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

Instructor: Shipra Agrawal

Industrial Engineering and Operations Research
Columbia University

$\mathcal{X}$-armed bandits, Monte Carlo Tree Search

# $\mathcal{X}$-armed bandits

Space $\mathcal{X}$ of arms. On playing $x_t \in \mathcal{X}$, observe reward $r_t = f(x_t) + \epsilon$, $\epsilon$ 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*, and has close connection to *active 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 $\ell$ 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 $\ell$.
- $\mathbb{R}^d$ 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

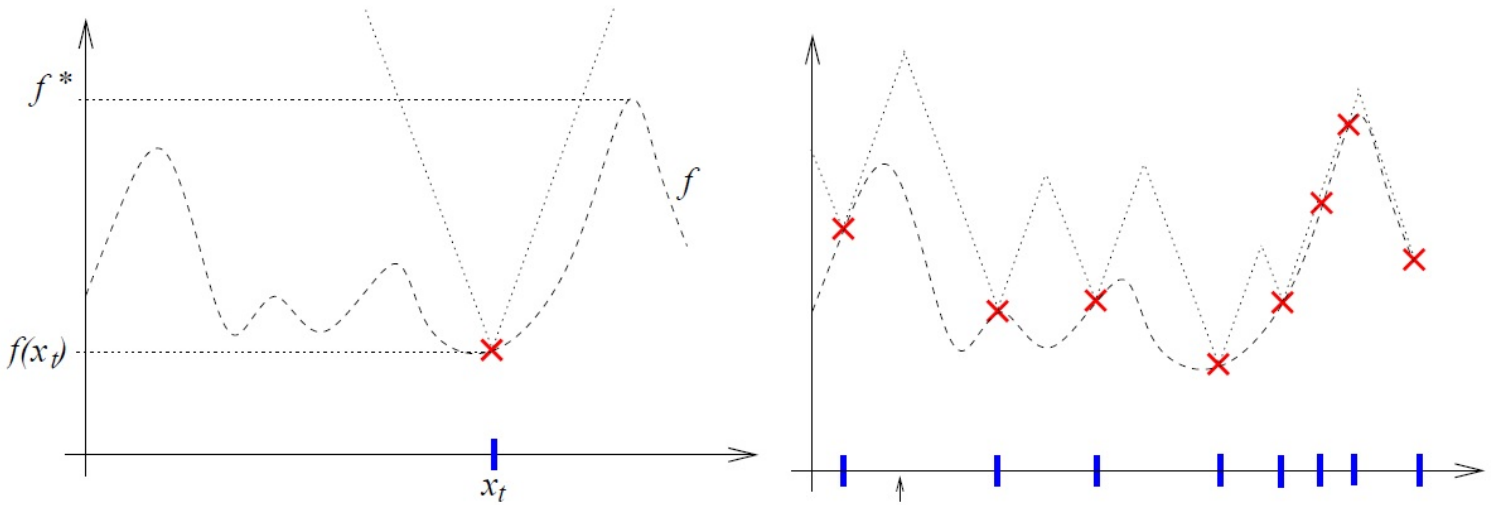


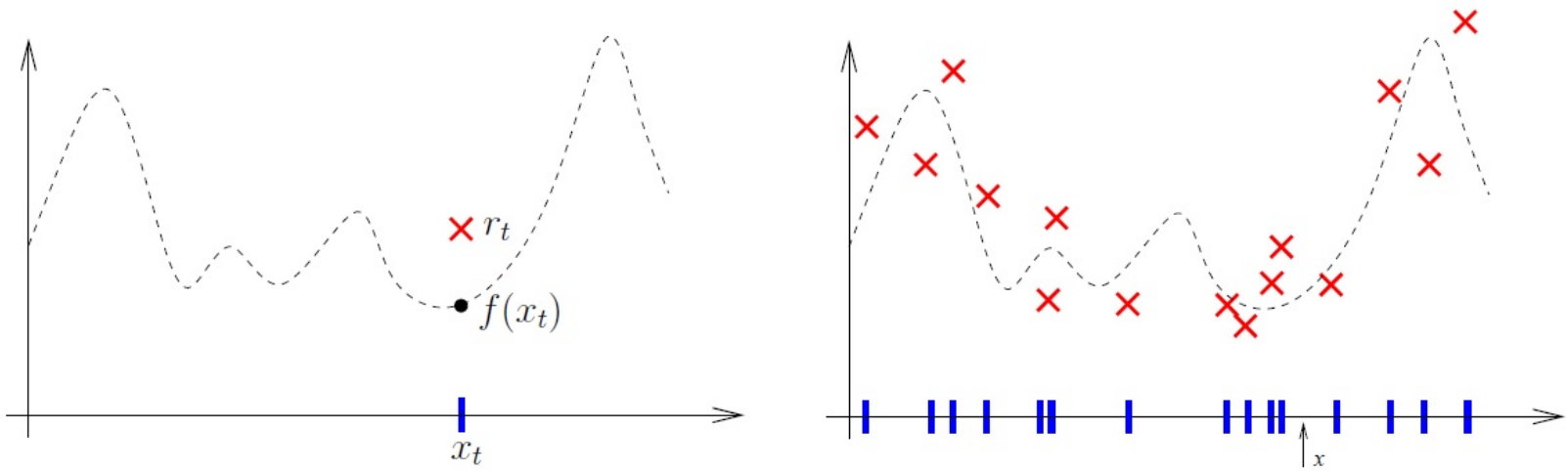

**Figure 3.1:** Left: The function $f$ (dotted line) is evaluated at a point $x_t$, which provides a first upper bound on $f$ (given the Lipschitz assumption). Right: several evaluations of $f$ enable the refinement of its upper-bound. The optimistic strategy samples the function at the point with highest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

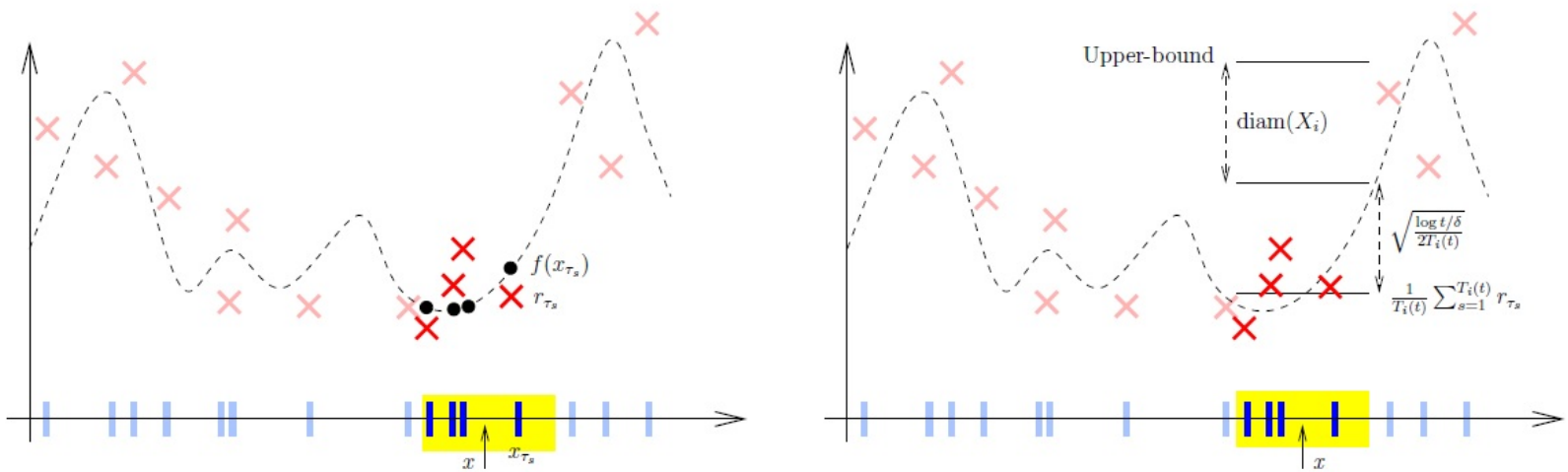



**Figure 3.3:** A possible way to define a high-probability bound on $f$ at any $x \in \mathcal{X}$ is to consider a subset $X_i \ni x$ and average the $T_i(t)$ rewards obtained in this subset $\sum_{s=1}^{T_i(t)} r_{\tau_s}$, then add a confidence interval term $\sqrt{\frac{\log(t/\eta)}{2T_i(t)}}$,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

- 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.

# For remaining lecture: only deterministic $\mathcal{X}$-armed bandits

Space $\mathcal{X}$ of arms. On playing $x_t \in \mathcal{X}$, observe reward $r_t = f(x_t)$ (no noise).
$\mathcal{X}$ is metric space equipped with metric $\ell$, and $f$ is Lipschitz:

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

Minimize "simple regret"

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

# Hierarchical partition

Define a hierarchical partitioning scheme for space $\mathcal{X}$

- Root $(0, 0)$ is the whole space $\mathcal{X}$
- $(h, j)$ is the $j^{th}$ cell at depth $h$, center $x_{h,j}$.
- $\delta(h)$ is upper bound on radius of all cells at depth $h$.

$$\delta(h) = \max_{j, x \in X_{h,j}} \ell(x_{h,j}, x)$$

# Hierarchical partitioning

We need following two properties for every cell $h, j$:

- (Decreasing radius) $\delta(h)$ decreases with $h$.
- (well shaped cells) There exists $\nu > 0$ such that for any $h$, all cells $j$ at depth $h$ contain a $\ell$-ball of radius $\nu\delta(h)$ centered at $x_{h,j}$.

# Optimistic zooming algorithm

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

Initialize set of leaf nodes $\mathcal{L} = (0, 0)$ (root).
For $t = 1, \ldots, n = \frac{T}{K}$,

- Pick $(h, j) \in \mathcal{L}$ with maximum value of $B(h, j) = f(x_{h,j}) + \delta(h)$
- Evaluate $f$ at center of all children $(h + 1, j_1) \ldots, (h + 1, j_K)$ of $(h, j)$
- Remove $(h, j)$ and add all its children of this cell to $\mathcal{L}$

At end of procedure, pick among evaluated cells, $x_{h,j}$ with highest value of $f(x_{h,j})$.

Note that in above, number of function evaluations is $T = nK$.

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

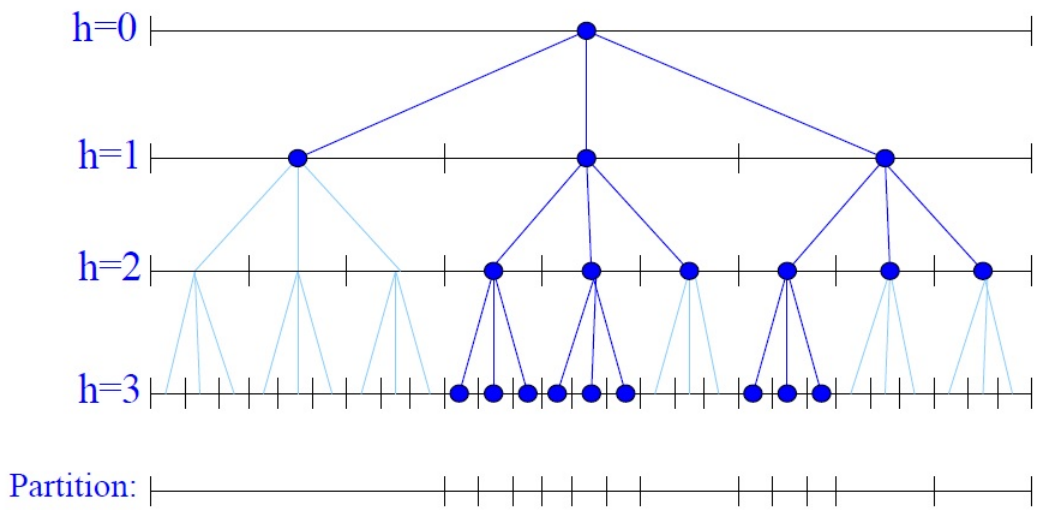



**Figure 3.4:** Hierarchical partitioning of the space $\mathcal{X}$ equivalently represented by a $K$-ary tree (here $K = 3$). The set of leaves of any subtree corresponds to a partition of $\mathcal{X}$.

# Examples

- $\mathcal{X} = [-1, 1]$, $\ell(x, y) = |x - y|^{\alpha}$. Define partition with $\delta(h) = 2^{-\alpha h}$, $\nu = 1$.
- $\mathcal{X} = [-1, 1]^D$, $\ell(x, y) = \|x - y\|_{\infty}^{\alpha}$, $\ell(x, y) = \|x - y\|_1^{\alpha}$, $\nu = 1$. Define partition with $\nu = 1$, $\delta(h) = 2^{-\alpha h}$ and $\delta(h) = D2^{-\alpha h}$ respectively.

# Regret Analysis

## Bound on simple regret

**Lemma**

*Let $h_{\max}, j_{\max}$ is deepest node to be expanded.*

$$r_n \leq \delta(h_{\max})$$

*If $\delta(h) = c\gamma^h$ for some $\gamma < 1$, $r_n \leq c\gamma^{h_{\max}}$*

Because $(h_{\max}, j_{\max})$ was picked for expansion at some time $t$, instead of the cell with $x^*$,

$$\begin{aligned} f(x^*) &\leq B(h_{\max}, j_{\max}) \\ &\leq f(x_{h_{\max}, j_{\max}}) + \delta(h_{\max}) \\ &\leq f(x_{\tilde{h}, \tilde{j}}) + \delta(h_{\max}) \end{aligned}$$

where $x_{\tilde{h}, \tilde{j}}$ denotes the final output, chosen to maximize value of $f$ among evaluated points.

# How large is $h_{\max}$?

## Worst-case lower bound

- uniform breadth first search would give $n \leq \frac{K^{h_{\max}+1}-1}{K-1}$, so that

$$h_{\max} = \Omega(\log(nK)/\log(K))$$

- If $\delta(h) = c\gamma^h$ for some $\gamma < 1$, this gives

$$r_n = O(n^{-C\frac{\log(1/\gamma)}{\log(K)}})$$

- For Euclidean space examples ($K = 2^D, \gamma = 2^{-\alpha}$), giving $O(n^{-C\alpha/D})$ regret.

Can we get better bound on $h_{\max}$ for the optimistic search?

# Problem-dependent regret bound

Use that our algorithm only expands $\delta(h)$-near optimal cells!

- if $(h, j)$ is NOT $\delta(h)$ near-optimal, then

$$\begin{aligned} B(h,j) &= f(x_{h,j}) + \delta(h) \\ &< f(x^*) - \delta(h) + \delta(h) \\ &\leq B(\text{cell}(x^*)) \end{aligned}$$

  ($\text{cell}(x^*)$ is the cell currently containing $x^*$.)
  Would rather expand $\text{cell}(x^*)$!

In worst case, many $\delta(h)$-near optimal cells.
Can we get better problem-dependent bound on $h_{\max}$?

# Near optimality dimension $d$

## Problem dependent parameter

- Smallest $d$ such that the subset of $\epsilon$-optimal points can fit at most $\frac{C}{\epsilon^d}$ disjoint balls of radius $\nu\epsilon$.

**Definition**

An $\nu$-near optimality dimension is the smallest $d \geq 0$ such that there exists $C > 0$, for all $\epsilon > 0$, the maximal number of disjoint $\ell$-balls of radius $\nu\epsilon$ in $\mathcal{X}_\epsilon$, is less than $C\epsilon^{-d}$.

Implies: $\delta(h)$-optimal points can occur in at most $\frac{C}{\delta(h)^d}$ cells at depth $h$.

Main observation

**Lemma**

*$h(n)$ be smallest $h \geq 0$ such that $\sum_{g=0}^{h} \frac{C}{\delta(g)^d} = n$, where $d$ is $\nu$-near optimality dimension. Then,*

$$h_{\max} \geq h(n).$$

**Lemma**

*If $\delta(h) = c\gamma^h$ for constant $\gamma < 1$, then*

$$h_{\max} \geq h(n) = \Omega(\frac{\log(n)}{d})$$

*If $d = 0$, then,*

$$h_{\max} \geq h(n) = \Omega(n)$$

Effectively, $\log(K)$ is replaced by near-optimality dimension $d$, instead of actual dimension $D$.

Following result is from [Munos 2014]. Here, $d$ is $\nu$-near optimality dimension, $n = T/K$.

**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:

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

# Examples

## Near-optimality dimension

Consider $\mathcal{X} = [-1, 1]^D$, $\ell(x, y) = \|x - y\|_\infty^\beta$, $f(x) = 1 - \|x\|_\infty^\alpha$.
Then,

$$d = D\left(\frac{1}{\beta} - \frac{1}{\alpha}\right)$$

# Example: Monte Carlo Tree Search in discounted MDP

- $\mathcal{X} = A^{\infty}$, all feasible infinite sequence of actions in a deterministic MDP starting at given state $s$.
- $\ell(\mathbf{a}, \mathbf{a}') = \frac{\gamma^{h(\mathbf{a},\mathbf{a}')}}{1-\gamma}$, where $h(\mathbf{a}, \mathbf{a}')$ is the length of longest common prefix.
- natural partition using transition tree with $\delta(h) = \frac{\gamma^h}{(1-\gamma)}$.

# Example: Monte Carlo Tree Search in discounted MDP

- Given starting state $s_1$ at node 0,

$$f(\mathbf{a}) = \sum_{t=1}^{\infty} r_{a_t}(s_t)\gamma^{t-1}$$

  Here, we have deterministic MDP, $s_{t+1}$ is the next state when $a_t$ is played in state $s_t$.
- Evaluating $f$ completely is not possible. Approximate $f$ at cell of depth $h$ by:

$$u(h, \mathbf{a}) = \sum_{t=1}^{h} r_{a_t}(s_t)\gamma^{t-1}$$

- Upper bound on $f$ for a cell:

$$B_t(h, \mathbf{a}) = u(\mathbf{a}, h) + \frac{\gamma^h}{1-\gamma} = u(h, \mathbf{a}) + \delta(h)$$

  $f(\mathbf{a}') + \delta(h) \geq B_t(h, \mathbf{a}) \geq f(\mathbf{a}')$ for all $\mathbf{a}'$ in this cell, as common prefix with center $\mathbf{a}$ is at least length $h$.

- Expanding a cell:
  - Children contain sequences with prefix $(\mathbf{a}^h, a), a = 1, \ldots, K$, let corresponding states at time $h+1$ be $s^1, \ldots, s^K$.
  - evaluate only $r_a(s^a)$,

$$u(\mathbf{a}a, h+1) = u(\mathbf{a}, h) + \gamma^h r_a(s^a)$$

# Algorithm

Initialize set of leaf nodes $\mathcal{L} = (0, 0)$ (root or starting state). For $t = 1, \ldots, n = T/K$,

- Pick cell $(h, j) \in \mathcal{L}$ (center $\mathbf{a}$ state $s$), with maximum value of

$$B(h, j) = u(\mathbf{a}, h) + \frac{\gamma^h}{1 - \gamma}$$

- Compute $r_a(s)$, $a = 1, \ldots, k$, update $u(\mathbf{a}a, h + 1)$ for $K$ children.
- Remove $(h, j)$ and add all its children of this cell to $\mathcal{L}$

At end, return cell $\tilde{h}, \tilde{j}$ with maximum value of $u(h, j)$. Pick the first action in this $\tilde{h}$ length action sequence $\tilde{\mathbf{a}}$.

# Regret

**Lemma**

*Let $h_{\max}$ is the depth of deepest node to be expanded.*

$$r_n \leq \delta(h_{\max}) = \frac{\gamma^{h_{\max}}}{1-\gamma}$$

Proof: Because it was picked for expansion

$$\begin{aligned} f(\mathbf{a}^*) &\leq B(h_{\max}, j_{\max}) \\ &\leq u(h_{\max}, j_{\max}) + \delta(h_{\max}) \\ &\leq u(\tilde{h}, \tilde{j}) + \delta(h_{\max}) \\ &\leq f(\tilde{\mathbf{a}}) + \delta(h_{\max}) \end{aligned}$$

# Regret Bound

How big is $h_{\max}$? In worst case (uniform expansion), $(K^h + 1 - 1)/(K - 1) = n$, so that

$$h_{\max} \simeq \frac{\log(nK)}{\log(K)}$$

to get regret

$$\delta(h_{\max}) = O(nK)^{-\frac{\log(1/\gamma)}{\log(K)}}$$

Can get this by uniform search!

# Problem-dependent bound

## Adapted concept of near-optimality dimension

Use that our algorithm only expands $\delta(h)$ optimal cells at depth $h$.

**Branching factor:** What is the maximum uniform branching factor (worst-case $K$) needed to cover $\delta(h)$-near-optimal prefixes of length $h$ for all $h \geq 1$.

$$\kappa = \lim \sup_{h \to \infty} \| \mathbf{a} \in \mathcal{A}^h : \sup_{\mathbf{a}'} f(\mathbf{a}\mathbf{a}') \geq f(\mathbf{a}^*) - \delta(h) \|^{1/h}$$

Effectively, can replace $K$ by $\kappa$. If $\kappa > 1$,

$$h_{\max} \geq \Omega(\frac{\log(n\kappa)}{\log(\kappa)})$$

If $\kappa = 1$,

$$h_{\max} = \Omega(n) \text{ giving } O(\gamma^{Cn}) \text{ regret}$$