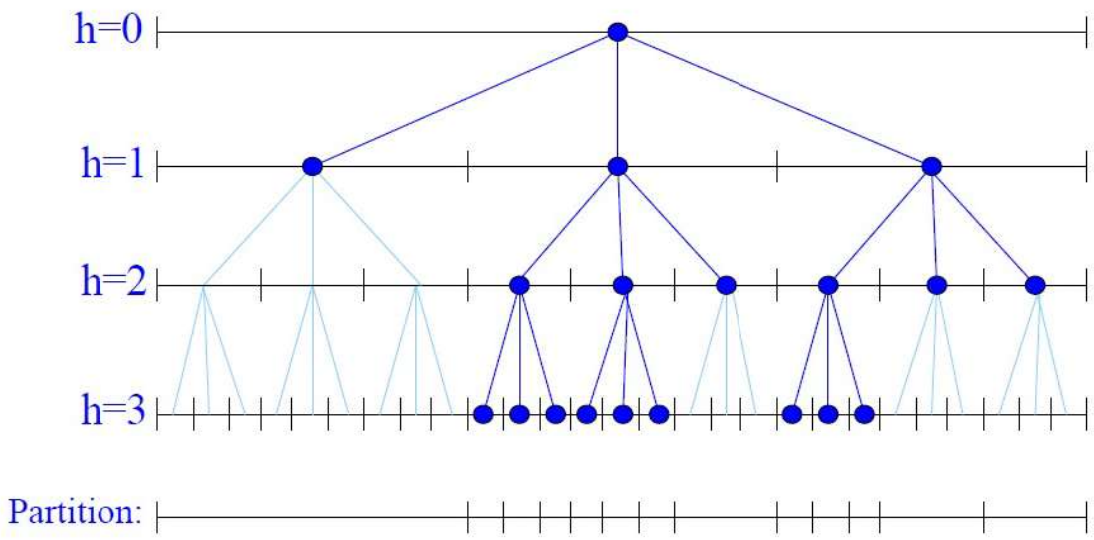


Figure 3.4: Hierarchical partitioning of the space $\mathcal{X}$ equivalently represented by a K-ary tree (here $K = 3$). The set of leaves of any subtree corresponds to a partition of $\mathcal{X}$.

- $x_{h,j}$ is center of cell h, j,
- $\delta(h) = \max_{j,x \in X_{h,j}} \ell(x_{h,j}, x)$ is upper bound on radius of all cells at depth h.

Algorithm for Deterministic $\mathcal{X}$-armed bandits

Define a hierarchical way to partition the space with following two properties (natural for Euclidean space) for every cell h, i:

- (Decreasing diameters) $\delta(h)$ decreases with h.
  - in above, $\delta(h) = \delta(h-1)/3$
- (well shaped cells) There exists $\nu > 0$ such that for any h, all cells at depth h contain a ℓ-ball of radius $\nu\delta(h)$.
  - in above $\nu = 1$

We will later see this is also natural to define when $\mathcal{X}$ is paths from root to leaf in a tree, like in Monte Carlo Tree Search problem for solving large MDPs.

Optimistic zooming algorithm

[Kleinberg et al. 2008, Bubeck et al. 2008, Munos 2014]

Adaptively zoom in the areas which look promising from an optimistic lense

Initialization: $\mathcal{T}_1 = \{(0,0)\}$ (root node)
for $t = 1$ to n **do**
  Select the leaf $(h,j) \in \mathcal{L}_t$ with maximum $b_{h,j} \stackrel{\text{def}}{=} f(x_{h,j}) + \delta(h)$ value.

  Expand this node: add to $\mathcal{T}_t$ the K children of (h,j) and evaluate the function at the points $\{x_{h+1,j_1}, \ldots, x_{h+1,j_K}\}$
end for
Return $x(n) = \arg\max_{(h,i) \in \mathcal{T}_n} f(x_{h,i})$

Simple regret bound

$$\mathcal{X}_\epsilon \stackrel{\text{def}}{=} \{x \in \mathcal{X}, f(x) \geq f^* - \epsilon\}$$

the set of ϵ-optimal states.

Definition 3.1. The **η-near-optimality dimension** is the smallest $d \geq 0$ such that there exists $C > 0$, for all $\epsilon > 0$, the maximal number of disjoint ℓ-balls of radius $\eta\epsilon$ with center in $\mathcal{X}_\epsilon$ is less than $C\epsilon^{-d}$.

In below, d is ν-near optimality dimension.

Corollary 3.3. Assume that $\delta(h) = c\gamma^h$ for some constants $c > 0$ and $\gamma < 1$.

- If $d > 0$, then the loss decreases polynomially fast:

$$r_n \leq \left(\frac{C}{1-\gamma^d}\right)^{1/d} n^{-1/d}.$$

- If $d = 0$, then the loss decreases exponentially fast:

$$r_n \leq c\gamma^{(n/C)-1}.$$