Dueling Bandits: From Two-dueling to Multi-dueling

Yihan Du
IIIS, Tsinghua University
Beijing, China
duyh18@mails.tsinghua.edu.cn

Siwei Wang
IIIS, Tsinghua University
Beijing, China
wangsw15@mails.tsinghua.edu.cn

Longbo Huang
IIIS, Tsinghua University
Beijing, China
longbohuang@mail.tsinghua.edu.cn

ABSTRACT

We study a general multi-dueling bandit problem, where an agent compares multiple options simultaneously and aims to minimize the regret due to selecting suboptimal arms. This setting generalizes the traditional two-dueling bandit problem and finds many real-world applications involving subjective feedback on multiple options. We start with the two-dueling bandit setting and propose two efficient algorithms, DoublerBAI and MultiSBM-Feedback. DoublerBAI provides a generic schema for translating known results on best arm identification algorithms to the dueling bandit problem, and achieves a regret bound of $O(\ln T)$. MultiSBM-Feedback not only has an optimal $O(\ln T)$ regret, but also reduces the constant factor by almost a half compared to benchmark results. Then, we consider the general multi-dueling case and develop an efficient algorithm MultiRUCB. Using a novel finite-time regret analysis for the general multi-dueling bandit problem, we show that MultiRUCB also achieves an $O(\ln T)$ regret bound and the bound tightens as the capacity of the comparison set increases. Based on both synthetic and real-world datasets, we empirically demonstrate that our algorithms outperform existing algorithms.

KEYWORDS

Multi-Armed Bandits; Dueling Bandits; Exploration-Exploitation Trade-off; Online Learning

ACM Reference Format:

Yihan Du, Siwei Wang, and Longbo Huang. 2020. Dueling Bandits: From Two-dueling to Multi-dueling. In *Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), Auckland, New Zealand, May 9–13, 2020,* IFAAMAS, 16 pages.

1 INTRODUCTION

The stochastic Multi-Armed Bandit (MAB) problem is a classic online learning problem and has been extensively studied [2, 4, 23]. It has a wide range of applications such as clinical trials [25], recommendation systems [14], and online advertisement [9]. In the MAB problem, an agent chooses one option from K alternatives, often called "arms," and observes a numerical reward at each time-step. The goal is to minimize the cumulative regret, defined as the expected difference between the actual reward collected and the offline optimal reward.

The dueling bandits problem [27] is an important variant of the MAB problem. In this problem, an agent chooses a pair of arms every time, but only observes the outcome of a noisy comparison between the two selected arms. This setting is particularly useful in applications involving implicit or subjective (human) feedback, such as information retrieval [11] and recommendation systems [14].

The original dueling bandit setting focuses on only comparing two arms at any time. In this work, we consider a general K-armed multi-dueling bandit problem which has been studied by [8, 22], and propose a novel finite-time analysis for that. In this setting, an agent selects a subset of arms with size at most m ($2 \le m \le K$), and observes pairwise dueling outcomes in the selected subset. The objective is to minimize the regret, being the advantage that the optimal arm has over the chosen arms, cumulated up to T plays.

This multi-dueling bandit model can be used in many real-world applications. For example, in information retrieval, the emergence of numerous ranking algorithms (often called "rankers," *e.g.*, PageRank [16] and BM25 [19]) necessitates efficient methods to evaluate these rankers. Conventional online ranker evaluation methods often use interlearving comparison [18, 20, 26], which produces a combined result list of two rankers and translates the user clicks on this list to preference feedback. Recently, several multileaving methods [7, 12, 15] have been proposed, which permits multiple rankers to be compared at once and provides detailed feedback about how these rankers compare to each other, using less data than sequential interleaving comparisons. However, previous works did not address the key issue of how to select a subset of rankers for each comparison, in order to balance between finding the potentially optimal ranker and presenting results of instantaneously good rankers to users, namely the exploration-exploitation trade-off. The multi-dueling bandit model, on the other hand, provides a principled way of selecting multiple rankers ("arms") for each comparison with the objective of guaranteeing few results of poor rankers to be presented to users (regret minimization).

Another application of the multi-dueling bandit model is the problem of online treatment decision in clinical trials. For instance, in motor function recovery, patients' motor responses to treatments are hard to quantify. Thus, treatment performance is evaluated by clinicians via pairwise comparisons [21]. Since clinical trial is expensive and time-consuming, it is more efficient to compare multiple treatments simultaneously in a single trial rather than conducting sequential trials on treatment pairs. The clinicians often provide a ranking of patients' recovery status in a trial, which can be transformed to all pairwise feedbacks. The multi-dueling bandit model can efficiently handle this sequential decision making problem to maximize treatment gains with lower economic costs.

Note that our algorithm and finite-time analysis for the multi-dueling bandit setting is not a trivial extension. Indeed, if one naively extends algorithms for two-dueling bandits to multi-dueling bandits by repeatedly performing the original strategies of selecting two arms, it is hard to simultaneously guarantee an efficient selection of comparing arms, a small overall regret, and that the

Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), B. An, N. Yorke-Smith, A. El Fallah Seghrouchni, G. Sukthankar (eds.), May 9–13, 2020, Auckland, New Zealand. © 2020 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

regret improves as m increases, three desired features of effective algorithms.

To design efficient algorithms for our problem, we first revisit the original dueling bandit problem, and propose two efficient algorithms, called DoublerBAI and MultiSBM-Feedback. Our algorithms build upon the Doubler and MultiSBM algorithms in [3], which reduces the dueling bandits problem to the conventional stochastic MAB problem. DoublerBAI incorporates Best Arm Identification (BAI) algorithms to the dueling bandit problem, and improves the regret bound of Doubler from $O((\ln T)^2)$ to optimal $O(\ln T)$. MultiSBM-Feedback, on the other hand, not only has an optimal regret bound of $O(\ln T)$, but also reduces the constant factor of the logarithmic term by almost a half, compared to benchmark results. This regret bound is comparable with that of UCB [4] in a standard MAB problem in terms of both order and factor. We then turn to the general formulation with comparing m arms, and propose an efficient algorithm, called MultiRUCB. We prove that MultiRUCB achieves an $O(\ln T)$ regret, and the regret bound tightens as the size of the comparison set m increases, which cannot be achieved by directly applying existing two-dueling bandit solutions. This implies that given the ability of simultaneously comparing more arms, MultiRUCB efficiently exploits more information, and its performance boosts as such ability increases.

While there have been previous work on the multi-dueling bandit problem [8, 22], to the best of our knowledge, this is the first work to provide a finite-time regret analysis for the general multi-dueling bandit problem. Moreover, we conduct experiments based on both the synthetic and real-world datasets [17]. The results demonstrate the superior performance of our algorithms over existing benchmarks.

2 PROBLEM SETTING

We consider a general K-armed multi-dueling bandit problem, where an agent is given a set of K arms, denoted by $\mathcal{X} := \{x_1, x_2, ..., x_K\}$. At each time-step $t \in \{1, 2, ..., T\}$, the agent selects a subset $\mathcal{A}_t \subset \mathcal{X}$ for comparison, where the size of $\mathcal{A}_t$ is constrained by $|\mathcal{A}_t| \leq m$ $(2 \leq m \leq K)$, and observes all pairwise dueling outcomes in $\mathcal{A}_t$. Specifically, dueling comparison works as follows [3]. Each arm $x_i \in \mathcal{X}$ has a latent utility distribution in $[0, 1]$ with expectation $\mu(x_i)$. Then, there is a link function $\phi : [0, 1] \times [0, 1] \mapsto [0, 1]$, based on which the probability that arm x_i beats arm x_j is given by $p_{ij} = \phi\big(\mu(x_i), \mu(x_j)\big)$. The dueling outcome for arm i and arm j at every time is an independent Bernoulli random variable that takes value 1, representing arm i beats arm j, with probability p_{ij}.

As in [3], in this paper, we focus on the following linear link function:[1]

$$\phi\big(\mu(x_i), \mu(x_j)\big) := \frac{\mu(x_i) - \mu(x_j) + 1}{2}.$$

We also assume without loss of generality that $\mu(x_1) > \mu(x_2) \geq ... \geq \mu(x_K)$. We use $P := [p_{ij}]$, whose ij-th entry is the preference probability p_{ij}, to denote the $K \times K$ preference matrix .

For the multi-dueling bandit problem, the expected cumulative regret up to time T is defined to be:

$$\mathbb{E}[R_T] := \sum_{t=1}^{T} \sum_{a \in \mathcal{A}_t} \frac{1}{|\mathcal{A}_t|} \Delta(x_1, a),$$

where $\Delta(x_i, x_j) := p_{ij} - \frac{1}{2} \in [-\frac{1}{2}, \frac{1}{2}]$ is a measure of the distinguishability between two arms. This regret measures the average advantage that the best arm has over the $|\mathcal{A}_t|$ arms being chosen at each time-step t. This implies that an expected zero regret can be achieved if and only if $\mathcal{A}_t = \{x_1\}$. Note that when $m = 2$, our problem becomes the original two-dueling bandit problem. For ease of notation, below we write Δ_{ij} for $\Delta(x_i, x_j)$ and Δ_i for $\Delta(x_1, x_i)$.

Note that our multi-dueling bandit formulation is different from [8]. In our setting, the algorithm can choose at most m different arms rather than an arbitrary subset of K arms at each time-step t. This scenario fits many practical applications better, as the number of arms being compared simultaneously is often constrained. While our multi-dueling bandit setting is the same to that in [22], we are the first to provide a finite-time regret analysis for this problem.

3 ALGORITHMS FOR TWO-DUELING BANDITS

We first start from the special case when $m = 2$, i.e., the original two-dueling bandit problem,[2] and propose two efficient algorithms DoublerBAI and MultiSBM-Feedback for achieving an optimal regret. Our algorithms build upon the Doubler and MultiSBM algorithms in [3].

3.1 DoublerBAI with Best Arm Identification Algorithms

To present our algorithm, we define a generic Best Arm Identification Machine (BAIM) as a procedure which performs a K-armed BAI algorithm with an internal timer and memory, e.g., LUCB [13]. A BAIM has five operations: Reset, Advance, Feedback, StopTest and Return. The Reset operation clears its state. The Advance operation decides the next arm to play. The Feedback operation updates its state with the observed information. The StopTest operation checks whether the internal BAI algorithm has terminated and the Return operation returns the identified best arm.

With the BAIM procedure above, Algorithm 1 presents the formal definition of DoublerBAI. Generally speaking, we first divide the time horizon into exponentially growing epochs, motivated by the doubling trick [5, 6]. Then, in each epoch i, we fix one arm $\bar{x}_i$ (the left arm) of the played duel $(\bar{x}_i, y_t)$, and adaptively choose the other arm y_t (the right arm) using an exploration-then-exploitation strategy.

In the stage of exploration (Lines 16-22), we choose the right arm y_t according to the sample strategy provided by S, the BAIM procedure, and feed back the dueling outcome b_t to S. Thus, S is actually estimating the probabilities of arms in $\mathcal{X}$ beating the fixed $\bar{x}_i$, and identifying the best arm. Once the internal BAI algorithm in S terminates and returns the identified best arm $\hat{x}_i$ (Lines 20-22), we

[1] We also extend our results to more general non-utility-based models [27] in Section 5, and show numerical results for the extended models in our experiments.

[2] When $m = 2$, having $|\mathcal{A}_t| = 1$ is equivalent to selecting (a_0, a_0) (in this case $\mathcal{A}_t$ only contains a single arm a_0) in the original two-dueling bandit problem. Therefore, our setting reduces to the original two-dueling bandit problem when $m = 2$.

Algorithm 1: DoublerBAI

Input: Exponentially growing sequence $\{T_i\}_{i \in \mathbb{N}}$, where $T_i = \lfloor a^{b^i} \rfloor$ $(a, b > 1)$

1 $S \leftarrow$ new BAIM over $\mathcal{X}$;
2 Set the identified best arm $\hat{x}_i$ = NULL for all epoch $i \in \{0, 1, \ldots\}$;
3 Set the length of epoch i $\tau_i = \begin{cases} T_0, & i = 0 \\ T_i - T_{i-1}, & i > 0 \end{cases}$;
4 **while** true **do**
5   **if** $\hat{x}_{i-1} \neq$ NULL **then**
6     $\bar{x}_i \leftarrow \hat{x}_{i-1}$;
7   **else**
8   Choose $\bar{x}_i$ randomly from $\mathcal{X}$;
9   Reset(S, $\delta_i = \frac{1}{\tau_{i+1}}$);
10   **for** $j = 1, \ldots, \tau_i$ **do**
11   **if** $\hat{x}_i \neq$ NULL **then**
12   `//exploit`
13       $y_t \leftarrow \hat{x}_i$;
14   Play $(\bar{x}_i, y_t)$;
15   **else**
16   `//explore`
17       $y_t \leftarrow$ Advance(S);
18   Play $(\bar{x}_i, y_t)$ and observe the binary dueling outcome b_t;
19   Feedback(S, b_t);
20   **if** StopTest(S)=true **then**
21         $\hat{x}_i \leftarrow$ Return(S);
22         $\tau_i^{\text{explore}} \leftarrow j$;
23     $t \leftarrow t + 1$;
24   $i \leftarrow i + 1$;

enter the exploitation stage (Lines 12-14) and y_t is chosen to be $\hat{x}_i$. For the choice strategy of the left arm $\bar{x}_i$ (Lines 5-8), if S terminates and returns a best arm in the previous epoch, i.e., $\hat{x}_{i-1} \neq$ `NULL`, then we set $\bar{x}_i$ to be the identified best arm $\hat{x}_{i-1}$ found in the previous epoch. Otherwise, we simply choose $\bar{x}_i$ randomly from $\mathcal{X}$.

The key of DoublerBAI is to identify the best arm with high probability in each epoch and fix the left arm in the next epoch as the identified arm. The error probability of the BAIM in each epoch is set according to the length of the next epoch. This guarantees that the expected regret of the left arm is a constant and the regret of the right arms is bounded by the internal regret of the BAIM.

The following theorem provides the regret bound for DoublerBAI.

Theorem 3.1. *Consider a K-armed utility-based two-dueling bandits game. Assume that the BAIM S in DoublerBAI has a sample complexity of $O(H \ln(\frac{H}{\delta}))$, where S outputs the best arm with probability at least $1-\delta$. Given an exponentially growing sequence $\{T_i\}_{i \in \mathbb{N}}$ with parameters $a, b > 1$, i.e., $T_i = \lfloor a^{b^i} \rfloor$, the expected regret of DoublerBAI is bounded by*

$$\mathbb{E}[R_T] = O((H \ln H)^b) + O(H \ln T) + O(H \ln H \ln \ln T) + O(\ln \ln T),$$

where $H := \sum_{i=2}^{K} \frac{1}{\Delta_i^2}$ is the problem complexity for a bandit instance.

Proof sketch. (Please refer to Section A of the supplementary material [1] for the full proof).

We first consider the regret incurred by the right arm y_t. Let $B(\delta)$ denote the supremum of the expected regret of S (the BAIM) to identify the best arm with probability at least $1 - \delta$. In epoch i, after fixing the left arm $\bar{x}_i$, we see that S is playing a standard BAI game in the stage of exploration by estimating the probabilities of arms in $\mathcal{X}$ to beat $\bar{x}_i$. Thus, in epoch i, the expected regret in S is $\mathbb{E}[\sum_{t=1}^{\tau_i^{\text{explore}}} \frac{\mu(x_1)-\mu(y_t)+1}{2}] \leq B(\frac{1}{\tau_{i+1}})$. Specifically, according to the definition of regret for dueling bandits, we observe that the expected regret of the right arm y_t in the stage of exploration, which exactly equals to the left-hand side of the inequality, can be bounded by $B(\frac{1}{\tau_{i+1}})$. Using the explore-then-exploit strategy, the expected regret of the right arm in epoch i can be bounded by $(1 - \frac{1}{\tau_{i+1}})B(\frac{1}{\tau_{i+1}}) + \frac{1}{\tau_{i+1}}O(\tau_i)$. Taking a summation over all epochs (there are $O(\ln \ln T)$ epochs), we obtain the main term $O(H \ln T)$ of the bound presented in Theorem 3.1.

Next, we consider the left arm. If the previous epoch returns an identified best arm $\hat{x}_{i-1}$ (with error probability at most $\frac{1}{\tau_i}$), then the left arm in epoch i is fixed as $\bar{x}_i = \hat{x}_{i-1}$, which incurs expected regret of $\frac{1}{\tau_i} \cdot O(\tau_i) + (1 - \frac{1}{\tau_i}) \cdot 0$. Otherwise, the left arm $\bar{x}_i$ is chosen randomly, which incurs linear expected regret of $O(\tau_i)$. However, one can prove that the latter case only occurs in early short epochs, and the regret can be bounded by $O((H \ln H)^b)$. □

Remark 1. Theorem 3.1 suggests that our DoublerBAI improves the upper bound over its baseline, i.e., Doubler in [3], from $O((\ln T)^2)$ (Theorem 3.1 in [3]) to $O(\ln T)$ by efficiently incorporating BAI algorithms. The upper bound of our DoublerBAI has an additional problem-dependent term $O((H \ln H)^b)$, caused by not being able to identify best arms due to insufficient epochs length. Yet, by setting b close to 1, $O((H \ln H)^b)$ becomes negligible for T large enough, which is also efficient in practice.

3.2 MultiSBM-Feedback with Multi-armed Bandit Algorithms

We now consider the second algorithm, MultiSBM-Feedback, which not only has an optimal regret bound of $O(\ln T)$, but also improves the constant factor of its baseline, i.e., MultiSBM in [3].

In MultiSBM-Feedback, we define a Singleton Bandit Machine (SBM) as a generic procedure representing a MAB algorithm with an internal timer and memory. In this work, we implement SBM with a variant of UCB [4] which satisfies the α-robustness property defined in [3]. Below we restate this definition.

Definition 3.2 (α-robustness). Let T_i be the number of times a (suboptimal) arm $x_i \in \mathcal{X}$ is played when running the policy T rounds. A MAB policy is said to be *α-robust* when it has the following property: for all $s \geq 4(\alpha + 4)\Delta_i^{-2} \ln(T)$, it holds that $\Pr[T_i > s] < \frac{2}{\alpha}(s/2)^{-\alpha}$.

An SBM has four operations: Reset, Advance, Feedback and AdditionalFeedback. The first three operations are inherited from

Algorithm 2: MultiSBM-Feedback

1 For all $x \in \mathcal{X}$: $S_x \leftarrow$ new SBM over $\mathcal{X}$, Reset (S_x);
2 $y_0 \leftarrow$ arbitrary element of $\mathcal{X}$;
3 $t \leftarrow 1$;
4 **while** true **do**
5     $x_t \leftarrow y_{t-1}$;
6     $y_t \leftarrow$ Advance(S_{x_t});
7   Play (x_t, y_t), observe choice b_t^y;
8   Feedback(S_{x_t}, b_t^y);
9   **if** $x_t \neq y_t$ **then**
10         $b_t^x \leftarrow 1 - b_t^y$, AdditionalFeedback($S_{y_t}, b_t^x$);
11     $t \leftarrow t + 1$;

MultiSBM. The last AdditionalFeedback is newly added, and plays an important role in improving the regret. AdditionalFeedback receives an additional feedback sent from some arm and updates the SBM's internal state with the additional feedback.

Algorithm 2 presents the procedure of MultiSBM-Feedback. Specifically, we operate K different SBMs in parallel, indexed by the K elements in $\mathcal{X}$. SBM S_x ($x \in \mathcal{X}$) performs an MAB algorithm via estimating the probabilities of arms in $\mathcal{X}$ to beat arm x. At each time-step t, we choose the right arm y_t of the duel (x_t, y_t) according to the strategy provided by SBM S_{x_t} and feed back the outcome b_t^y ($b_t^y = 1$ if y_t wins against x_t, otherwise $b_t^y = 0$) to S_{x_t}. If the two arms are different, we invoke AdditionalFeedback to collect outcome $b_t^x = 1 - b_t^y$ to S_{y_t} (Lines 9-10). In the next time-step, the right arm x_{t+1} is chosen to be y_t. In other words, the right arm in each time-step equals to the left arm in the next time-step.

The key of AdditionalFeedback is to exploit additional feedback from the perspective of x_t, to augment the information in S_{y_t}. This is because after one pull, the outcome of x_t beating y_t and that of y_t beating x_t can be respectively fed back to S_{y_t} and S_{x_t}. Thus, S_{y_t} receives an additional feedback from x_t without pulling x_t, which helps S_{y_t} augment its empirical observations on x_t. Note that in any SBM S_x, the empirical observations received from operations Feedback and AdditionalFeedback are independent. Thus, the Chernoff-Hoeffding bound used in our theoretical analysis still holds.

Algorithm 3 presents the procedure of a SBM. ρ_k denotes the number of times arm $x_k \in \mathcal{X}$ has been pulled. s_k denotes the number of times this SBM receives additional feedback sent from arm x_k. The operation GetAdditionalFeedback is to obtain an additional feedback sent from some left arm x_t in Algorithm 2, which we label as x_j in Algorithm 3. If no additional feedback is sent to this SBM, GetAdditionalFeedback simply returns `NULL`. Every time before SBM pulls (advances) an arm, it invokes GetAdditionalFeedback and updates its empirical observations with the additional feedback received from some arm x_j (Lines 6-9).

The following theorem bounds the expected regret of MultiSBM-Feedback.

Algorithm 3: Implementation of SBM

Input: Confidence interval parameter α
1 $\forall x_k \in \mathcal{X}$, set $\hat{\mu}_k = \infty$;
2 $\forall x_k \in \mathcal{X}$, set $\rho_k = 0$;
3 $\forall x_k \in \mathcal{X}$, set $s_k = 0$;
4 $t \leftarrow 1$;
5 **while** true **do**
6   b^{x_j} =GetAdditionalFeedback();
7   **if** $b^{x_j} \neq$ NULL **then**
8         $\hat{\mu}_j = \frac{\hat{\mu}_j \cdot (\rho_j + s_j) + b^{x_j}}{\rho_j + s_j + 1}$;
9         $s_j = s_j + 1$;
10   Let i be the index maximizing $\hat{\mu}_i + \sqrt{\frac{(\alpha+2)\ln t}{2(\rho_i + s_i)}}$; // $\frac{x}{0} := 1$ `for any` x
11   Play x_i, update $\hat{\mu}_i$, increment ρ_i by 1;
12     $t \leftarrow t + 1$;

Theorem 3.3. *Consider a K-armed utility-based two-dueling bandits game. The expected regret of MultiSBM-Feedback, which implements an SBM defined in Algorithm 3, is bounded by*

$$\mathbb{E}[R_T] \leq \min\left\{\sum_{i>1} \frac{(\alpha+2)\Delta_{max}}{\Delta_i^2} \ln T,\ \sum_{i>1} \frac{2(\alpha+2)}{\Delta_i} \ln T\right\} + \frac{(\alpha+8)\Delta_{max}}{2\alpha} K + \sum_{j>1}\sum_{i>1} O\Big(\frac{\alpha\Delta_{max}}{\Delta_j^2}\big(\ln\ln T + \ln K + \ln(\frac{1}{\Delta_i})\big)\Big),$$

where $\Delta_{max} := \max_{i>1} \Delta_i$ *and the confidence interval parameter* $\alpha = \max\{3, \frac{\ln K}{\ln\ln T}\}$.

Proof sketch. (Please refer to Section B of the supplementary material [1] for the full proof).

According to MultiSBM-Feedback (Algorithm 2), the right arm in each time-step equals to the left arm in the next time-step. Thus, in order to bound the total regret, it suffices to bound the number of times the right arm is suboptimal. Because the right arm is advanced by the SBM indexed by the left arm, we consider the regret from two parts, i.e., suboptimal right arms advanced by S_{x_1} and by S_x ($x \neq x_1$).

We first analyze the latter part. Because the number of times a suboptimal arm $x \neq x_1$ being advanced in any SBM is $O(\ln T)$, according to the results of UCB [4], the number of times x becomes the left arm is $O(K \ln T)$, i.e., the internal timer of S_x ($x \neq x_1$) is order of $O(K \ln T)$. Thus, the number of times a suboptimal right arm advanced by S_x is $O(\ln(K \ln T))$.

Next, we analyze the former part. By exploiting the additional feedbacks, we can prove that in S_{x_1}, $\sum_{i>1} \rho_i(t) = \sum_{i>1} s_i(t)$ for any internal time t. This is because every time S_{x_1} pulls a suboptimal arm ($\sum_{i>1} \rho_i(t)$ increments by 1), it must has received an additional feedback before ($\sum_{i>1} s_i(t)$ increments by 1). Thus, we can prove an expected upper bound of $O(\ln T)$ for $\rho_i(t) + s_i(t)$ ($i > 1$). Therefore, taking a summation over $i > 1$, we obtain a tighter upper bound of $\sum_{i>1} \rho_i(t)$ compared to the original MultiSBM, where the order is still $O(\ln T)$, while the constant shrinks by a half. □

Remark 2. Theorem 3.3 suggests that our MultiSBM-Feedback not only has an optimal regret bound of $O(\ln T)$, but also improves the constant factor of its benchmark result in MultiSBM. This improvement is achieved by additionally exploiting the feedback from the duel. Moreover, the regret bound of MultiSBM is comparable to that of UCB [4] in a standard MAB setting in terms of both order and factor.

Algorithm 4: MultiRUCB

Input: $\alpha > \frac{1}{2}$

1 $\mathbf{W} = [w_{ij}] \leftarrow \mathbf{0}_{K\times K}$;
2 $B \leftarrow \varnothing$;
3 **for** $t = 1, \ldots, T$ **do**
4 $\quad \mathbf{U} := [u_{ij}] = \frac{\mathbf{W}}{\mathbf{W}+\mathbf{W}^T} + \sqrt{\frac{\alpha \ln t}{\mathbf{W}+\mathbf{W}^T}}$;
5 $\quad$ // Element − wise operation; $\frac{x}{0} := 1$ for any x
6 $\quad u_{ii} \leftarrow \frac{1}{2}$ for all $i \in \{1, \ldots, K\}$;
7 $\quad C \leftarrow \{x_c \mid u_{cj} \geq \frac{1}{2}, \ \forall j \in \{1, \ldots, K\}\}$;
8 $\quad$ **if** $C = \varnothing$ **then**
9 $\quad\quad$ Randomly choose m different arms for $\mathcal{A}_t$ from $\mathcal{X}$;
10 $\quad \mathcal{B} \leftarrow \mathcal{B} \bigcap C$;
11 $\quad$ **if** $|C| = 1$ **then**
12 $\quad\quad \mathcal{B} \leftarrow C$;
13 $\quad\quad \mathcal{A}_t \leftarrow C$;
14 $\quad$ **if** $1 < |C| \leq m$ **then**
15 $\quad\quad \mathcal{A}_t \leftarrow C$;
16 $\quad$ **if** $|C| > m$ **then**
17 $\quad\quad$ Choose m different arms for $\mathcal{A}_t$ from C using the following strategy:
18 $\quad\quad$ **if** $\mathcal{B} = \varnothing$ **then**
19 $\quad\quad\quad$ Uniformly choose m different arms for $\mathcal{A}_t$ from C;
20 $\quad\quad$ **else**
21 $\quad\quad\quad$ With probability of $\frac{1}{2}$, add $x_c \in \mathcal{B}$ into $\mathcal{A}_t$ and uniformly add $x_c \in C \setminus \mathcal{B}$ into $\mathcal{A}_t$;
22 $\quad\quad\quad$ With probability of $\frac{1}{2}$, uniformly choose m different arms for $\mathcal{A}_t$ from $C \setminus \mathcal{B}$;
23 $\quad$ Play $\mathcal{A}_t$ and observe all pairwise feedback in $\mathcal{A}_t$;
24 $\quad$ For any pairwise feedback between $x_j, x_k \in \mathcal{A}_t$, increment w_{jk} or w_{kj} depending on which arm wins;

4 MULTIRUCB FOR MULTI-DUELING BANDITS

In this section, we consider the general case $2 \leq m \leq K$, where we can simultaneously compare multiple arms. We propose an efficient algorithm, called MultiRUCB, for the general multi-dueling bandit problem. We conduct a finite-time regret analysis and show that the regret of MultiRUCB is $O(\ln T)$ and tightens as the comparison set size m increases. To the best of our knowledge, this is the first finite-time regret analysis for multi-dueling bandits.

Algorithm 4 presents the procedure of MultiRUCB. We define matrix $W_{K\times K}$ to record the empirical observations, whose ij-th entry denotes the number of times we observe x_i beating x_j ($x_i, x_j \in \mathcal{X}$). Motivated by [30], we also define the relative upper confidence bound matrix $U_{K\times K}$, whose ij-th entry optimistically estimates the preference probability p_{ij}. We maintain a candidate set C which contains potential optimal arms and an empty or singleton set $\mathcal{B}$ which contains the hypothesized optimal arm. Note that the hypothesized optimal arm is removed from $\mathcal{B}$ once it loses to another arm (Line 10). At each time-step t, we choose the comparison set $\mathcal{A}_t$ differently according to the size of C. If $C = \varnothing$ (Lines 8 – 9), we randomly choose m different arms into $\mathcal{A}_t$ from $\mathcal{X}$, which is the trivial case and shown to occur infrequently in our analysis.

Next we discuss three non-trivial cases:

(a) If $|C| = 1$, we are left with a single potential optimal arm x_c, which is hypothesized to be the optimal arm. We put the single arm into $\mathcal{B}$ and $\mathcal{A}_t$ (Lines 11 – 13).
(b) If $1 < |C| \leq m$, all potential optimal arms in C can be compared simultaneously. We simply put all of them into $\mathcal{A}_t$ (Lines 14 – 15).
(c) If $|C| > m$, we cannot put all potential optimal arms into $\mathcal{A}_t$ at once. To choose m different arms from C, if $\mathcal{B}$ is not empty, we give priority to the hypothesized optimal arm in $\mathcal{B}$ and choose the other arms uniformly at random. Otherwise, we uniformly and randomly choose m different arms into $\mathcal{A}_t$ from C (Lines 16 – 22).

The key of MultiRUCB is to exploit as much information as possible from one pull to target $\mathcal{A}_t = \{x_1\}$. C maintains a candidate pool for the potential optimal arms. When C contains multiple arms, which implies that the confidence region of some suboptimal arms are loose, we explore all of them simultaneously as possible. In the case this cannot be done, we wish to put optimal arm x_1 into $\mathcal{A}_t$. Thus, we give priority to the hypothesized optimal arm using the choice strategy define in Lines 18 – 22. This is because x_1 is the most efficient arm to determine the sub-optimality of other arms.

The following theorem provides the regret bound for MultiRUCB.

THEOREM 4.1. *Consider a K-armed multi-dueling bandits game, where the number of comparing arms is at most m at every time. Given $\alpha > 1$, the expected regret of MultiRUCB is bounded by*

$$\mathbb{E}[R_T] \leq \left[\left(\frac{2(4\alpha-1)K^2}{2\alpha-1}\right)^{\frac{1}{2\alpha-1}} \frac{2\alpha-1}{\alpha-1}\right]\Delta_{max}$$
$$+ \min\Bigg\{D\Delta_{max}\ln T,$$
$$(8+2D\ln 2D)\Delta_{max} + \frac{m+1}{m-1}\sum_{i>1}\frac{4\alpha\Delta_{max}}{\Delta_i^2}\ln T\Bigg\},$$

where $D := \sum_{i>1}\frac{4\alpha}{\Delta_i^2} + \sum_{1<i<j}\frac{4\alpha}{C_m^2\Delta_{ij}^2}$ *and* $C_m^2 := \frac{m(m-1)}{2}$.

Proof sketch. (Please refer to Section C of the supplementary material [1] for the full proof).

We see that after $C(\delta) := \left(\frac{(4\alpha-1)K^2}{(2\alpha-1)\delta}\right)^{\frac{1}{2\alpha-1}}$ time-steps, any preference probability p_{ij} ($x_i, x_j \in \mathcal{X}$) will lie in its estimated confidence interval with probability at least $1-\delta$ (Lemma 1 in [30]). Thus, with probability at least $1-\delta$, after $C(\delta)$ time-steps, x_1 exists in

C ($u_{1i} \geq p_{1i} \geq \frac{1}{2}$, $\forall i$). In order to bound the regret after $C(\delta)$ time-steps, it suffices to bound the number of times cases (b) or (c) occurs. For ease of notation, we define two subcases (c-1) and (c-2) of case (c). They respectively refer to the two situations where x_1 is added to $\mathcal{A}_t$ and not.

We first bound the sum of the number of times case (b) and case (c-1) occur. Let $\widetilde{N}_{1i}(t)$ ($i > 1$) denote the number of dueling outcomes between x_1 and x_i we have observed, between time $C(\delta) + 1$ and t. After $C(\delta)$ time-steps, every time case (b) occurs, we can observe at least one outcome of duel between x_1 and some x_i ($i > 1$) ($\sum_{i>1} \widetilde{N}_{1i}(t)$ increments by 1). Every time case (c-1) occurs, we can observe outcomes of $m - 1$ duels between x_1 and x_i ($i > 1$) ($\sum_{i>1} \widetilde{N}_{1i}(t)$ increments by $m - 1$). According to the definition of C, we can prove $\widetilde{N}_{1i}(t) \leq \frac{4\alpha}{\Delta_i^2} \ln t$. Thus, taking a summation over $i > 1$, the total number of times case (b) and case (c-1) occur, between time $C(\delta) + 1$ and t, is bounded by $\sum_{i>1} \widetilde{N}_{1i}(t) \leq \sum_{i>1} \frac{4\alpha}{\Delta_i^2} \ln t$.

Next we bound the number of times case (c-2) occurs. We use $\widetilde{N}_{ij}(t)$ ($1 < i < j$) to denote the number of dueling outcomes between x_i and x_j we have observed between time $C(\delta)+1$ and t. After $C(\delta)$ time-steps, every time case (c-2) occurs, we can observe outcomes of C_m^2 duels between x_i and x_j ($x_i, x_j \in \mathcal{X} \backslash \{x_1\}, x_i \neq x_j$), i.e., $\sum_{1<i<j} \widetilde{N}_{ij}(t)$ increments by C_m^2. According to the definition of C, we can prove $\widetilde{N}_{ij}(t) \leq \frac{4\alpha}{\Delta_{ij}^2} \ln t$, implying $\sum_{1<i<j} \widetilde{N}_{ij}(t) \leq \sum_{1<i<j} \frac{4\alpha}{\Delta_{ij}^2} \ln t$. Since each occurrence of case (c-2) increments $\sum_{1<i<j} \widetilde{N}_{ij}(t)$ by C_m^2, the number of times case (c-2) occurs between time $C(\delta) + 1$ and t is bounded by $\sum_{1<i<j} \frac{4\alpha}{C_m^2 \Delta_{ij}^2} \ln t$. Therefore, we obtain the term $D\Delta_{max} \ln T$ in Theorem 4.1.

Another term $(8+2D \ln 2D)\Delta_{max} + \frac{m+1}{m-1} \sum_{i>1} \frac{4\alpha\Delta_{max}}{\Delta_i^2} \ln T$ in Theorem 4.1 can be obtained by exploiting a geometric distribution with success probability $\frac{1}{2}$, following the procedures in [30]. Specifically, we first need to investigate when $\mathcal{B}$ is set. Define $\widehat{T}_\delta$ as the smallest time satisfying $\widehat{T}_\delta > C(\frac{\delta}{2}) + D \ln \widehat{T}_\delta$, where $\widehat{T}_\delta$ is guaranteed to exist because the left side of the inequality grows linearly with $\widehat{T}_\delta$ and the right side grows logarithmically. It is easy to prove $\widehat{T}_\delta \leq 2C(\frac{\delta}{2}) + 2D \ln 2D$. According to the definition of $\widehat{T}_\delta$, with probability at least $1 - \frac{\delta}{2}$, there exists a time $T_\delta \in (C(\frac{\delta}{2}), \widehat{T}_\delta]$ when case (a) occurs. This implies that with probability at least $1 - \frac{\delta}{2}$, $\mathcal{B}$ has been set as $\mathcal{B} = \{x_1\}$ from time T_δ on.

Then, we know that from time T_δ on, if MultiRUCB carries out case (c), case(c-1) will occur with probability of $\frac{1}{2}$. Let $\widehat{N}^b(t)$, $\widehat{N}_1^c(t)$ and $\widehat{N}_2^c(t)$ denote the number of times case (b), (c-1) and (c-2) occur between time $T_\delta + 1$ and t, respectively. We also introduce two sets of random variables, $\{\tau_0, \tau_1, \tau_2, ...\}$ and $\{n_1, n_2, ...\}$. Define $\tau_0 := T_\delta$ and τ_l as the l^{th} time case (c-1) occurs after time T_δ. Define n_l as the number of times case (c-2) occurs between τ_{l-1} and τ_l. Similar to the above analysis, we can prove that with probability at least $1 - \frac{\delta}{2}$, between time $T_\delta + 1$ and t, case (c-1) occurs at most $L_1^c(t) := \sum_{i>1} \frac{4\alpha}{(m-1)\Delta_i^2} \ln t$ times. Moreover, with probability at least $1 - \frac{\delta}{2}$, for any time $t > T_\delta$, if case (c-1) has occurred $L_1^c(t)$ times, all suboptimal arms x_i ($i > 1$) satisfy $u_{i1} < \frac{1}{2}$ and case (c-2) cannot occur. Thus, we can bound $\widehat{N}_2^c(t)$ by $\sum_{l=1}^{L_1^c(t)} n_l$. Since n_l counts the number of times it takes for case (c) to produce one case (c-1), we can use the conclusion about geometric random variables to bound $\sum_{l=1}^{L_1^c(t)} n_l$. Therefore, we have that with probability at least $1 - \delta$, $\forall t > T_\delta$, $\widehat{N}_2^c(t) \leq \sum_{l=1}^{L_1^c(t)} n_l \leq 2 \sum_{i>1} \frac{4\alpha}{(m-1)\Delta_i^2} \ln t + 4 \ln \frac{2}{\delta}$. Taking summation over T_δ, $\widehat{N}^b(t)$, $\widehat{N}_1^c(t)$ and $\widehat{N}_2^c(t)$, we obtain the term $(8 + 2D \ln 2D)\Delta_{max} + \frac{m+1}{m-1} \sum_{i>1} \frac{4\alpha\Delta_{max}}{\Delta_i^2} \ln T$ in Theorem 4.1.

At last, integrating the confidence term with respect to δ, we obtain the expected regret bound in Theorem 4.1. □

Remark 3. Theorem 4.1 suggests that compared to the two-dueling bandit solutions, MultiRUCB has the same $O(\ln T)$ regret. However, by exploiting more information from one pull, the regret bound of MultiRUCB tightens as the comparison set size m increases, which is unachievable through only repeating existing two-dueling bandit solutions. This implies that our extension of the algorithm and finite-time analysis from two-dueling to multi-dueling is non-trivial and useful. Moreover, to the best of our knowledge, MultiRUCB is the first algorithm providing a finite-time regret analysis for multi-dueling bandits.

5 EXTENSION OF THE LINK FUNCTION

Our analysis of DoublerBAI and MultiSBM-Feedback assumes the linear link function $\phi\big(\mu(x_i), \mu(x_j)\big) := \frac{\mu(x_i)-\mu(x_j)+1}{2}$. In this section, we generalize the linear link function to more general non-utility-based models in [27].

It can be verified that our analysis still holds when $\Delta(\cdot, \cdot)$ satisfies the following property:

Property 1. For some $\gamma > 0$ and any two arms $x_i, x_j \in \mathcal{X}$,

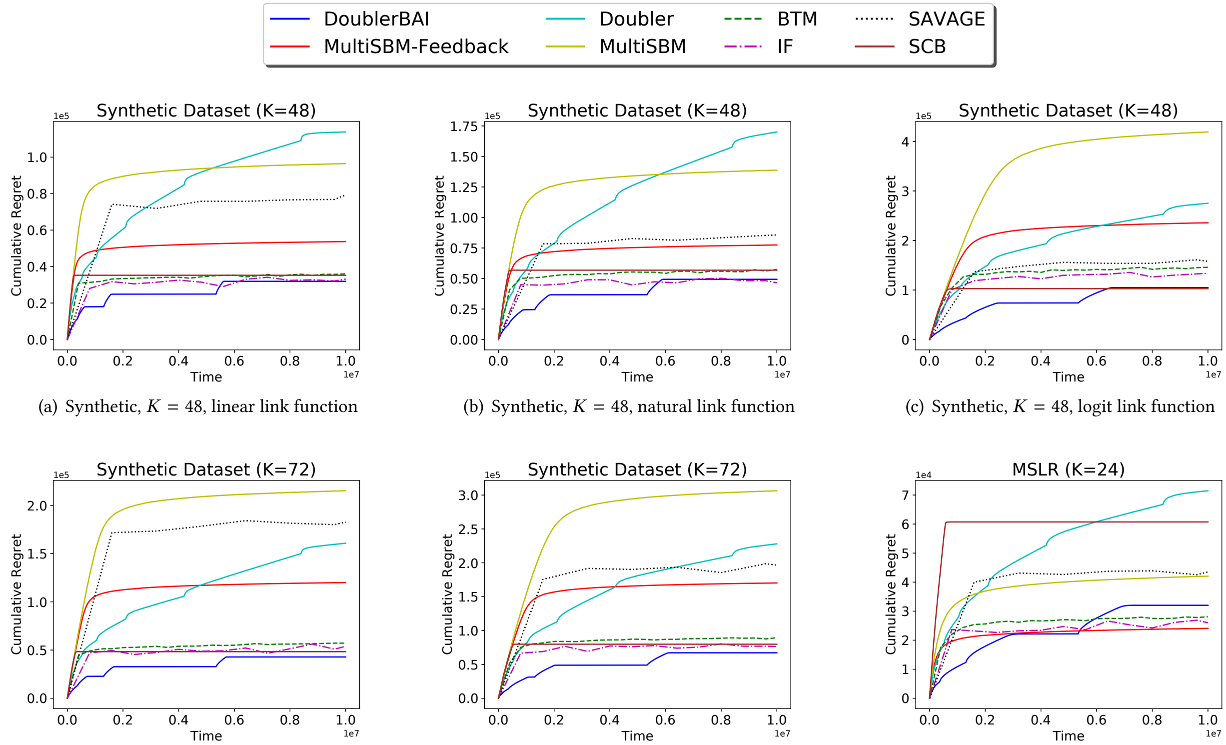
$$\Delta(x_1, x_i) \leq \gamma(\Delta(x_1, x_j) - \Delta(x_i, x_j)).$$

This property holds for a wide family of $\Delta(\cdot, \cdot)$. The main idea is that our analysis holds if the regret in the dueling bandits problem can be bounded by the regret seen by the BAIM (in Doubler) and SBM (in MultiSBM-Feedback) with some positive γ. The effect of γ on the regret bound of DoublerBAI and MultiSBM-Feedback is shown in the following corollaries:

Corollary 5.1. *Consider a K-armed two-dueling bandits game, in which $\Delta(\cdot, \cdot)$ satisfies Property 1 with parameter γ. Assume that the BAIM S in Line 1 of DoublerBAI has a sample complexity of $O(H \ln(\frac{H}{\delta}))$, where S outputs the best arm with probability at least $1-\delta$. Given an exponentially growing sequence $\{T_i\}_{i\in\mathbb{N}}$ of parameters $a, b > 1$ (i.e., $T_i = \lfloor a^{b^i} \rfloor$), the expected regret of DoublerBAI is bounded by*

$$\mathbb{E}[R_T] = O((H \ln H)^b) + O(H \ln T) + O(H \ln H \ln \ln T) + O(\ln \ln T),$$

where $H := \sum_{i=2}^{K} \frac{1}{\Delta_i^2}$ is the problem complexity for a bandit instance.



(a) Synthetic, $K = 48$, linear link function

(b) Synthetic, $K = 48$, natural link function

(c) Synthetic, $K = 48$, logit link function

(d) Synthetic, $K = 72$, linear link function

(e) Synthetic, $K = 72$, natural link function

(f) MSLR, $K = 24$

Figure 1: Regret results of two-dueling bandits on the synthetic (a-e) and MSLR (f) dataset.

Corollary 5.2. *Consider a K-armed two-dueling bandits game, in which $\Delta(\cdot,\cdot)$ satisfies Property 1 with parameter γ. The expected regret of MultiSBM-Feedback, which implements an SBM defined in Algorithm 3, is bounded by*

$$\mathbb{E}[R_T] \le \min\left\{\sum_{i>1}\frac{(\alpha+2)\Delta_{max}}{\Delta_i^2}\ln T, \sum_{i>1}\frac{2(\alpha+2)}{\Delta_i}\ln T\right\} + \frac{(\alpha+8)\Delta_{max}}{2\alpha}K + \sum_{j>1}\sum_{i>1} O\Big(\frac{\gamma\alpha\Delta_{max}}{\Delta_j^2}\big(\ln\ln T + \ln K + \ln(\frac{1}{\Delta_i})\big)\Big),$$

where $\Delta_{max} := \max_{i>1}\Delta_i$ and the confidence interval parameter $\alpha = \max\{3, \frac{\ln K}{\ln\ln T}\}$.

Note that γ does not affect the regret bound of DoublerBAI and the main term in the regret bound of MultiSBM-Feedback. This is because when fixing $x_j = x_1$ in Property 1, γ just vanishes and does not affect our analysis.

6 EXPERIMENTS

We conduct experiments for two-dueling bandits and multi-dueling bandits on both the synthetic and real-world datasets. In our synthetic datasets, the expected utilities of K arms are set as $\mu(x_1) = 0.8$ and $\mu(x_2), \ldots, \mu(x_K)$ forming a geometric sequence with $\mu(x_2) = 0.7, \mu(x_K) = 0.2$. Moreover, besides the linear link function, we also conduct experiments for two additional link functions, natural and logit, which are respectively defined as follows:

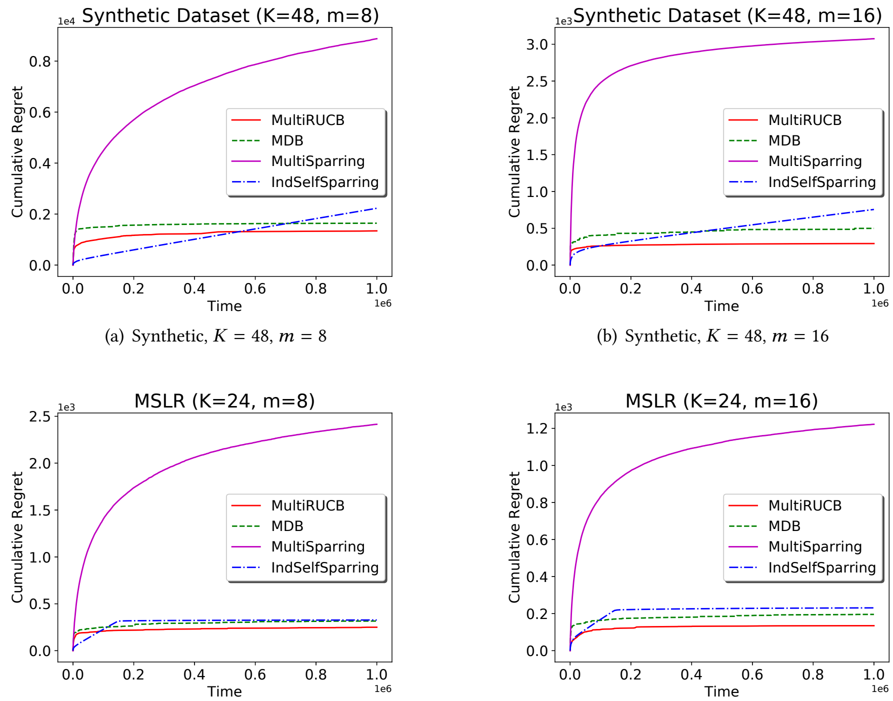
$$\phi_{\texttt{natural}}(\mu(x_i),\mu(x_j)) := \frac{\mu(x_i)}{\mu(x_i)+\mu(x_j)},$$

$$\phi_{\texttt{logit}}(\mu(x_i),\mu(x_j)) := \frac{1}{1+\exp\big(\mu(x_j)-\mu(x_i)\big)}.$$

For the real-world dataset, we use the Microsoft Learning to Rank (MSLR) dataset [17] in information retrieval, which contains query-document pairs labeled with relevance scores. Our setup follows that of [29], which estimates a preference matrix for 136 rankers. Each ranker can be regarded as an arm in dueling bandits since it is a function mapping a query of the user to a document ranking. We use a submatrix of 24 rankers selected from the full preference matrix. We remark here that the choice of $K = 24$ here is made to ensure a total order of the chosen arms (since their relations are obtained from a preference matrix), such that the existence of an optimal arm is guaranteed. The presented results are averaged over 50 independent runs for each algorithm.

6.1 Two-dueling Bandits Experiments

For the special case of our general setting, i.e., two-dueling bandits, we compare DoublerBAI and MultiSBM-Feedback with their baselines Doubler and MultiSBM [3], and other state-of-the-art algorithms including IF [27], BTM [28], SAVAGE [24] and SCB in [29]. For DoublerBAI, we choose the LUCB algorithm in [13] as the BAIM, and set parameters $a = 10,\ b = 1.1$. For the finite-horizon



(a) Synthetic, $K = 48$, $m = 8$

(b) Synthetic, $K = 48$, $m = 16$

(c) MSLR, $K = 24$, $m = 8$

(d) MSLR, $K = 24$, $m = 16$

Figure 2: Regret results of multi-dueling bandits on the synthetic (a-b) and MSLR (c-d) dataset. The results are obtained using the linear link function.

algorithms, IF, BTM and SAVAGE, we obtain each point of their regret curves by resetting the horizon to the corresponding time value. As shown in Fig. 1, DoublerBAI and MultiSBM-Feedback not only achieve significant improvements over their baselines, Doubler and MultiSBM, but also outperform the other state-of-the-art algorithms. In particular, compared to MultiSBM, MultiSBM-Feedback reduces the regret by approximately a half, which matches our theoretical analysis.

6.2 Multi-dueling Bandits Experiments

For the general multi-dueling bandit setting, we compare MultiRUCB with three state-of-the-art algorithms including MDB [8], IndSelfSparring [22] and MultiSparring (the multi-dueling extension of Sparring [3]). Fig. 2 plots the average cumulative regrets for 50 independent runs in the cases $m = 8$ and $m = 16$[3]. In addition, the variances of cumulative regrets at the 10^6 timestep corresponding to Fig. 2 (a-d) are also presented in Section D of the supplementary material [1] due to the space limit. The experimental results show that our MultiRUCB not only achieves the best regret performance, but also ensures the smallest variances among all the compared algorithms on both the synthetic and MSLR dataset. This demonstrates the superiority of MultiRUCB in practice, compared to existing algorithms for multi-dueling bandits. Moreover, among all the compared algorithms, MultiRUCB is the only algorithm possessing a finite-time analysis.

7 CONCLUSION

In this work, we study a general multi-dueling bandit problem, which has extensive real-world applications involving simultaneous duels of multiple options. For the special case of our setting, two-dueling bandits, we propose two efficient algorithms DoublerBAI and MultiSBM-Feedback, both achieving $O(\ln T)$ regret and outperforming existing algorithms. For the general multi-dueling bandits, we propose MultiRUCB and provide the first finite-time analysis for the problem. We prove that MultiRUCB achieves an $O(\ln T)$ regret. We also show that its regret improves as the capacity of the comparison set increases. Our experimental results based on both synthetic and real-world datasets demonstrate the performance superiority of our algorithms, compared to other state-of-the-art algorithms.

ACKNOWLEDGMENTS

The work is supported in part by the National Natural Science Foundation of China Grant 61672316, the Zhongguancun Haihua Institute for Frontier Information Technology and the Turing AI Institute of Nanjing.

[3]The results are similar for other m values.

REFERENCES

[1] AAMAS2020 Submission. Supplementary Material. https://www.dropbox.com/s/nzscfpdsj3ne099/supp_camready_0215.pdf?dl=0.
[2] Shipra Agrawal and Navin Goyal. 2012. Analysis of thompson sampling for the multi-armed bandit problem. In *Conference on Learning Theory*. 39–1.
[3] Nir Ailon, Zohar Karnin, and Thorsten Joachims. 2014. Reducing dueling bandits to cardinal bandits. In *Proceedings of the 31st International Conference on Machine Learning*. 856–864.
[4] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine learning* 47, 2-3 (2002), 235–256.
[5] Peter Auer and Ronald Ortner. 2010. UCB revisited: Improved regret bounds for the stochastic multi-armed bandit problem. *Periodica Mathematica Hungarica* 61, 1-2 (2010), 55–65.
[6] Lilian Besson and Emilie Kaufmann. 2018. What Doubling Tricks Can and Can't Do for Multi-Armed Bandits. *arXiv preprint arXiv:1803.06971* (2018).
[7] Brian Brost, Ingemar J Cox, Yevgeny Seldin, and Christina Lioma. 2016. An improved multileaving algorithm for online ranker evaluation. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 745–748.
[8] Brian Brost, Yevgeny Seldin, Ingemar J Cox, and Christina Lioma. 2016. Multi-dueling bandits and their application to online ranker evaluation. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. 2161–2166.
[9] Deepayan Chakrabarti, Ravi Kumar, Filip Radlinski, and Eli Upfal. 2009. Mortal multi-armed bandits. In *Advances in Neural Information Processing Systems*. 273–280.
[10] William Feller. [n. d.]. An introduction to probability theory and its applications. *1957* ([n. d.]).
[11] Katja Hofmann, Shimon Whiteson, and Maarten de Rijke. 2013. Balancing exploration and exploitation in listwise and pairwise online learning to rank for information retrieval. *Information Retrieval* 16, 1 (2013), 63–90.
[12] Kojiro Iizuka, Takeshi Yoneda, and Yoshifumi Seki. 2019. Greedy optimized multileaving for personalization. In *Proceedings of the 13th ACM Conference on Recommender Systems*. ACM, 413–417.
[13] Shivaram Kalyanakrishnan, Ambuj Tewari, Peter Auer, and Peter Stone. 2012. PAC Subset Selection in Stochastic Multi-armed Bandits.. In *Proceedings of the 29th International Conference on Machine Learning*, Vol. 12. 655–662.
[14] Pushmeet Kohli, Mahyar Salek, and Greg Stoddard. 2013. A fast bandit algorithm for recommendations to users with heterogeneous tastes. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence*. 1135–1141.
[15] Harrie Oosterhuis and Maarten de Rijke. 2017. Sensitive and scalable online evaluation with theoretical guarantees. In *Proceedings of the 26th ACM on Conference on Information and Knowledge Management*. ACM, 77–86.
[16] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank citation ranking: Bringing order to the web.* Technical Report. Stanford InfoLab.
[17] Tao Qin and Tie-Yan Liu. 2013. Introducing LETOR 4.0 Datasets. *CoRR* abs/1306.2597 (2013). http://arxiv.org/abs/1306.2597
[18] Razieh Rahimi and Azadeh Shakery. 2017. Online learning to rank for cross-language information retrieval. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 1033–1036.
[19] Stephen E Robertson and Steve Walker. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR'94*. Springer, 232–241.
[20] Anne Schuth, Katja Hofmann, and Filip Radlinski. 2015. Predicting search satisfaction metrics with interleaved comparisons. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 463–472.
[21] Yanan Sui and Joel Burdick. 2014. Clinical online recommendation with subgroup rank feedback. In *Proceedings of the 8th ACM Conference on Recommender Systems*. ACM, 289–292.
[22] Yanan Sui, Vincent Zhuang, Joel W. Burdick, and Yisong Yue. 2017. Multi-dueling Bandits with Dependent Arms. In *Proceedings UAI, Conference on Uncertainty in Artificial Intelligence*.
[23] William R Thompson. 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika* 25, 3/4 (1933), 285–294.
[24] Tanguy Urvoy, Fabrice Clerot, Raphael Féraud, and Sami Naamane. 2013. Generic exploration and k-armed voting bandits. In *Proceedings of the 30th International Conference on Machine Learning*. 91–99.
[25] Sofía S Villar, Jack Bowden, and James Wason. 2015. Multi-armed bandit models for the optimal design of clinical trials: benefits and challenges. *Statistical science: a review journal of the Institute of Mathematical Statistics* 30, 2 (2015), 199.
[26] Huazheng Wang, Ramsey Langley, Sonwoo Kim, Eric McCord-Snook, and Hongning Wang. 2018. Efficient exploration of gradient space for online learning to rank. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 145–154.
[27] Yisong Yue, Josef Broder, Robert Kleinberg, and Thorsten Joachims. 2012. The k-armed dueling bandits problem. *J. Comput. System Sci.* 78, 5 (2012), 1538–1556.
[28] Yisong Yue and Thorsten Joachims. 2011. Beat the mean bandit. In *Proceedings of the 28th International Conference on Machine Learning*. 241–248.
[29] Masrour Zoghi, Zohar S Karnin, Shimon Whiteson, and Maarten De Rijke. 2015. Copeland dueling bandits. In *Advances in Neural Information Processing Systems*. 307–315.
[30] Masrour Zoghi, Shimon Whiteson, Remi Munos, and Maarten De Rijke. 2014. Relative upper confidence bound for the K-armed dueling bandit problem. In *Proceedings of the 31st International Conference on Machine Learning*. II–10.