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

Ehsan Emamjomeh-Zadeh EMAMJOME@USC.EDU
Chen-Yu Wei CHENYU.WEI@USC.EDU
Haipeng Luo HAIPENGL@USC.EDU
David Kempe DAVID.M.KEMPE@GMAIL.COM
University of Southern California

Editors: Vitaly Feldman, Katrina Ligett and Sivan Sabato

Abstract

We revisit the problem of online learning with sleeping experts/bandits: in each time step, only a subset of the actions are available for the algorithm to choose from (and learn about). The work of Kleinberg et al. (2010) showed that there exist no-regret algorithms which perform no worse than the *best ranking of actions* asymptotically. Unfortunately, achieving this regret bound appears computationally hard: Kanade and Steinke (2014) showed that achieving this no-regret performance is at least as hard as PAC-learning DNFs, a notoriously difficult problem.

In the present work, we relax the original problem and study computationally efficient *no-approximate-regret* algorithms: such algorithms may exceed the optimal cost by a multiplicative constant *in addition to* the additive regret. We give an algorithm that provides a no-approximate-regret guarantee for the general sleeping expert/bandit problems. For several canonical special cases of the problem, we give algorithms with significantly better approximation ratios; these algorithms also illustrate different techniques for achieving no-approximate-regret guarantees.

1. Introduction

Online learning with a fixed set of actions is a well-studied problem: the learner sequentially selects one of N actions and receives some feedback on the actions' losses. In the full-information setting (i.e., the *expert* problem (Freund and Schapire, 1997)), the feedback is the losses of all actions, while in the bandit setting (i.e., the multi-armed bandit problem (Auer et al., 2002)), the feedback is only the loss of the chosen action. In either case, the learner's goal is to minimize her regret over T rounds, defined as the difference between her total loss and the loss of the best fixed action. It is well-known that efficient algorithms exist with sublinear regret of order $\mathcal{O}(\sqrt{T})$ (known as "no-regret" algorithms).

In many situations, however, not every action is available every time. Take horse racing as an example, where each action corresponds to betting on a horse. While there is a fixed set of horses each season, only a small subset of them are competing in any one race; thus, only a subset of actions are available to choose from. Other examples include recommendation systems where products are not available at all times; this is particularly relevant for food (where each action corresponds to a restaurant or a special meal) or news (where each action corresponds to a category of news).

To capture these situations, a model called sleeping expert/bandit has been proposed (Freund et al., 1997), where in each round, only some actions are *awake* (i.e., available to be chosen and learned about), while the others are *asleep*. The standard regret measure no longer makes sense; in particular, there might not even be a fixed action which is awake all the time. Freund et al. (1997)

Table 1: Summary of main results. N is the total number of actions, K is an upper bound on the number of available actions at each round, T is the number of rounds, Z is the number of actions with zero loss at each round. Except for the first algorithm, all others can be implemented efficiently.

Algorithm	Approx. Ratio (α)	α-approx. Regret	Feedback	Constraint
Kleinberg et al. (2010)	1	$\mathcal{O}(\sqrt{NKT\log N})$	bandit	inefficient
Treat different $\mathcal{A}_t$ independently (Section 2)	1	$\sqrt{N^K KT}$	bandit	
HATT (Section 3.1)	$\mathcal{O}(\log K)$	$\mathcal{O}(N^2)$	full-info	$Z=1$
HOPP (Section 3.2)	$\mathcal{O}(K^2)$	$\mathcal{O}(N^4)$	full-info	$Z=2$
Bandit-HATT (Section 4)	$\mathcal{O}(\log K)$	$\mathcal{O}(N\sqrt{KT}+N^2K)$	bandit	$Z=1$
LEVEL (Section 4)	N	N^2	bandit	

proposed to measure regret against a particular action only for the rounds when this action is awake. As an alternative, Kleinberg et al. (2010) proposed to measure regret against the *best ranking* of the N actions, which naturally selects the available action with the highest ranking in each round. This latter performance measure is especially suited for applications such as horse racing or recommendation systems, and is the focus of our work.

In this setup, Kleinberg et al. (2010) proposed algorithms with optimal regret $\mathcal{O}(\sqrt{NT\log(N)})$ for full-information feedback and $\mathcal{O}(\sqrt{NKT\log(N)})$ for bandit feedback; here, K is an upper bound on the number of available actions in each round. Kleinberg et al. (2010) made no assumptions at all about how the available sets and actions' losses are chosen; i.e., their results hold in the adversarial setting. Unfortunately, their algorithms are computationally inefficient — they require maintaining information about all $N!$ rankings explicitly. On the other extreme, a trivial algorithm that treats each possible available subset independently achieves regret that is exponential in K.

The computational inefficiency of these algorithms is no accident. It was showed in (Kanade and Steinke, 2014) that achieving no-regret performance for this problem is at least as hard as PAC-learning DNFs, a notoriously difficult problem. Follow-up work thus focused on developing efficient no-regret algorithms under additional assumptions, such as imposing distributional assumptions (see **Related work** below).

In this paper, we take a different approach to get around the computational hardness: we still consider completely adversarial environments, but measure the learner's performance by α-*approximate regret* (for some approximation ratio $\alpha > 1$), which compares the learner's total loss to α times that of the best ranking. Such approximate regret measures have been studied in other online learning problems, such as (Garber, 2017; Roughgarden and Wang, 2018), but to our knowledge, our work is the first to consider them for the sleeping experts/bandits problem. Our most general algorithm is a simple and efficient algorithm with approximation ratio $\alpha = N$ and regret $\mathcal{O}(N^2)$ (independent of T), even under bandit feedback (Section 4).

We also consider two cases with special structures in losses and develop different algorithms with much better approximation ratios (see Table 1 for a summary). First, for the case when in each round, there is only one zero-loss action, we improve the approximation ratio to $\log(K)$, both

under full-information feedback (Section 3.1) and bandit feedback (Section 4; in the latter case, the regret becomes $\mathcal{O}(\sqrt{T})$). Note that the "one zero-loss action" structure is very common — in the horse racing example, there is only one winner in each race,[1] and in a multi-class classification problem, only one class is the correct one. Our algorithm is based on a novel way of aggregating several instances of the classic HEDGE algorithm (Freund and Schapire, 1997) over action pairs via a tournament.

One might wonder whether in this restricted setting, the aforementioned computational hardness still applies. Indeed, we generalize the argument of (Kanade and Steinke, 2014) and confirm that, even for this simple special case, obtaining no-regret algorithms is computationally hard (Theorem 6).

Next, we consider the case with two zero-loss actions in each round (e.g., betting on the winner or the runner-up has zero loss), and develop an algorithm in the full-information setting with approximation ratio $\mathcal{O}(K^2)$ and regret $\mathcal{O}(N^4)$ (Section 3.2). While the algorithm is also based on aggregating HEDGE instances, it is significantly more complex and requires hedging over *pairs of pairs* as well as *triples*. Our results shed light on how to deal with a small number of zero-cost actions, which is a common situation for machine learning problems with sparse rewards. Indeed, sparse rewards are studied in several recent works in the easier setting of fixed action sets (e.g., (Kwon and Perchet, 2016; Bubeck et al., 2018)).

Related work Several works propose efficient algorithms with exact regret (i.e., $\alpha = 1$) guarantees under additional assumptions. The original work of Kleinberg et al. (2010) considers a setting where the losses follow a fixed distribution, while Kanade et al. (2009), Neu and Valko (2014), and Saha et al. (2020) consider a setting where the action availability follows a fixed distribution. Hazan et al. (2012) study the case when $K = 2$ and achieve nearly optimal regret. Recently, Shayestehmanesh et al. (2019) studied a special case in which actions never wake up after falling asleep.

2. Problem Setting and Preliminaries

We consider the problem of online learning with a changing action set, also called the sleeping expert/bandit problem. Similar to the standard expert/bandit setting, the learner is faced with a set of actions $[N] = \{1, \dots, N\}$. However, in each round t, only a subset $\mathcal{A}_t \subseteq [N]$ is *available*, and the learner can only choose actions from $\mathcal{A}_t$ in that round. More precisely, the protocol is as follows. For each round $t = 1, \dots, T$, the adversary first chooses $\mathcal{A}_t \subseteq [N]$ and $\ell_t(a) \in \{0, 1\}$ for all $a \in \mathcal{A}_t$, with $\mathcal{A}_t$ revealed to the learner. Then, the learner chooses an action $a \in \mathcal{A}_t$, suffers loss $\ell_t(a_t)$, and receives some feedback. We consider two different settings with different feedback: 1) in the *full-information* setting, the feedback is $(\ell_t(a))_{a \in \mathcal{A}_t}$, i.e., the losses of all actions in $\mathcal{A}_t$; 2) in the *bandit* setting, the feedback is $\ell_t(a_t)$, i.e., the loss of the chosen action.

Both $\mathcal{A}_t$ and ℓ_t are decided by the adversary without any distributional assumptions. We assume that the losses are binary, i.e., $\ell_t(a) \in \{0, 1\}$. The goal of the learner is to be competitive with the best *ranking* of actions. A ranking σ specifies a total order on $[N]$, which is given by a bijection $m_\sigma : [N] \to [N]$, giving the position in the ranking for each element in $[N]$. Due to frequency of use in our paper, we reserve the letter σ itself for the mapping $\sigma : 2^{[N]} \setminus \emptyset \to [N]$ defined by $\sigma(\mathcal{S}) = \operatorname{argmin}_{x \in \mathcal{S}} m_\sigma(x)$. That is, $\sigma(\mathcal{S})$ is the highest-ranked element of $\mathcal{S}$, according to m_σ. We write $\sigma(\{i, j\})$ or $\sigma(\{i, j, k\})$ as $\sigma(i, j)$ or $\sigma(i, j, k)$ for simplicity.

1. In this example, in addition to the loss of betting on each horse, the bettor observes the ranking of each race as well. However, in the adversarial setting that we consider, this extra information is not useful since the ranking can be arbitrary from round to round.

For a fixed ranking σ, we define its choice at time t as its highest-ranked action among $\mathcal{A}_t$ — using our notation, this can be written as $\sigma(\mathcal{A}_t)$. One standard way to measure the performance of the learner is to compare her total loss with that of the best ranking, formally defined as the *regret*: $\mathrm{Reg}_T = \sum_{t=1}^T \ell_t(a_t) - L^*$, where $L^* = \min_\sigma \sum_{t=1}^T \ell_t(\sigma(\mathcal{A}_t))$ is the total loss of the best ranking. An algorithm with regret sublinear in T performs almost as well as the best ranking in the long run.

Unfortunately, it was shown that achieving sublinear regret is computationally at least as hard as PAC-learning DNFs, for which no polynomial-time (in N) algorithm is known (Kanade and Steinke, 2014). Therefore, we pursue the relaxed goal of providing polynomial-time algorithms that guarantee sub-linear α-*approximate* regret, defined as follows: $\mathrm{Reg}_T^\alpha = \sum_{t=1}^T \ell_t(a_t) - \alpha L^*$. Phrased in another way, our results can all be written as $\sum_{t=1}^T \ell_t(a_t) \le \alpha L^* + \beta(T)$ for some $\beta(T)$ which grows sub-linearly in T; our goal is to make α and $\beta(T)$ as small as possible.

Some of our guarantees depend on the (largest) cardinality of the sets $\mathcal{A}_t$ of available actions, denoted by K. Note that achieving $\alpha = 1$ and $\beta(T) = \mathcal{O}\left(\sqrt{\left(\sum_{i=1}^K \binom{N}{i}\right) KT}\right) = \mathcal{O}(\sqrt{N^K KT})$ efficiently is trivial. One simply treats each possible $\mathcal{A}_t$ as an independent problem with a fixed action set and runs a separate standard bandit algorithm with regret $\mathcal{O}(\sqrt{KT})$, then combines all regret bounds with a Cauchy-Schwarz inequality.[2] In contrast, our bounds are all polynomial in N and K.

Notation. We use $\mathbb{1}[\mathcal{E}]$ as an indicator function, which is 1 if the event $\mathcal{E}$ is true and 0 otherwise. We use $\Delta_\mathcal{S}$ to denote the probability simplex over a set $\mathcal{S}$, i.e., $\Delta_\mathcal{S} = \left\{\boldsymbol{p} : \mathcal{S} \to [0,1] \;\middle|\; \sum_{a\in\mathcal{S}} p(a) = 1\right\}$.

2.1. Preliminaries: The Hedge Algorithm

Most of our algorithms are based on the classic HEDGE algorithm for the expert learning problem (Freund and Schapire, 1997), which we review below. The setting of the expert learning problem is the same as our problem with full-information feedback, except the action set $\mathcal{A}_t = \mathcal{S}$ is fixed throughout. The HEDGE algorithm is given as Algorithm 1 — we use $\mathcal{S}$ instead of $\mathcal{A}$ for its (fixed) action set and $[0, R]$ for the loss range, because we will later invoke it with different choices of $\mathcal{S}$ and R.

The performance guarantee of the HEDGE algorithm is captured by, e.g., Theorem 2.4 of Cesa-Bianchi and Lugosi (2006). The following lemma slightly extends their result for general values of R, which will be needed in the analysis in Section 4.

Lemma 1 *Algorithm 1 ensures:* $\mathbb{E}\left[\sum_{t=1}^T \ell_t(a_t)\right] \le \frac{\eta R}{1-\mathrm{e}^{-\eta R}} \cdot \mathbb{E}\left[\min_{a^*\in\mathcal{S}} \sum_{t=1}^T \ell_t(a^*)\right] + \frac{R \ln |\mathcal{S}|}{1-\mathrm{e}^{-\eta R}}$.

Proof Let $w_1(a) = 1$ for all $a \in \mathcal{S}$ and define $w_{t+1}(a) = w_t(a)\mathrm{e}^{-\eta \ell_t(a)}$. Also, define $W_t = \sum_{a\in\mathcal{S}} w_t(a)$. Then clearly, $p_t(a) \propto w_t(a)$ and $p_t(a) = w_t(a)/W_t$. Using these definitions, we have

$$\ln \frac{W_{T+1}}{W_1} = \sum_{t=1}^T \ln \frac{W_{t+1}}{W_t} = \sum_{t=1}^T \ln \frac{\sum_{a\in\mathcal{S}} w_t(a)\mathrm{e}^{-\eta\ell_t(a)}}{W_t} = \sum_{t=1}^T \ln \left(\sum_{a\in\mathcal{S}} p_t(a)\mathrm{e}^{-\eta\ell_t(a)}\right).$$

2. For more details, see (Abernethy, 2010, Lemma 3) for the case with full-information and $K = 2$.

Algorithm 1 HEDGE (parameter: η)

Input: $\mathcal{S}$
forall $a \in \mathcal{S}$ **do** $p_1(a) = \frac{1}{|\mathcal{S}|}$.
for $t = 1, 2, \ldots, T$ **do**
  Sample $a_t \sim \boldsymbol{p}_t$.
  Receive $\ell_t(a) \in [0, R]$ for all $a \in \mathcal{S}$.
  Let $\boldsymbol{p}_{t+1} = \mathrm{EWU}(\boldsymbol{p}_t, \boldsymbol{\ell}_t)$.
end

Algorithm 2 EWU (Exponential Weight Update)

Input: $\boldsymbol{p}_t \in \Delta_{\mathcal{S}}, \boldsymbol{\ell}_t \in [0, R]^{\mathcal{S}}$.
Parameter: $\eta > 0$.
forall $a \in \mathcal{S}$ **do** $p_{t+1}(a) = \frac{p_t(a)\mathrm{e}^{-\eta \ell_t(a)}}{\sum_{a' \in \mathcal{S}} p_t(a')\mathrm{e}^{-\eta \ell_t(a')}}$.
return $\boldsymbol{p}_{t+1}$.

Since $\ell_t(a) \in [0, R]$ and $\mathrm{e}^{-\eta R x}$ is convex in x, we have

$$\mathrm{e}^{-\eta \ell_t(a)} \leq \frac{\ell_t(a)}{R}\mathrm{e}^{-\eta R} + \left(1 - \frac{\ell_t(a)}{R}\right) = 1 - \frac{1 - \mathrm{e}^{-\eta R}}{R}\ell_t(a).$$

Thus,

$$\ln\left(\sum_{a \in \mathcal{S}} p_t(a)\mathrm{e}^{-\eta \ell_t(a)}\right) \leq \ln\left(1 - \frac{1 - \mathrm{e}^{-\eta R}}{R}\sum_{a \in \mathcal{S}} p_t(a)\ell_t(a)\right) \leq -\frac{1 - \mathrm{e}^{-\eta R}}{R}\sum_{a \in \mathcal{S}} p_t(a)\ell_t(a),$$

because $\ln(1 - x) \leq -x$ for $x \geq 0$. On the other hand, for any $a^* \in \mathcal{S}$,

$$\ln \frac{W_{T+1}}{W_1} \geq \ln \frac{w_{T+1}(a^*)}{W_1} \geq \ln \frac{\mathrm{e}^{-\eta \sum_{t=1}^{T} \ell_t(a^*)}}{|\mathcal{S}|} = -\eta \sum_{t=1}^{T} \ell_t(a^*) - \ln |\mathcal{S}|.$$

Combining both inequalities, we get

$$\sum_{t=1}^{T}\sum_{a \in \mathcal{S}} p_t(a)\ell_t(a) \leq \frac{\eta R}{1 - \mathrm{e}^{-\eta R}} \min_{a^* \in \mathcal{S}} \sum_{t=1}^{T} \ell_t(a^*) + \frac{R \ln |\mathcal{S}|}{1 - \mathrm{e}^{-\eta R}}.$$

Taking expectation on both sides finishes the proof. ■

3. The Full-information Setting

In this section, we consider two special cases in the full-information setting; we obtain approximate regret bounds whose approximation ratio depends only on K, the maximum cardinality of $\mathcal{A}_t$. These

two special cases are the following: 1) in each round t, exactly one action has loss 0, i.e., for all t, $\sum_{a\in\mathcal{A}_t} \mathbb{1}[\ell_t(a)=0]=1$, and 2) in each round t, exactly two actions have loss 0, i.e., for all t, $\sum_{a\in\mathcal{A}_t} \mathbb{1}[\ell_t(a)=0]=2$. We remark again that these structures correspond to problems with sparse rewards, studied in previous work as well (Kwon and Perchet, 2016; Bubeck et al., 2018).

The first case is reminiscent of multi-class classification with 0-1 loss: there is only one "label" that is correct and incurs zero loss; other labels all incur a loss of one. In a typical classification problem, the learner uses *features* as side information to infer labels; in our problem, we may view the available action set $\mathcal{A}_t$ as the side information. For this case, in Section 3.1, we give an algorithm called HATT (Hedges Aggregated with Tournament Trees) which guarantees that the total loss of the learner is upper-bounded by $\mathcal{O}(\log_2 K)L^* + \mathcal{O}(N^2)$.

For the second case, in Section 3.2, we design another (more involved) algorithm called HOPP (Hedges Over Pairs of Pairs) whose loss is upper-bounded by $\mathcal{O}(K^2)L^* + \mathcal{O}(N^4)$. Note that we get a worse approximation ratio in this case compared to the first case.

When the number of possible zero-loss actions exceeds 2, it is not clear how to efficiently obtain an approximate regret bound where α is a function of K and $\beta(T)$ is polynomial in K. However, an approximation ratio of N is still achievable, even in the bandit setting, as shown in Section 4.

The algorithms in Sections 3.1 and 3.2 are based on similar ideas. They maintain several sub-algorithms, each dealing with a constant-size sub-problem (e.g., a 2-expert algorithm that compares the performance of actions i, j in the rounds when they are both available). Then, when given $\mathcal{A}_t$, a meta-algorithm aggregates the recommendations of these sub-algorithms and generates the final $a_t \in \mathcal{A}_t$. The design of the sub-problems and their losses has the following two key properties:

Property 1 *Whenever the learner makes a mistake (i.e., $\ell_t(a_t) = 1$), there is at least one sub-algorithm which also makes a mistake in its sub-problem.*

Property 2 *Whenever the best ranking σ makes no mistake (i.e., $\ell_t(\sigma(\mathcal{A}_t)) = 0$), it also makes no mistake for all of the defined sub-problems.*

These two properties are sufficient to ensure that algorithms with sub-linear regret for the sub-problems also guarantee good approximate regret bounds for the original problem.

3.1. The HATT Algorithm for One Zero-Loss Action

We begin with an algorithm for the case of a single zero-loss action per round. Recall that the sleeping experts algorithm by Kleinberg et al. (2010) is based on the idea of "hedging over all rankings" — that is, viewing each ranking of actions as an "expert" in HEDGE. This leads to (exact) regret bounds with respect to the best ranking, but requires keeping track of $N!$ experts in total. Instead of keeping track of an expert for each permutation, our algorithm only maintains one expert for each pair of actions. This results in a coarser representation, but we show that it still achieves good guarantees. In other words, while Kleinberg et al. (2010) maintains one algorithm that learns over exponentially many experts, we maintain $\binom{N}{2}$ HEDGE algorithms, each learning over *two* actions. Then, a meta algorithm combines the recommendations of all 2-expert HEDGE algorithms and decides on the final action the learner should choose.

To learn the preference between the pair of actions $\{i, j\} \subset [N]$ with $i \neq j$, HATT simply runs an instance $\mathcal{H}_{i,j}$ of HEDGE (Algorithm 1) with $\mathcal{S} = \{i, j\}$. HATT then uses the following *tournament* approach as the meta algorithm to combine the recommendations of all HEDGE algorithms. In each

Algorithm 3 HATT (Hedges Aggregated with Tournament Trees)

forall $i < j$ **do** set $p_1^{i,j}(i) = p_1^{i,j}(j) = \frac{1}{2}$.
for $t = 1, \ldots, T$ **do**
  Receive $\mathcal{A}_t$ and let $(a_t, U_t) = \text{TOURNAMENT}(\mathcal{A}_t, (p_t^{i,j})_{i,j})$.
  Choose a_t and suffer loss $\ell_t(a_t)$.
  Learn $\ell_t(a)$ for all $a \in \mathcal{A}_t$ and let z_t be such that $\ell_t(z_t) = 0$.
  forall i *with* $\{i, z_t\} \in U_t$ **do** $c_t^{i,z_t}(i) = 1,\ c_t^{i,z_t}(z_t) = 0,\ p_{t+1}^{i,z_t} = \text{EWU}\left(p_t^{i,z_t}, c_t^{i,z_t}\right)$.
  forall *other* $i < j$ **do** let $c_t^{i,j}(\cdot) = 0$ and $p_{t+1}^{i,j} = p_t^{i,j}$.
end

Algorithm 4 TOURNAMENT

Input: $\mathcal{A}_t$: available action set at time t
$P_t = (p_t^{i,j})_{i,j}$: distributions of hedges over all pairs $\{i, j\}$
Initialization: $U_t = \emptyset$.
forall $i < j$ **do** sample $a_t^{i,j} \sim p_t^{i,j}$.
Let $\mathcal{T}$ be a balanced binary tree with exactly $|\mathcal{A}_t|$ leaves, each mapped to a distinct action in $\mathcal{A}_t$.
foreach *leaf* v **do** let winner(v) be the action v is mapped to.
foreach *internal node* v*, in bottom-up order* **do**
  if v *has one child* v' **then** set winner(v) = winner(v').
  else let i, j be the winners at the two children of v; set winner$(v) = a_t^{i,j}$, and add $\{i, j\}$ to U_t.
end
return *winner*(*root of* $\mathcal{T}$), U_t.

round t, HATT creates a single-elimination tournament tree $\mathcal{T}_t$ with $|\mathcal{A}_t|$ leaves, and thus depth $1 + \lceil \log_2(|\mathcal{A}_t|) \rceil$. It assigns each element in $\mathcal{A}_t$ to one leaf of $\mathcal{T}_t$ (arbitrarily). Then the actions perform a single-elimination tournament following $\mathcal{T}_t$ to generate the final winner a_t. For each pair of actions (i, j), the winner and loser are determined by the HEDGE algorithm $\mathcal{H}_{i,j}$. Notice that each action is involved in at most $\log_2 K$ comparisons in each round. We will show that this is the regret approximation ratio of HATT.

More formally, in Algorithm 3, $p_t^{i,j}$ denotes the p_t maintained by the HEDGE instance $\mathcal{H}_{i,j}$; we use $p_t^{i,j}(i)$ and $p_t^{i,j}(j)$ to denote the probabilities for the actions i and j, respectively. Note that $p_t^{i,j}$ is shorthand for $p_t^{\{i,j\}}$, so $p_t^{i,j}$ and $p_t^{j,i}$ are always the same, and we only run one instance of HEDGE for each pair $\{i, j\}$ (similarly for the notation $c_t^{i,j}$ and $a_t^{i,j}$ below). In Algorithm 4, each HEDGE instance $\mathcal{H}_{i,j}$ samples a winner $a_t^{i,j}$ according to $p_t^{i,j}$, and a tournament is run. In this process, a set U_t is used to record all pairs involved in the tournament.

After choosing the final winner a_t of the tournament, HATT receives the loss feedback. We let z_t denote the unique zero-loss action; hence, for all $a \in \mathcal{A}_t \setminus \{z_t\}$, the loss is $\ell_t(a) = 1$. Then, for all pairs in U_t that involve z_t, the algorithm updates the corresponding HEDGE instance with the natural loss vector: action z_t has loss 0, and the other action has loss 1. For all other pairs $\{i, j\}$, the

algorithm does not make any updates, although for notational convenience in the analysis, we still define a loss vector $c_t^{i,j}$ to be the all-zero vector, so that $p_{t+1}^{i,j} = p_t^{i,j} = \mathrm{EWU}\left(p_t^{i,j}, c_t^{i,j}\right)$ holds.

The performance of HATT is summarized in the following theorem:

Theorem 2 HATT *(Algorithm 3) guarantees that*

$$\mathbb{E}\left[\sum_{t=1}^{T} \ell_t(a_t)\right] \leq \frac{\eta(1+\lceil \log_2(K)\rceil)}{1-\mathrm{e}^{-\eta}} \mathbb{E}\left[\min_{\sigma} \sum_{t=1}^{T} \ell_t(\sigma(\mathcal{A}_t))\right] + \binom{N}{2} \cdot \frac{\ln 2}{(1-\mathrm{e}^{-\eta})}.$$

In particular, when $\eta = 1$, the above is no more than $\mathcal{O}(\log_2(K)) \cdot \mathbb{E}\left[\sum_{t=1}^{T} \ell_t(\sigma(\mathcal{A}_t))\right] + \mathcal{O}\left(N^2\right)$.

Note that the approximation ratio is only logarithmic in K, and the additive regret term is also independent of T. The proof of Theorem 2 can be obtained by directly combining the following three lemmas ($a_t^{i,j}$ and $c_t^{i,j}$ are as defined in Algorithm 4 and Algorithm 3, respectively). Lemmas 3 and 5 assert that HATT ensures Properties 1 and 2, respectively.

Lemma 3 *In Algorithm 3, whenever the learner makes a mistake (i.e., $\ell_t(a_t) = 1$), there must be a* HEDGE *algorithm which also makes a mistake. More formally, for every t,*

$$\ell_t(a_t) \leq \sum_{i<j} c_t^{i,j}(a_t^{i,j}).$$

Proof If $\ell_t(a_t) = 0$, then the inequality clearly holds. If $\ell_t(a_t) = 1$, by the tournament approach, there must exist an $i \neq z_t$ with $\{i, z_t\} \in U_t$ and $a_t^{i,z_t} = i$. Thus we have

$$c_t^{i,z_t}(a_t^{i,z_t}) = c_t^{i,z_t}(i) = \ell_t(i)\mathbb{1}[\{i, z_t\} \in U_t] = 1.$$

Thus the inequality also holds when $\ell_t(a_t) = 1$. ■

Lemma 4 *Algorithm 3 guarantees that for all $i < j$,*

$$\mathbb{E}\left[\sum_{t=1}^{T} c_t^{i,j}(a_t^{i,j})\right] \leq \frac{\eta}{1-\mathrm{e}^{-\eta}} \mathbb{E}\left[\min_{\sigma} \sum_{t=1}^{T} c_t^{i,j}(\sigma(i,j))\right] + \frac{\ln 2}{1-\mathrm{e}^{-\eta}}.$$

Proof Note that importantly, the value of $c_t^{i,j}$ is decided independently of $a_t^{i,j}$ (although it could depend on other $a_t^{i',j'}$). We can therefore apply Lemma 1 with $R = 1$ and $\mathcal{S} = \{i, j\}$, which proves the lemma. ■

Lemma 5 *Algorithm 3 guarantees that for all rankings σ,*

$$\sum_{i<j} c_t^{i,j}(\sigma(i,j)) \leq (1+\lceil \log_2(K)\rceil) \cdot \ell_t(\sigma(\mathcal{A}_t)).$$

Proof If $\ell_t(\sigma(\mathcal{A}_t)) = 0$, then $\sigma(\mathcal{A}_t) = z_t$ (i.e., σ ranks z_t first among $\mathcal{A}_t$), and thus $\sigma(i, z_t) = z_t$ for all $i \in \mathcal{A}_t$. Therefore,

$$\sum_{i<j} c_t^{i,j}(\sigma(i,j)) = \sum_{i\in\mathcal{A}_t, i\neq z_t} c_t^{i,z_t}(\sigma(i,z_t)) = \sum_{i\in\mathcal{A}_t, i\neq z_t} c_t^{i,z_t}(z_t) = 0.$$

If $\ell_t(\sigma(\mathcal{A}_t)) = 1$, then

$$\sum_{i<j} c_t^{i,j}(\sigma(i,j)) \leq \sum_{i\in\mathcal{A}_t i\neq z_t} \mathbb{1}[\{i, z_t\} \in U_t] \leq 1 + \lceil \log_2(K) \rceil.$$

In both cases,

$$\sum_{i<j} c_t^{i,j}(\sigma(i,j)) \leq (1 + \lceil \log_2(K) \rceil) \cdot \ell_t(\sigma(\mathcal{A}_t)).$$

■

We are now ready to prove the theorem.

Proof [of Theorem 2] We apply Lemmas 3, 4, 5 successively:

$$\begin{aligned}
\mathbb{E}\left[\sum_{t=1}^{T} \ell_t(a_t)\right] &\leq \mathbb{E}\left[\sum_{i<j}\sum_{t=1}^{T} c_t^{i,j}(a_t^{i,j})\right] \\
&\leq \frac{\eta}{1-\mathrm{e}^{-\eta}} \mathbb{E}\left[\min_{\sigma} \sum_{i<j}\sum_{t=1}^{T} c_t^{i,j}(\sigma(i,j))\right] + \sum_{i<j} \frac{\ln 2}{1-\mathrm{e}^{-\eta}} \\
&\leq \frac{\eta(1+\lceil \log_2(K) \rceil)}{1-\mathrm{e}^{-\eta}} \mathbb{E}\left[\min_{\sigma} \sum_{t=1}^{T} \ell_t(\sigma(\mathcal{A}_t))\right] + \binom{N}{2} \cdot \frac{\ln 2}{1-\mathrm{e}^{-\eta}}.
\end{aligned}$$

This completes the proof. ■

Finally, we point out that even in this simple case with one zero-loss action, achieving no-regret performance (i.e. $\alpha = 1$) is still as hard as PAC-learning DNFs, as shown below.

Theorem 6 *If there exists a computationally efficient no-regret algorithm for the sub-class of sleeping expert problems which always have exactly one zero-loss action, then there exists a computationally efficient algorithm for PAC-learning DNFs under arbitrary distributions.*

We do not have a better lower bound on the approximation ratio for polynomial-time algorithms; these kinds of computation-constrained lower bounds are scarce in the literature. However, we note that, together with (Awasthi et al., 2010), our proof of Theorem 6 implies that achieving an approximation ratio better than $\mathcal{O}(K^{1/3})$ in the general case would improve the state-of-the-art for agnostically learning disjunctions with polynomial-time algorithms.

Proof Our hardness proof is heavily based on the hardness result in Kanade and Steinke (2014). They reduce from PAC-learning of DNFs to *agnostic learning of disjunctions*, and from that problem to achieving no-regret performance with high probability against the best ranking in sleeping expert problems.

The key observation is that the instances of the sleeping expert problem produced by the reduction in Kanade and Steinke (2014) are already almost of the restricted form of Theorem 6: (1) the set of available actions always satisfies $|\mathcal{A}_t| = K$, (2) the losses $\ell_t(a) \in \{0, 1\}$ are always binary, and (3) the loss vector $\boldsymbol{\ell}_t$ always has exactly one 0 or exactly one 1. Only the third property is different from our model of exactly one 0. Our proof therefore provides a reduction from their instances to ours.

Let $\mathcal{E}_t^0 = [\sum_{a \in \mathcal{A}_t} \mathbb{1}[\ell_t(a) = 0] = 1]$ be the event that the loss vector in round t has exactly one zero, and $\mathcal{E}_t^1 = [\sum_{a \in \mathcal{A}_t} \mathbb{1}[\ell_t(a) = 1] = 1]$ the event that the loss vector in round t has exactly one one.

Assume that there is an algorithm $\mathcal{Z}_0$ which always achieves no regret for instances in which all loss vectors have exactly one zero. That is, for any binary-loss sequence $\boldsymbol{\ell}_t$ that satisfies $\mathcal{E}_t^0$ for all t, the algorithm $\mathcal{Z}_0$ outputs $a_1, \ldots, a_T$ such that for all σ,

$$\mathbb{E}\left[\sum_{t=1}^{T} \ell_t(a_t) - \sum_{t=1}^{T} \ell_t(\sigma(\mathcal{A}_t))\right] = o(T).$$

We will give a reduction showing how to leverage $\mathcal{Z}_0$ to obtain an algorithm $\mathcal{Z}_{01}$ which achieves the same no-regret guarantee for instances in which all loss vectors have exactly one zero or exactly one one. The algorithm $\mathcal{Z}_{01}$ works as follows.

- Upon receiving the available action set $\mathcal{A}_t$, $\mathcal{Z}_{01}$ passes $\mathcal{A}_t$ to $\mathcal{Z}_0$, and chooses the action $a_t \in \mathcal{A}_t$ returned by $\mathcal{Z}_0$.
- The algorithm observes losses $\ell'_t(a)$ for all $a \in \mathcal{A}_t$, and can determine which of $\mathcal{E}_t^0, \mathcal{E}_t^1$ holds.
  - If $\mathcal{E}_t^0$ holds, then with probability $\frac{1}{K-1}$, $\mathcal{Z}_{01}$ sets $\boldsymbol{\ell}_t$ to be $\boldsymbol{\ell}'_t$; with the remaining probability $\frac{K-2}{K-1}$, it uniformly randomly draws z_t from $\mathcal{A}_t$, sets $\ell_t(z_t) = 0$, and $\ell_t(a) = 1$ for all $a \in \mathcal{A}_t \setminus \{z_t\}$.
  - If $\mathcal{E}_t^1$ holds, then $\mathcal{Z}_{01}$ uniformly randomly draws z_t from the $(K-1)$ zero-loss actions. It sets $\ell_t(z_t) = 0$ and $\ell_t(a) = 1$ for all $a \in \mathcal{A}_t \setminus \{z_t\}$.
- $\mathcal{Z}_{01}$ then passes the loss vector $\boldsymbol{\ell}_t$ to $\mathcal{Z}_0$.

The loss vectors $\boldsymbol{\ell}_t$ always have exactly one zero entry. The expected losses are as follows:

- Conditioned on $\mathcal{E}_t^0$, we have $\mathbb{E}[\ell_t(a)] = \frac{1}{K-1} \cdot \ell'_t(a) + \frac{K-2}{K-1} \cdot \frac{K-1}{K} = \frac{K-2}{K} + \frac{\ell'_t(a)}{K-1}$.
- Conditioned on $\mathcal{E}_t^1$, we have $\mathbb{E}[\ell_t(a)] = \frac{K-2}{K-1} \cdot \mathbb{1}[\ell'_t(a) = 0] + 1 \cdot \mathbb{1}[\ell'_t(a) = 1] = \frac{K-2}{K-1} + \frac{\ell'_t(a)}{K-1}$.

Therefore,

$$\begin{aligned}
&\frac{1}{K-1}\cdot\mathbb{E}\left[\sum_{t=1}^{T}\ell'_t(a_t)\right]-\frac{1}{K-1}\cdot\sum_{t=1}^{T}\ell'_t(\sigma(\mathcal{A}_t))\\
&=\mathbb{E}\left[\sum_{t=1}^{T}\ell_t(a_t)-\frac{(K-2)\cdot\mathbb{1}[\mathcal{E}_t^0]}{K}-\frac{(K-2)\cdot\mathbb{1}[\mathcal{E}_t^1]}{K-1}\right]\\
&\quad-\mathbb{E}\left[\sum_{t=1}^{T}\ell_t(\sigma(\mathcal{A}_t))-\frac{(K-2)\cdot\mathbb{1}[\mathcal{E}_t^0]}{K}-\frac{(K-2)\cdot\mathbb{1}[\mathcal{E}_t^1]}{K-1}\right]\\
&=\mathbb{E}\left[\sum_{t=1}^{T}\ell_t(a_t)\right]-\mathbb{E}\left[\sum_{t=1}^{T}\ell_t(\sigma(\mathcal{A}_t))\right]=o(T),
\end{aligned}$$

where the last line is guaranteed by our assumption that $\mathcal{Z}_0$ is no-regret. Multiplying by $K-1$, we also obtain that

$$\mathbb{E}\left[\sum_{t=1}^{T}\ell'_t(a_t)-\ell'_t(\sigma(\mathcal{A}_t))\right]=o(T).$$

To finish the reduction from the case of Kanade and Steinke (2014) to our case, we need to further argue that the algorithm $\mathcal{Z}_{01}$ with sublinear expected regret can be transformed into an algorithm that has sublinear regret with high probability.

To see this, one simply runs T copies of $\mathcal{Z}_{01}$ simultaneously and aggregates them via Hedge to decide the final output. By Hoeffding's inequality, with probability at least $1-\delta$, one of the T copies must have regret smaller than its expectation plus $\mathcal{O}(\sqrt{T\ln(1/\delta)})$ (since the range of regret is $[-T,T]$). Also note that Hedge itself has $\mathcal{O}(\sqrt{T\ln(T/\delta)})$ regret against any one of the copies with probability $1-\delta$. Combining these two statements, we have thus constructed a new algorithm which has sublinear regret with high probability. This completes the proof. ■

3.2. The HOPP Algorithm for Two Zero-Loss Actions

The case of two zero-loss actions is significantly more complicated. Again, we want to design sub-problems with Properties 1 and 2. To achieve these properties, it is now not sufficient any more to define sub-problems comparing only two actions, as we did in Section 3.1. This is because it is now possible that a ranking makes no mistake ($\ell_t(\sigma(\mathcal{A}_t))=0$), while making mistakes in some pairwise comparisons ($\ell_t(\sigma(i,j))=1$). For example, consider the case when the first, second, and third actions according to the ranking σ have losses $0,1,0$, respectively. Then σ does not make a mistake in this round because its top choice receives zero loss. However, in the sub-problem that compares the second and third actions, σ does make a mistake because its choice among the two actions incurs a loss of 1. This would violate Property 2.

To address the above issue, we design sub-problems as "comparing two pairs of actions," as well as "choosing among three actions." The hedges for triples of actions are standard. For each set $S\subseteq\mathcal{A}$ with $|S|=3$, there is a separate HEDGE that recommends one of the three actions in S. This instance is updated only when $S\subseteq\mathcal{A}_t$ turns out to contain both zero-loss actions, in which case the loss vector is the natural one following ℓ_t. See the last part of Algorithm 5.

The subproblems for pairs of actions are more intricate and non-standard, and we next explain them in detail. Each such sub-problem compares a pair $X = \{i, j\}$ of actions with another pair $Y = \{k, l\}$, where i, j, k, l are all distinct. The algorithm HOPP uses a separate 2-expert HEDGE $\mathcal{H}_{X,Y}$ to learn each such sub-problem (X, Y). This instance is only updated when both X and Y are in $\mathcal{A}_t$. In this case, only when one of X or Y consists of both of the two zero-loss actions do we assign positive loss to the other pair. More precisely, if $\ell_t(i) = \ell_t(j) = 0$, then choosing Y in this sub-problem incurs a loss of 1; similarly, if $\ell_t(k) = \ell_t(l) = 0$, then choosing X incurs a loss of 1. In all other cases, we define both actions' losses as 0. We also define the *choice* of a ranking σ for this sub-problem as follows: if $\sigma(i, j, k, l) \in X$, then the choice of σ is X; otherwise, it is Y. This way, when a ranking σ makes no mistake in the original problem ($\ell_t(\sigma(\mathcal{A}_t)) = 0$), it also has zero loss in all sub-problems. This ensures that Property 2 holds. (The preceding arguments are formalized in Lemma 11.)

To make Property 1 also hold, we design complex rules for aggregating the recommendations of all hedges so that every time the learner suffers loss 1 in the original problem, it must also suffer positive loss in some sub-problem. For this purpose, we define *good pairs* in the sub-algorithm SELECTIONRULE (Algorithm 6). A good pair $X \subseteq \mathcal{A}_t$ is a pair such that for all disjoint pairs $Y \subseteq \mathcal{A}_t$, the hedge $\mathcal{H}_{X,Y}$ chooses X as the winner. It is possible that no pair is good, or that more than one pair is good. For each possibility, we discuss how to choose the final a_t (see Algorithm 6). The following lemma shows that Algorithm 6 indeed considers all cases.

Lemma 7 *For the good pairs defined above, the following hold: 1) Any two good pairs must have one common action; 2) Either all good pairs have one common action, or there are exactly three good pairs, and they are of the form $\{i, j\}, \{j, k\}, \{k, i\}$.*

Proof If X, Y were disjoint, then for X to be good, $\mathcal{H}_{X,Y}$ has to choose X, but for Y to be good, $\mathcal{H}_{X,Y}$ has to choose Y. So X, Y must intersect. This also directly implies the second statement. ■

The case when there are exactly three good pairs of the form $\{i, j\}, \{j, k\}, \{k, i\}$ is the only case in which the algorithm needs to also consult the hedges over triples. The approximate regret guarantee of HOPP is given by the following theorem.

Theorem 8 HOPP *ensures:* $\mathbb{E}\left[\sum_{t=1}^{T} \ell_t(a_t)\right] \leq \mathcal{O}(K^2)\mathbb{E}\left[\min_\sigma \sum_{t=1}^{T} \ell_t(\sigma(\mathcal{A}_t))\right] + \mathcal{O}(N^4)$.

Note that the approximation ratio $\mathcal{O}(K^2)$ is significantly worse than the case with one zero-loss action, but is still only a function of K (and not N). The additive regret term is also worse, but still independent of T. To prove Theorem 8, we make use the following three lemmas (the notation in the lemmas is defined in Algorithms 5 and 6). Again, Lemmas 9 and 11 assert that HOPP satisfies Properties 1 and 2.

Lemma 9 HOPP *guarantees that*

$$\ell_t(a_t) \leq \sum_{X,Y \text{ disjoint}} c_t^{X,Y}(A_t^{X,Y}) + \sum_{S:|S|=3} d_t^S(b_t^S).$$

Proof If $\ell_t(a_t) = 0$, then the inequality clearly holds. Therefore, we only need to consider the case $\ell_t(a_t) = 1$.

First, for all the cases except when there are exactly three good pairs of the form $\{i, j\}, \{j, k\}, \{k, i\}$, we prove that the pair Z_t of zero-loss actions cannot be good:

Algorithm 5 HOPP (Hedges Over Pairs of Pairs)

forall *pairs* X, Y *with* $X \cap Y = \emptyset$ **do** set $p_1^{X,Y}(X) = p_1^{X,Y}(Y) = \frac{1}{2}$.
forall *triples* $S = \{i, j, k\}$ *of actions* **do** set $q_1^S(i) = q_1^S(j) = q_1^S(k) = \frac{1}{3}$.
for $t = 1, \ldots, T$ **do**
  Receive $\mathcal{A}_t$ and let $a_t = \text{SELECTIONRULE}\left(\mathcal{A}_t, (p_t^{X,Y})_{X,Y}, (q_t^S)_S\right)$.
  Choose a_t, suffer loss $\ell_t(a_t)$, and learn $\ell_t(a)$ for all $a \in \mathcal{A}_t$.
  Let $Z_t = \{a : \ell_t(a) = 0\}$ be the pair of actions with zero loss.
  forall *disjoint pairs* X, Y **do**
    Define $c_t^{X,Y}(X) = \mathbb{1}[Z_t = Y \text{ and } X \subseteq \mathcal{A}_t]$ and $c_t^{X,Y}(Y) = \mathbb{1}[Z_t = X \text{ and } Y \subseteq \mathcal{A}_t]$.
    Update $p_{t+1}^{X,Y} = \text{EWU}\left(p_t^{X,Y}, c_t^{X,Y}\right)$.
  end
  forall *triples* S **do**
    Define $d_t^S(i) = \ell_t(i) \cdot \mathbb{1}[Z_t \subseteq S \subseteq \mathcal{A}_t],\ \forall i \in S$.
    Update $q_{t+1}^S = \text{EWU}\left(q_t^S, d_t^S\right)$.
  end
end

Algorithm 6 SELECTIONRULE

Input: $\mathcal{A}_t$: available action set at time t
$(p_t^{X,Y})_{X,Y}$: hedge probabilities for all disjoint pairs
$(q_t^S)_S$: hedge probabilities for all triples S
Initialization:
forall *distinct pairs* X, Y **do** sample $A_t^{X,Y} \sim p_t^{X,Y}$.
forall *triples* S **do** sample $b_t^S \sim q_t^S$.
Pair $X \subseteq \mathcal{A}_t$ is a *good pair* if $A_t^{X,Y} = X$ for all $Y \subseteq \mathcal{A}_t$ such that $X \cap Y = \emptyset$.
if *there is no good pair* **then** arbitrarily choose an $a_t \in \mathcal{A}_t$.
else if *there is a common action in all good pairs* **then** let a_t be such a common action.
else there are exactly three good pairs of the form $\{i, j\}, \{j, k\}, \{k, i\}$; let $a_t = b_t^{\{i,j,k\}}$.
return a_t.

- If there is no good pair, then clearly Z_t cannot be good.
- If there is exactly one good pair, then Z_t would be that pair. Therefore, the algorithm would have selected an element of Z_t, implying that $\ell_t(a_t) = 0$, a contradiction.
- If all good pairs have one common action, and Z_t is one of them, then the algorithm selects an element in the intersection of the good pairs. In particular, the element $a_t \in Z_t$, so $\ell_t(a_t) = 0$, a contradiction.

Since Z_t is not a good pair, there exists a pair $X \subseteq \mathcal{A}_t$ such that $A_t^{X,Z_t} = X$, and thus

$$c_t^{X,Z_t}(A_t^{X,Z_t}) = c_t^{X,Z_t}(X) = 1,$$

proving the lemma statement. The only remaining case is when there are exactly three good pairs $\{i,j\},\{j,k\},\{k,i\}$. If Z_t is not one of these pairs, then the exact same argument holds; otherwise, since $\ell_t(a_t)=1$, we must have $Z_t=\{i,j\}$ and $a_t=k$ and therefore,

$$d_t^{i,j,k}(b_t^{i,j,k}) = d_t^{i,j,k}(k) = \ell_t(k) = 1,$$

finishing the proof. ■

Lemma 10 HOPP *ensures that for all disjoint pairs X,Y,*

$$\mathbb{E}\left[\sum_{t=1}^{T} c_t^{X,Y}(A_t^{X,Y})\right] \le \frac{\eta}{1-\mathrm{e}^{-\eta}}\mathbb{E}\left[\min_{\sigma}\sum_{t=1}^{T} c_t^{X,Y}(\sigma(X,Y))\right] + \frac{\ln 2}{1-\mathrm{e}^{-\eta}},$$

where $\sigma(X,Y)=X$ if $\sigma(X\cup Y)\in X$ and $\sigma(X,Y)=Y$ otherwise. Also, for all triples S,

$$\mathbb{E}\left[\sum_{t=1}^{T} d_t^{S}(b_t^{S})\right] \le \frac{\eta}{1-\mathrm{e}^{-\eta}}\mathbb{E}\left[\min_{\sigma}\sum_{t=1}^{T} d_t^{S}(\sigma(S))\right] + \frac{\ln 3}{1-\mathrm{e}^{-\eta}}.$$

Proof Note that the value of $c_t^{X,Y}$ is independent of $A_t^{X,Y}$, and the value of d_t^S is independent of b_t^S. Therefore, the first bound is obtained by applying Lemma 1 with $R=1$ and $\mathcal{S}=\{X,Y\}$, and the second bound by applying the same lemma with $R=1$ and $\mathcal{S}=S$. ■

Lemma 11 HOPP *ensures that for all rankings σ,*

$$\sum_{X,Y \text{ disjoint}} c_t^{X,Y}(\sigma(X,Y)) \le \binom{K-2}{2}\cdot \ell_t(\sigma(\mathcal{A}_t)) \text{ and } \sum_{S:|S|=3} d_t^{S}(\sigma(S)) \le (K-2)\cdot \ell_t(\sigma(\mathcal{A}_t)).$$

Proof If $\ell_t(\sigma(\mathcal{A}_t))=0$, then $\sigma(X,Z_t)=Z_t$ for every $X\subseteq\mathcal{A}_t$. Also, $\sigma(Z_t\cup\{i\})\in Z_t$ for every $i\in\mathcal{A}_t$. Therefore, by the construction of $c_t^{X,Y}$ and d_t^S, we have

$$\sum_{X,Y \text{ disjoint}} c_t^{X,Y}(\sigma(X,Y)) = \sum_{X\subseteq\mathcal{A}_t\setminus Z_t} c_t^{X,Z_t}(\sigma(X,Z_t)) = \sum_{X\subseteq\mathcal{A}_t\setminus Z_t} c_t^{X,Z_t}(Z_t) = 0.$$
$$\sum_{S:|S|=3} d_t^{S}(\sigma(S)) = \sum_{i\in\mathcal{A}_t\setminus Z_t} d_t^{Z_t\cup\{i\}}(\sigma(Z_t\cup\{i\})) = 0.$$

When $\ell_t(\sigma(\mathcal{A}_t))=1$, we have $\sum_{X,Y \text{ disjoint}} c_t^{X,Y}(\sigma(X,Y)) = \sum_{X\subseteq\mathcal{A}_t\setminus Z_t} 1 \le \binom{K-2}{2} = \binom{K-2}{2}\cdot \ell_t(\sigma(\mathcal{A}_t))$, proving the first inequality. For the second inequality, we use $\sum_S d_t^S(\sigma(S)) \le \sum_{i\in\mathcal{A}_t\setminus Z_t} 1 \le K-2 = (K-2)\cdot\ell_t(\sigma(\mathcal{A}_t))$. ■

We are now ready to prove the theorem.

Proof [of Theorem 8] We apply Lemmas 9, 10, 11 successively:

$$\begin{aligned}
\mathbb{E}\left[\sum_{t=1}^{T}\ell_t(a_t)\right] &\le \mathbb{E}\left[\sum_{t=1}^{T}\left(\sum_{X,Y}c_t^{X,Y}(A_t^{X,Y})+\sum_{S:|S|=3}d_t^S(b_t^S)\right)\right] \\
&\le \frac{\eta}{1-\mathrm{e}^{-\eta}}\mathbb{E}\left[\min_{\sigma}\sum_{t=1}^{T}\left(\sum_{X,Y}c_t^{X,Y}(\sigma(X,Y))+\sum_{S:|S|=3}d_t^S(\sigma(S))\right)\right]+\mathcal{O}\left(\frac{N^4}{1-\mathrm{e}^{-\eta}}\right) \\
&\le \frac{\eta}{1-\mathrm{e}^{-\eta}}\cdot\mathbb{E}\left[\mathcal{O}(K^2)\min_{\sigma}\sum_{t=1}^{T}\ell_t(\sigma(\mathcal{A}_t))\right]+\mathcal{O}\left(\frac{N^4}{1-\mathrm{e}^{-\eta}}\right).
\end{aligned}$$

Picking $\eta = 1$ completes the proof. ■

4. The Bandit Setting

For the bandit setting, we consider two regimes. The first is the setting of Section 3.1, i.e., in each round, exactly one action has zero loss. We show how to adapt Algorithm 3 to the bandit setting while maintaining the same $\mathcal{O}(\log K)$ approximation ratio, albeit at the cost of larger additive regret. Then, we consider the bandit model without any assumptions on the sizes of available action sets or numbers of zero-loss actions. In this case, we give an algorithm with approximation ratio $\mathcal{O}(N)$.

Bandit-HATT. We begin by considering the setting of Section 3.1, i.e., in each round t, exactly one action z_t has loss 0, while all others have loss 1. We show how to combine the ideas of Algorithm 3 with the "inverse-propensity weighting" technique to turn the algorithm into a bandit algorithm.

Since the algorithm does not learn the loss of all actions, we cannot define $c_t^{i,j}$ as in Algorithm 3. However, notice that when the learner happens to draw the zero-loss action at time t (i.e., $\ell_t(a_t) = 0$), she can infer all other actions' losses. Based on this observation, we can define an unbiased estimator for the $c_t^{i,j}$ in Algorithm 3. First, we define an exploration indicator ρ_t, which is drawn independently in each round t, and is 1 with probability μ and 0 otherwise. If $\rho_t = 1$, then a_t is drawn uniformly randomly from $\mathcal{A}_t$; otherwise, a_t is set to the output of Algorithm 4 (as in the full-information setting). Then, we define $c_t^{i,j}(i) = \ell_t(i)\cdot\frac{|\mathcal{A}_t|\cdot\mathbb{1}[\rho_t=1]\cdot\mathbb{1}[\ell_t(a_t)=0]}{\mu}$ if $a_t \in \{i,j\} \in U_t$; otherwise, $c_t^{i,j}(i) = 0$. This number is always accessible because when $\ell_t(a_t) = 0$, the learner can infer the losses of all actions. Note that the $|\mathcal{A}_t|\cdot\mathbb{1}[\rho_t = 1]\cdot\mathbb{1}[\ell_t(a_t) = 0]/\mu$ factor has an expectation of 1 because $\rho_t = 1$ happens with probability μ, and when $\rho_t = 1$, $a_t = z_t$ with probability $1/|\mathcal{A}_t|$. So we see that the $c_t^{i,j}$ in Algorithm 7 are exactly unbiased estimators for the $c_t^{i,j}$ defined in Algorithm 3.

Note that the scaling by μ in the definition of $c_t^{i,j}$ results in values that are not in $[0, 1]$; this is why we needed the more general bound of Lemma 1 for the analysis of Hedge.

Also note that the way we construct the estimators is different from the standard way for the multi-armed bandit problem (Auer et al., 2002), i.e., the special case when $\mathcal{A}_t$ is fixed for all t. The standard way would require computing the exact probability of choosing each action, which is complicated for our algorithm. Moreover, for our problem, to design algorithms with Properties 1 and 2, it is also important to assign non-zero losses to Hedges only when we know exactly what the loss vector is. This is also the reason that we are unable to generalize HOPP to the bandit setting to deal with two zero-loss actions — with bandit feedback the learner can never be sure what the entire loss vector is.

Algorithm 7 Bandit-HATT

forall $i < j$ **do** set $p_1^{i,j}(i) = p_1^{i,j}(j) = \frac{1}{2}$.
for $t = 1, \ldots, T$ **do**
  Receive $\mathcal{A}_t$.
  Let $(\widehat{a}_t, U_t) = \textsc{Tournament}(\mathcal{A}_t, (p_t^{i,j})_{i,j})$.
  Draw $\rho_t \sim \text{Bernoulli}(\mu)$.
  if $\rho_t = 1$ **then** let $a_t \sim \text{Uniform}(\mathcal{A}_t)$ **else** let $a_t = \widehat{a}_t$.
  Choose a_t and suffer loss $\ell_t(a_t)$.
  if $\rho_t = 1$ *and* $\ell_t(a_t) = 0$ **then**
    // In this case, $z_t = a_t$.
    forall i *with* $\{i, z_t\} \in U_t$ **do**
      $c_t^{i,z_t}(i) = \frac{|\mathcal{A}_t| \cdot \mathbb{1}[\rho_t = 1] \cdot \mathbb{1}[\ell_t(a_t) = 0]}{\mu}$, $c_t^{i,z_t}(z_t) = 0$, $p_{t+1}^{i,z_t} = \text{EWU}\left(p_t^{i,z_t}, c_t^{i,z_t}\right)$.
    end
    forall *other* $i < j$ **do** let $c_t^{i,j}(\cdot) = 0$ and $p_{t+1}^{i,j} = p_t^{i,j}$.
  end
  else forall $i < j$ **do** let $c_t^{i,j}(\cdot) = 0$ and $p_{t+1}^{i,j} = p_t^{i,j}$.
end

For Bandit-HATT, we prove the following theorem. Note that the bound enjoys the same $\mathcal{O}(\log(K))$ approximation ratio as in the full-information setting, but suffers $\mathcal{O}(\sqrt{T})$ additive regret.

Theorem 12 Bandit-HATT *(Algorithm 7) guarantees that for any ranking σ,*

$$\mathbb{E}\left[\sum_{t=1}^{T} \ell_t(a_t)\right] \leq \frac{(1 + \lceil \log_2(K) \rceil) \cdot \frac{K\eta}{\mu}}{1 - \mathrm{e}^{-\frac{K\eta}{\mu}}} \cdot \mathbb{E}\left[\sum_{t=1}^{T} \ell_t(\sigma(\mathcal{A}_t))\right] + \mathcal{O}\left(\frac{KN^2}{\mu\left(1 - \mathrm{e}^{-\frac{K\eta}{\mu}}\right)} + \mu T\right).$$

Letting $\mu = \min\left\{N\sqrt{\frac{K}{T}}, 1\right\}$, $\eta = \frac{\mu}{K}$, the above is bounded by $\mathcal{O}(\log(K)) \cdot \mathbb{E}\left[\sum_{t=1}^{T} \ell_t(\sigma(\mathcal{A}_t))\right] + \mathcal{O}\left(N\sqrt{KT} + KN^2\right)$.

Proof By the same argument as in the proof of Lemma 3, there exists some i such that $\widehat{a}_t^{i,z_t} = i$ and

$$\mathbb{1}[\ell_t(a_t) = 0] \cdot \ell_t(\widehat{a}_t) \leq \mathbb{1}[\ell_t(a_t) = 0] \cdot \ell_t(i) \cdot \mathbb{1}[\{i, z_t\} \in U_t].$$

Multiplying both sides by $\frac{|\mathcal{A}_t| \cdot \mathbb{1}[\rho_t = 1]}{\mu}$, we get

$$\ell_t(\widehat{a}_t) \cdot \frac{|\mathcal{A}_t| \cdot \mathbb{1}[\rho_t = 1] \cdot \mathbb{1}[\ell_t(a_t) = 0]}{\mu} \leq c_t^{i,z_t}(i).$$

Thus,

$$\ell_t(\widehat{a}_t) \cdot \frac{|\mathcal{A}_t| \cdot \mathbb{1}[\rho_t = 1] \cdot \mathbb{1}[\ell_t(a_t) = 0]}{\mu} \leq \sum_{i<j} c_t^{i,j}(\widehat{a}_t^{i,j}).$$

Algorithm 8 The LEVEL algorithm

forall *actions* $a \in [N]$ **do** let level$(a) \leftarrow 0$.
for $t = 1, \dots, T$ **do**
  Let $a_t \in \operatorname{argmin}_{a \in \mathcal{A}_t} \text{level}(a)$.
  Choose action a_t and incur loss $\ell_t(a_t)$.
  if $\ell_t(a_t) = 1$ **then** increment level(a_t) by 1.
end

By Lemma 1 with $R = \frac{K}{\mu}$, we have

$$\mathbb{E}\left[\sum_{t=1}^{T} c_t^{i,j}(\widehat{a}_t^{i,j})\right] \leq \frac{\frac{K\eta}{\mu}}{1 - \mathrm{e}^{-\frac{K\eta}{\mu}}} \cdot \mathbb{E}\left[\min_{\sigma} \sum_{t=1}^{T} c_t^{i,j}(\sigma(i,j))\right] + \frac{(\ln 2)\frac{K}{\mu}}{1 - \mathrm{e}^{-\frac{K\eta}{\mu}}}.$$

Then by the same argument as in the proof of Lemma 5, we have

$$c_t^{i,j}(\sigma(i,j)) \leq (1 + \lceil \log_2(K) \rceil)\ell_t(\sigma(\mathcal{A}_t)) \cdot \frac{|\mathcal{A}_t| \cdot \mathbb{1}[\rho_t = 1] \cdot \mathbb{1}[\ell_t(a_t) = 0]}{\mu}.$$

Combining all of the above, we get

$$\begin{aligned}
&\mathbb{E}\left[\ell_t(\widehat{a}_t) \cdot \frac{|\mathcal{A}_t| \cdot \mathbb{1}[\rho_t = 1] \cdot \mathbb{1}[\ell_t(a_t) = 0]}{\mu}\right] \\
&\leq \frac{\frac{K\eta}{\mu}(1 + \lceil \log_2(K) \rceil)}{1 - \mathrm{e}^{-\frac{K\eta}{\mu}}} \cdot \mathbb{E}\left[\sum_{t=1}^{T} \ell_t(\sigma(\mathcal{A}_t)) \cdot \frac{|\mathcal{A}_t| \cdot \mathbb{1}[\rho_t = 1] \cdot \mathbb{1}[\ell_t(a_t) = 0]}{\mu}\right] \\
&\qquad + \mathcal{O}\left(\frac{N^2 \frac{K}{\mu}}{1 - \mathrm{e}^{-\frac{K\eta}{\mu}}}\right).
\end{aligned}$$

Taking the expectation over a_t and ρ_t:

$$\mathbb{E}\left[\sum_{t=1}^{T} \ell_t(\widehat{a}_t)\right] \leq \frac{(1 + \lceil \log_2(K) \rceil) \cdot \frac{K\eta}{\mu}}{1 - \mathrm{e}^{-\frac{K\eta}{\mu}}} \cdot \mathbb{E}\left[\sum_{t=1}^{T} \ell_t(\sigma(\mathcal{A}_t))\right] + \mathcal{O}\left(\frac{KN^2}{\mu\left(1 - \mathrm{e}^{-\frac{K\eta}{\mu}}\right)}\right).$$

Finally, using that $\mathbb{E}\left[\sum_{t=1}^{T} \mathbb{1}[\widehat{a}_t \neq a_t]\right] = \mu T$ completes the proof. ■

The LEVEL Algorithm. Finally, we consider the most challenging setup: bandit feedback without any restrictions on the number of zero-loss actions. The algorithm we present is inspired by similar ideas of Blum et al. (2018) for a very different problem, where a perfect ranking exists. This is generally not true in our setting, and our analysis is also new. The idea is to keep track of a *level* for each action, and to always choose an action a_t with the smallest level among all available actions in $\mathcal{A}_t$. If the chosen action suffers a loss of 1, then that action will be moved down by one level, i.e., its level increases by one (see Algorithm 8). Note that this algorithm is deterministic, and we have the following deterministic guarantee:

Theorem 13 *The* LEVEL *algorithm ensures:* $\sum_{t=1}^{T} \ell_t(a_t) \leq N \min_\sigma \sum_{t=1}^{T} \ell_t(\sigma(\mathcal{A}_t)) + \frac{N(N-1)}{2}$.

The proof of this theorem makes use of the following key lemma.

Lemma 14 *Let* $\text{level}_t(a)$ *be the level of action* a *at the beginning of round* t*. Then for every* t, a*, and* σ*,* $\text{level}_t(a) \leq m_\sigma(a) - 1 + \sum_{\tau=1}^{t-1} \ell_\tau(\sigma(\mathcal{A}_\tau))$*, where* $m_\sigma(a)$ *is the rank of* a *under* σ*.*

Proof We use induction on t. When $t = 1$, the inequality clearly holds. Suppose that the following holds for all a:

$$\text{level}_t(a) \leq m_\sigma(a) - 1 + \sum_{\tau=1}^{t-1} \ell_\tau(\sigma(\mathcal{A}_\tau)).$$

We prove the bound for $t + 1$.

If the level of an action a does not change at time t (i.e., $\text{level}_{t+1}(a) = \text{level}_t(a)$), then the induction step is simple:

$$\text{level}_{t+1}(a) = \text{level}_t(a) \leq m_\sigma(a) - 1 + \sum_{\tau=1}^{t-1} \ell_\tau(\sigma(\mathcal{A}_\tau)) \leq m_\sigma(a) - 1 + \sum_{\tau=1}^{t} \ell_\tau(\sigma(\mathcal{A}_\tau)).$$

Now consider an action a with $\text{level}_{t+1}(a) \neq \text{level}_t(a)$. By our algorithm, this is only possible for $a = a_t$, and only when $\ell_t(a_t) = 1$. Therefore, we only need to prove that $\text{level}_{t+1}(a_t) \leq m_\sigma(a_t) + \sum_{\tau=1}^{t} \ell_\tau(\sigma(\mathcal{A}_\tau))$ under the assumption that $\ell_t(a_t) = 1$. First, if $\ell_t(\sigma(\mathcal{A}_t)) = 1$, then

$$\text{level}_{t+1}(a_t) = \text{level}_t(a_t) + 1 \leq m_\sigma(a_t) + \sum_{\tau=1}^{t-1} \ell_\tau(\sigma(\mathcal{A}_\tau)) = m_\sigma(a_t) - 1 + \sum_{\tau=1}^{t} \ell_\tau(\sigma(\mathcal{A}_\tau)).$$

Second, if $\ell_t(\sigma(\mathcal{A}_t)) = 0$, then since $\ell_t(a_t) = 1$, we have $\sigma(\mathcal{A}_t) \neq a_t$. Because $\sigma(\mathcal{A}_t)$ is the action that σ ranks highest among $\mathcal{A}_t$, we have $m_\sigma(\sigma(\mathcal{A}_t)) \leq m_\sigma(a_t) - 1$. Therefore,

$$\begin{aligned}\text{level}_{t+1}(a_t) = \text{level}_t(a_t) + 1 &\overset{(*)}{\leq} \text{level}_t(\sigma(\mathcal{A}_t)) + 1 \;\leq\; m_\sigma(\sigma(\mathcal{A}_t)) + \sum_{\tau=1}^{t-1} \ell_\tau(\sigma(\mathcal{A}_\tau)) \\ &\leq\; m_\sigma(a_t) - 1 + \sum_{\tau=1}^{t-1} \ell_\tau(\sigma(\mathcal{A}_\tau)) \;=\; m_\sigma(a_t) - 1 + \sum_{\tau=1}^{t} \ell_\tau(\sigma(\mathcal{A}_\tau)).\end{aligned}$$

In the step marked $(*)$, we used the specific choice of a_t made in the algorithm. This finishes the induction. ■

Proof [of Theorem 13] Observe that the sum of $\text{level}_t(a)$ over a is always the total number of mistakes the learner has made up to time $t - 1$. Therefore, for any ranking σ,

$$\begin{aligned}\sum_{t=1}^{T} \ell_t(a_t) = \sum_{a \in [N]} \text{level}_{T+1}(a) &\leq \sum_{a \in [N]} \left(m_\sigma(a) - 1 + \sum_{t=1}^{T} \ell_t(\sigma(\mathcal{A}_t)) \right) \\ &= \frac{N(N-1)}{2} + N \sum_{t=1}^{T} \ell_t(\sigma(\mathcal{A}_t)),\end{aligned}$$

where the inequality is by Lemma 14. ■

With LEVEL, we can actually deal with any sleeping expert/bandit problems with real-valued losses $\ell_t(a) \in [0,1]$. A reduction from the case of real-valued losses to binary losses can be done with random rounding: when facing a loss $\ell_t(a)$, the algorithm generates a randomized version $\ell'_t(a)$, which is 1 with probability $\ell_t(a)$ and 0 otherwise; then $\ell'_t(\cdot)$ is fed to the LEVEL algorithm as given above. This preserves the expectation of the losses suffered by the learner and any ranking (i.e., $\mathbb{E}[\ell'_t(a_t)] = \mathbb{E}[\ell_t(a_t)]$, $\mathbb{E}[\ell'_t(\sigma(\mathcal{A}_t))] = \mathbb{E}[\ell_t(\sigma(\mathcal{A}_t))]$ for any t and any σ), and thus does not affect the expected regret.

Note that while the LEVEL algorithm can handle the most general case and enjoys $\mathcal{O}(N^2)$ additive regret, the approximation ratio is N, which could be much larger than K.

5. Conclusions

We revisited the problem of online learning with changing action sets in the adversarial setting and developed the first efficient algorithms with approximate regret guarantees, for both the general setting with bandit feedback and several special cases where significant improvements are obtained. One clear open question is whether poly(K) approximation ratio is achievable generally, without restrictions on the number of zero-loss actions, even for the full-information setting. An intermediate step would be to show that for any constant number z of zero-loss actions, there is an algorithm with regret approximation ratio $O(K^{f(z)})$ for some function f; we have so far only shown algorithms for $z \leq 2$. Perhaps an even more basic question is whether there is a single algorithm that works when the number of zeros $z \in \{0,1\}$ can change between rounds, and the algorithm does not know the number of zeros in a given round. Another direction is to improve the additive $\mathcal{O}(\sqrt{T})$ regret for the bandit setting with one zero-loss action.

Acknowledgement

We thank Elad Hazan and He Jiang for working with us in the early stage of this project, and thank anonymous reviewers for providing very constructive comments. EE and DK were supported in part by grants NSF IIS-1619458 and ARO W911NF1810208. HL and CYW were supported in part by NSF IIS1755781 and NSF IIS1943607.

References

Jacob D Abernethy. Can we learn to gamble efficiently? In *Proc. 23rd Conference on Learning Theory*, pages 318–319, 2010.

Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002.

Pranjal Awasthi, Avrim Blum, and Or Sheffet. Improved guarantees for agnostic learning of disjunctions. In *Conference on Learning Theory*, 2010.

Avrim Blum, Yishay Mansour, and Jamie Morgenstern. Learning what's going on: Reconstructing preferences and priorities from opaque transactions. *ACM Transactions on Economics and Computation (TEAC)*, 6(3-4):1–20, 2018.

Sébastien Bubeck, Michael Cohen, and Yuanzhi Li. Sparsity, variance and curvature in multi-armed bandits. In *Algorithmic Learning Theory*, 2018.

Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge University Press, 2006.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, August 1997.

Yoav Freund, Robert E Schapire, Yoram Singer, and Manfred K Warmuth. Using and combining predictors that specialize. In *Proc. 29th ACM Symp. on Theory of Computing*, pages 334–343, 1997.

Dan Garber. Efficient online linear optimization with approximation algorithms. In *Proc. 31st Advances in Neural Information Processing Systems*, pages 627–635, 2017.

Elad Hazan, Satyen Kale, and Shai Shalev-Shwartz. Near-optimal algorithms for online matrix prediction. In *Proc. 25th Conference on Learning Theory*, 2012.

Varun Kanade and Thomas Steinke. Learning hurdles for sleeping experts. *ACM Transactions on Computation Theory (TOCT)*, 6(3):11, 2014.

Varun Kanade, H Brendan McMahan, and Brent Bryan. Sleeping experts and bandits with stochastic action availability and adversarial rewards. In *Proc. 12th Intl. Conf. on Artificial Intelligence and Statistics*, 2009.

Robert Kleinberg, Alexandru Niculescu-Mizil, and Yogeshwer Sharma. Regret bounds for sleeping experts and bandits. *Machine Learning*, 80(2-3):245–272, 2010.

Joon Kwon and Vianney Perchet. Gains and losses are fundamentally different in regret minimization: The sparse case. *The Journal of Machine Learning Research*, 17(1):8106–8137, 2016.

Gergely Neu and Michal Valko. Online combinatorial optimization with stochastic decision sets and adversarial losses. In *Proc. 28th Advances in Neural Information Processing Systems*, pages 2780–2788, 2014.

Tim Roughgarden and Joshua R Wang. An optimal algorithm for online unconstrained submodular maximization. In *Proc. 31st Conference on Learning Theory*, 2018.

Aadirupa Saha, Pierre Gaillard, and Michal Valko. Improved sleeping bandits with stochastic actions sets and adversarial rewards. In *International Conference on Machine Learning*, 2020.

Hamid Shayestehmanesh, Sajjad Azami, and Nishant A Mehta. Dying experts: Efficient algorithms with optimal regret bounds. In *Proc. 33rd Advances in Neural Information Processing Systems*, pages 9983–9992, 2019.