Adversarial Online Learning with Changing Action Sets: Efficient Algorithms with Approximate Regret Bounds

Ehsan Emamjomeh-Zadeh, **Chen-Yu Wei**, Haipeng Luo, David Kempe

University of Southern California

A horse racing example

- There is a set of horses.
- In each round, a subset of horses are chosen (arbitrarily) to compete with each other.
- In each round, you can predict the winner. If you correctly predict it, you get $20.

A horse racing example

If not all horses join the competition in all rounds
→ a "learning from sleeping experts" problem
[Kleinberg, Niculescu-Mizil, Sharma'08]

Input: N (number of horses / experts / actions)

For $t = 1, 2, \dots, T$

  Environment reveals $S_t \subseteq [N]$

  Choose $a_t \in S_t$ and suffers $\ell_t(a_t)$

  Observe $\ell_t(a)$ for all $a \in S_t$

  // $\ell_t(a) = 0$ if a wins; $\ell_t(a) = 1$ otherwise

Measuring the Performance

For sleeping expert problem, one benchmark is the "total loss of the best ranking" [Kleinberg, Niculescu-Mizil, Sharma'08].

$$\hat{L} \triangleq \sum_{t=1}^{T} \ell_t(a_t)$$

$$L^\star \triangleq \min_{\sigma \in \text{ ranking over } [N]} \sum_{t=1}^{T} \ell_t\left(\sigma(S_t)\right)$$

Example. Let σ specifies the ranking $3 \to 4 \to 1 \to 2 \to 5$. Then

$$\sigma(\{2,4,5\}) = 4$$

Note: the learner need not pick a ranking; only the benchmark $L^\star$ is defined though a ranking.

$$\text{Reg} \triangleq \hat{L} - L^\star$$

Known results for sleeping expert problems

Reg = $o(T)$ is possible. Hedging over the strategies (#strategies = $N!$) using the exponential weight algorithm [Kleinberg, Niculescu-Mizil, Sharma'08]:

$$\text{Reg} = \hat{L} - L^{\star} = O\left(\sqrt{T \log(N!)}\right) = O\left(\sqrt{NT \log(N)}\right)$$

∃ Computationally efficient algorithms with $o(T)$ regret if either ℓ_t or S_t is i.i.d.
[Kanade, McMahan, Bryan' 09, Neu and Valko'14, Saha, Gaillard, Valko'20]

Computationally hard to get $o(T)$ regret when both ℓ_t and S_t are adversarial.
[Kanade and Steinke'11]
At least as hard as PAC learning DNF (disjunctive normal form) functions, for which no poly(N)-time algorithm is known.

Our Work

Motivation Is there a polynomial-time algorithm whose performance is comparable to the best ranking when both ℓ_t and S_t are adversarial?

→ Relaxing the regret definition:

$$\alpha\text{-Reg} \triangleq \hat{L} - \alpha L^\star \qquad \text{for some } \alpha \geq 1$$

α is called the "approximation ratio" ($\hat{L} \lesssim \alpha L^\star$ if α-Reg $= o(T)$)

Goals
- polynomial-time algorithm
- making α as small as possible
- α-Reg $= o(T)$

Result Overview

$$\hat{L} \leq \alpha L^\star + (\alpha\text{-Reg})$$

$K \triangleq$ an upper bound of $|S_t|$
$Z \triangleq$ # of zero-loss actions per round
$Z \leq K \leq N$

full-info: The learner observes $\ell_t(a)\ \forall a \in S_t$
bandit: The learner only observes $\ell_t(a_t)$

feedback	α	α-Reg	Requirement
full-info or bandit	N	N^2	
full-info	$O(\log K)$	$O(N^2)$	Binary loss with $Z = 1$
bandit	$O(\log K)$	$O\left(N\sqrt{KT} + N^2 K\right)$	Binary loss with $Z = 1$
full-info	$O(K^2)$	$O(N^4)$	Binary loss with $Z = 2$

In the following, we focus on the case of binary losses.

LEVEL algorithm (similar to [Blum, Mansour, Morgenstern'18])

$\text{level}(a) \leftarrow 0 \quad \forall a \in [N]$

For $t = 1, \dots, T$:

  Choose $a_t \in \text{argmin}_{a \in S_t} \text{ level}(a)$

  If $\ell_t(a_t) = 1$:

    $\text{level}(a_t) \leftarrow \text{level}(a_t) + 1$

Theorem. $\hat{L} \leq NL^\star + N^2$

HATT (Hedges Aggregated with Tournament Trees)

Setting full-information; in each round, exactly one $a \in S_t$ has loss 0 (others have loss 1)

We call this action the "winner" of that round

Idea Reducing the sleeping expert problem to pairwise comparison

Base Algorithm: Hedge for a 2-action problem

$$p_1(1) = p_1(2) = \frac{1}{2}$$

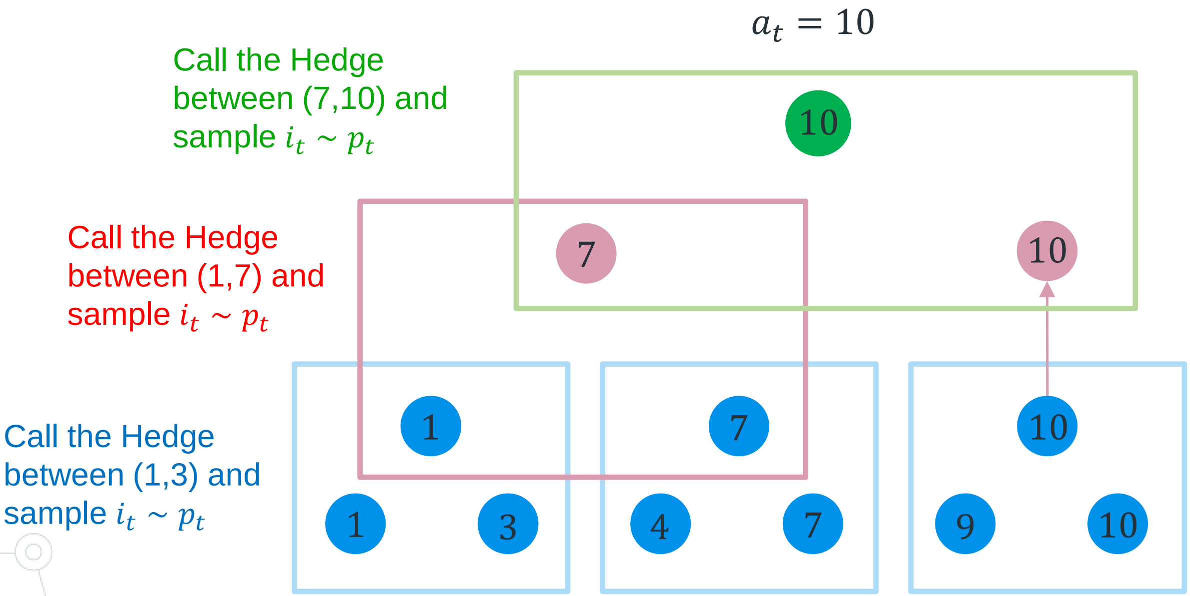
For $t = 1, 2, \dots, T$:

For $i \in \{1, 2\}$, update $p_{t+1}(i) \propto p_t(i)\, e^{-\eta \ell_t(i)}$

HATT (Hedges Aggregated with Tournament Trees)

Initiate a Hedge algorithm between any pair of actions (i, j). (totally $\binom{N}{2}$ of them)

In a round with $S_t = \{1, 3, 4, 7, 9, 10\}$



After receiving ℓ_t, update all hedges that <u>appears in the tree</u> and <u>involves the winner</u>.

HATT (Hedges Aggregated with Tournament Trees)

Theorem If in every round, there is exactly one $a \in [N]$ with $\ell_t(a) = 0$, and $|S_t| \le K$, then $\mathbb{E}\left[\hat{L}\right] \le O(\log K)L^\star + O\left(N^2\right)$.

Summary

- Our work gives first **efficient** algorithms that tackle **adversarial** sleeping experts / bandits.
- We propose four algorithms with **approximate regret bounds** for general or special cases.

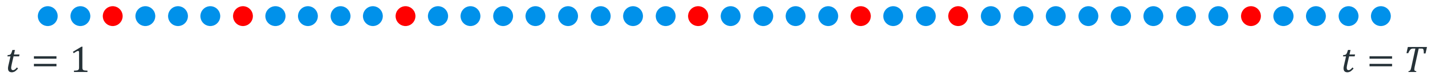
Future Work

- Any algorithm with $\mathrm{poly}(K)$ approximation ratio for $Z \geq 3$ cases.
- Further improving the approximation ratio in all cases.
- Providing approximation-ratio lower bounds under computational constraints.

Bandit-HATT

Theorem $\mathbb{E}[\hat{L}] \le O(\log K)L^\star + O\left(N\sqrt{KT} + N^2K\right)$

Idea Reducing bandit to full-information
(standard technique for bandit classification, e.g., Banditron, Newtron)



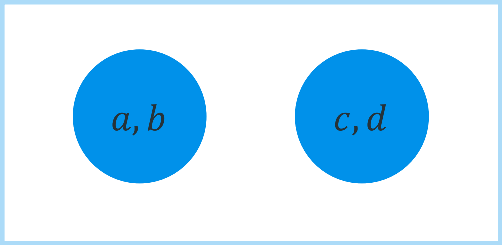
In time step marked with ●: perform uniform exploration → winner is revealed with probability $\ge \frac{1}{K}$
→ perform full-information update

Essentially becomes the full-info case

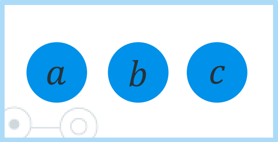
In time step marked with ●: choosing a_t using the tournament tree as before (but without update)

HOPP (Hedges Over Pairs of Pairs)

Hedge ((a, b), (c, d)) with distinct a, b, c, d



Hedge (a, b, c) with distinct a, b, c



Algorithm

Initialize $\binom{N}{4} \times 3 + \binom{N}{3}$ Hedges

For $t = 1, 2, \dots, T$:

  Use a complicated decision rule to aggregate the output of all Hedges.

  Observe $\ell_t(a) \ \ \forall a \in S_t$.

  Update some of the hedges.

Theorem If in every round, there is exactly two $a \in [N]$ with $\ell_t(a) = 0$ (others have losses of 1) and $|S_t| \le K$, then $\mathbb{E}[\hat{L}] \le O(K^2)L^\star + O(N^4)$.