Dueling Bandits with Dependent Arms

Bangrui Chen
ORIE, Cornell
Ithaca, NY 14850
bc496@cornell.edu

Peter Frazier
ORIE, Cornell
Ithaca, NY 14850
pf98@cornell.edu

Abstract

We consider online content recommendation with implicit feedback through pairwise comparisons. We study a new formulation of the dueling bandit problems in which arms are dependent and regret occurs when neither pulled arm is optimal. We propose a new algorithm, Comparing The Best (CTB), with computational requirements appropriate for problems with few arms, and a variation of this algorithm whose computation scales to problems with many arms. We show both algorithms have constant expected cumulative regret. We demonstrate through numerical experiments on simulated and real dataset that these algorithms improve significantly over existing algorithms in the setting we study.

1 Introduction

We consider bandit learning in personalized content recommendation with implicit pairwise comparisons. In our model, we offer pairs of items to a user and record implicit feedback on which offered item is preferred, seeking to learn the user’s preferences over items quickly, while also ensuring that the fraction of time we fail to offer a high-quality item in our offered assortment is small. Implicit pairwise comparisons avoid the inaccuracy of user ratings [1] and the difficulty of engaging users in providing explicit feedback. We model items as related through covariates, and model regret as depending on the utility to the user of the best offered item.

For example, consider a recommender system that delivers news articles to subscribers through push notifications. Each news item is available through several different news sources, and so when we notify users of news, we may offer two different versions of the same news item from different sources, and record which one the user selects, so as to determine the user’s favored news source.

The problem that we study is a variant of the dueling bandit problem [2], in which a sequence of rounds are played, and in each round we forward two arms to the user and receive noisy feedback from the user telling us which arm is preferred. When an arm is preferred in a round, we say that arm “wins the duel” in that round.

Previous work on dueling bandits has focused on choosing between ranking algorithms [3], where query results from two rankers are interleaved [4], and the ranking algorithm that provided the first result chosen by the user is declared the winner of the duel. Perhaps because of their focus on the ranking setting, previous work has focused on models that are appropriate for few arms that do not model the dependence between arms, and where zero regret requires pulling the best arm twice.

We study a formulation of the dueling bandits problem with two differences from the standard setting that make it more appropriate for online content recommendation. First, we model the dependence between arms by endowing arms with covariates which are then modeled as interacting through a known function (for example, the inner product) with some latent vector describing the user’s preferences, providing the user’s utility for that arm. For example, in our example above, covariates for a news source could be determined by fitting a topic model to the text in past news items from that source, and the user’s preference vector could be interpreted as the user’s preference for each
At time $t$, the system chooses two items and forwards them to the user. We denote the index of these work well in high dimension with a large number of arms, which is called Scalable-CTB (SCTB).

Comparing The Best (CTB) which works well for few arms and in section 5 we prove CTB has constant expected cumulative regret. In section 6 we propose a scalable version of CTB that could work well in high dimension with a large number of arms, which is called Scalable-CTB (SCTB). In section 7 we compare CTB and SCTB with three benchmarks using both simulated and real dataset. The numerical results suggest CTB and SCTB outperform all benchmarks we considered.

The paper is structured as follows. In section 3 we formulate our problem. In section 4 we introduce Comparing The Best (CTB) which works well for few arms and in section 5 we prove CTB has constant expected cumulative regret. In section 6 we propose a scalable version of CTB that could work well in high dimension with a large number of arms, which is called Scalable-CTB (SCTB). In section 7 we compare CTB and SCTB with three benchmarks using both simulated and real dataset. The numerical results suggest CTB and SCTB outperform all benchmarks we considered.

## 2 Related Work

Most work on dueling bandits focuses on the case that arms are independent, and zero regret occurs only when both of the pulled arms are optimal. Under these assumptions, [5] shows that the lower bound for the expected cumulative regret up to time $T$ is $O(N \log(T))$. Under the assumption that a Condorcet winner (an arm that has winning probability greater or equal to $\frac{1}{2}$ versus any other arms) exists, algorithms were proposed that can reach the optimal regret bound under different conditions in the finite horizon setting, such as Interleaved Filter (IF) [5] and Beat the Mean (BTM) [6]. [7] proposed Relative Upper Confidence Bound (RUCB), which is the first algorithm that can reach the optimal regret bound in the horizonless setting. Recent research focuses on the case where a Condorcet winner does not exist. [8] proposed two algorithms, Copeland Confidence Bound (CCB) and Scalable Copeland Bandits (SCB), which have an optimal regret bound without the assumption that a Condorcet winner exists. Because they model arms as independent, these algorithms all require a large number of iterations before they can find the best arm, especially with many arms.

In contrast, our paper focuses on the setting where arms are dependent and regret is only related to the best arm shown. As far as we know, no other previous work on dueling bandits studies this setting. Outside of the dueling bandit literature, in the literature on active learning, [9] considers an active learning problem that lacks the bandit objective based on cumulative regret but is otherwise similar. The primary purpose of [9] is to sort the arms based on the user’s preference. It proposes a novel algorithm, the Query Selection Algorithm (QSA), with the property that, when the number of arms $N$ is sufficiently large, the expected number of operations to sort $N$ arms in the noiseless case is $d \log(N)$ ($d$ is the dimension of the space) instead of $N \log(N)$. They also show a high probability result in the noisy case.

## 3 Problem Formulation

Denote the user’s latent preference vector as $\theta \in \mathbb{R}^d$. There are $N$ ($N \geq 2$) distinct arms lying in the $d$-dimensional space, denoted as $A_1, \cdots, A_N$. The user prefers $A_i$ over $A_j$ if and only if $u(\theta, A_i) > u(\theta, A_j)$, where $u(\cdot, \cdot)$ is a known utility function. We assume that arms are ordered by the user’s preferences, $u(\theta, A_1) > u(\theta, A_2) > \cdots > u(\theta, A_N)$. Throughout this paper, we use the terms “arm $A_i$” and “item $A_i$” interchangeably.

At time $t$, the system chooses two items and forwards them to the user. We denote the index of these two arms at time $t$ as $X_{t,1}$ and $X_{t,2}$. Suppose at time $t$, we forward item $A_i$ and $A_j$ to the user and the user prefers $A_i$ over $A_j$. Then the user chooses his/her favourite based on the following manner: the user chooses item $A_i$ with probability $p_{i,j} > 0.5$ and item $A_j$ with probability $1 - p_{i,j}$. The system can observe the user’s feedback $Y_t \in \{-1, +1\}$, where $Y_t = 1$ means the user prefers $A_{X_{t,1}}$, and $Y_t = -1$ means the user prefers $A_{X_{t,2}}$. Here we denote $p = \min_{i<j} p_{i,j}$, which is the lower bound of the probability that the user will choose his/her favourite item. Based on its definition, we know $p > 0.5$. 


We briefly discuss two discrete choice models that can take advantage of our framework. The first model is logit model or Bradley-Terry model. In this case, the utility function is \( u(\theta, A_i) = \theta \cdot A_i \) and the user will choose arm \( A_i \) over \( A_j \) with probability \( p_{i,j} = \frac{\exp(u(\theta, A_i))}{\exp(u(\theta, A_i)) + \exp(u(\theta, A_j))} \). Another example is to model the user’s choice with probit model. In this case, the utility function is still the inner product, but \( p_{i,j} = \Phi(u(\theta, A_i) - u(\theta, A_j)) \) where \( \Phi(\cdot) \) is the cdf for a standard normal distribution given the difference between the two utilities follows a standard normal distribution.

The regret \( r(t) \) at time \( t \) is defined as \( r(t) = u(\theta, A_{\text{best}}) - \max \{ u(\theta, A_1), \ldots, u(\theta, A_N) \} \), which is the difference in expected utility between the best arm overall and the best arm that the user can choose from those offered. Since we think of the user as receiving the utility of the single arm chosen, we measure with respect to the best available arm, rather than some other function of the two offered arms. The cumulative regret up to time period \( T \) is defined as: \( R(T) = \sum_{t=1}^{T} r(t) \).

We measure the quality of an algorithm by its expected cumulative regret. Now, we develop two algorithms CTB and SCTB, and show they have constant expected cumulative regret.

### 4 Methods

In this section, we propose an algorithm Comparing The Best (CTB) for this problem setting.

**Definition 4.1.** Each pair of arms \( A_i \) and \( A_j \) uniquely decides a winning space \( H_{i,j} := \{ X \in \mathbb{R}^d : u(X, A_i) \geq u(X, A_j) \} \).

Based on the definition of winning space, we know when \( \theta \in H_{i,j} \), the user with this \( \theta \) will prefer item \( A_i \) over \( A_j \). When the utility function is a linear function, this winning space is a half space. Further \( H_{i,j} \) and \( H_{j,i} \) are different winning spaces and their union is \( \mathbb{R}^d \). Throughout this paper, we use the phrases “arm \( A_i \) wins over arm \( A_j \) in a duel”, and “winning space \( H_{i,j} \) wins the duel” interchangeably.

Each pair of arms determines two winning spaces and all winning spaces partition the space \( \mathbb{R}^d \) into a number of cells. Each cell can be written as intersection of winning spaces. To formally define a cell, we first denote \( H^k_{i,j} = H_{i,j} \) when \( k = 0 \) and \( H^k_{i,j} = H_{j,i} \) when \( k = 1 \). Then for a binary vector \( V \), we denote the \( k \)th element of \( V \) as \( V[k] \). Finally, we define a cell as:

**Definition 4.2.** The cell \( C \) corresponding to a \( \frac{N(N-1)}{2} \) length binary vector \( V \) is defined as

\[
C := \cap_{k<j} H^{1/(2N-1)}_{i,j}:|i-j| \cdot \frac{2(2N-1)(i-1)}{2} \cdot j-i] .
\]

In this definition, we use the \( \lceil \frac{(2N-1)(i-1)}{2} \right \rceil + j-i \)th element of \( V \) to denote whether \( C \) belongs to \( H_{i,j} \) or \( H_{j,i} \), for \( i < j \). In this paper, we use \( C(V) \) to denote a cell that is defined by binary vector \( V \).

Consider a sequence of binary vectors, all of length \( \frac{N(N-1)}{2} \) that are ordered lexicographically. Denote \( V_k \) the \( k \)th binary vector corresponding to a non-empty cell. Denote the number of non-empty cells as \( M \), and denote the cells as \( C_1, C_2, \ldots, C_M \) where \( C_i \) is corresponding to \( V_i \). With this definition, \( C_1 \) is the cell that corresponding to \( V_1 = [0, 0, \ldots, 0] \) and thus \( C_1 = \cap_{i<j} H_{i,j} \). Further we know \( C_1 \) is the cell that contains \( \theta \). This is because for any arm \( A_i \) and \( A_j \) with \( i < j \), since the user prefers \( A_i \) over \( A_j \), we know \( u(\theta, A_i) > u(\theta, A_j) \), which means \( \theta \in H_{i,j} \). Denote \( J_k = \{ (i, j) | C_k \subseteq H_{i,j} \} \), which is the collection of index of the winning spaces that contains \( C_k \). Figure 1 is an illustration of winning spaces and cells.

For each cell \( C_i \), denote the number of times that a winning space contains \( C_i \) wins a duel before time \( t \) as \( m_i(t) \). Formally,

\[
m_i(t) = \sum_{k=1}^{t} \mathbb{1}\{Y_k = -1, C_i \subseteq H_{X_k,1}, X_{k,2} \} + \sum_{k=1}^{t} \mathbb{1}\{Y_k = 1, C_i \subseteq H_{X_k,2}, X_{k,1} \} .
\]

Each cell \( C_i \) assigns a preference order to the arms. Denote \( B(i) \) as the best arm suggested by cell \( i \). Formally, \( B(i) \) is the unique arm index \( j \) such that \( C_i \subseteq H_{j,k}, \forall k \neq j \). Since \( C_1 \) is the cell that contains the true \( \theta \), we know \( B(1) = 1 \).
Figure 1: Illustration of winning spaces and cells. The index of the cell and its corresponding binary vectors are: $C_1$ and $(0, 0, 1)$; $C_2$ and $(0, 1, 0)$; $C_3$ and $(0, 1, 1)$; $C_4$ and $(0, 1, 1)$; $C_5$ and $(1, 0, 0)$; $C_6$ and $(1, 1, 0)$; $C_7$ and $(1, 1, 1)$.

Here we propose an algorithm *Comparing The Best* (CTB), which selects two arms that are “most likely” to be the best arm and forwards them to the user.

```
for $t \leq T$ do
    Step 1: Denote $i_t = \text{arg max}_i m_i(t)$, break ties arbitrarily;
    Step 2: Pick $X_{t,1} = B(i_t)$;
    Step 3: Denote $j_t = \text{arg max}_j B(j) \neq B(i_t) m_j(t)$, break ties arbitrarily;
    Step 4: Pick $X_{t,2} = B(j_t)$;
    Step 5: Observe the noisy feedback $Y_t$ and update $m_i(t)$ based on Equation [1];
    Step 6: $t = t + 1$;
end
```

**Algorithm 1: Comparing The Best (CTB)**

CTB stores a value for every cell, which can require an excessive amount of memory for problems with many arms. We introduce a scalable version of CTB in Section 6, but we first prove that CTB has constant expected regret.

### 5 Theoretical Results

In this section, we prove the expected cumulative regret of CTB is bounded by a constant.

The main idea behind our proof is to show that for each cell $C_i$ with $B(i) \neq 1$, $E[\sum_{t=0}^{\infty} 1\{m_i(t) \geq m_1(t)\}]$ is bounded by a constant. We show this in turn by showing $m_1(t) - m_i(t)$ can be modeled by a random walk with a larger probability of increasing than of decreasing. The following lemma, whose proof is in the supplement, allows us to bound the number of times this stochastic process takes negative values.

**Lemma 1.** For any $0.5 < p \leq 1$, suppose $Z(t)$ is a stochastic process with filtration $\mathcal{F}_t$, $Z(0) = 0$ and $P(Z(t+1) = Z(t) + 1|\mathcal{F}_t) \geq p$ and $P(Z(t+1) = Z(t) - 1|\mathcal{F}_t) = 1 - P(Z(t+1) = Z(t) + 1|\mathcal{F}_t)$, then we have $E[\sum_{t=0}^{\infty} 1\{Z(t) \leq 0\}] \leq \frac{p}{(2p-1)p}$.

We now proceed with the larger proof by defining

$$q_{i,j}(t) = \sum_{k=1}^{t} \mathbb{1}\{X_{k,1} = i, X_{k,2} = j, Y_k = -1\} + \sum_{k=1}^{t} \mathbb{1}\{X_{k,1} = j, X_{k,2} = i, Y_k = 1\},$$

(2)

which is the number of times that we forward $A_i$ and $A_j$ to the user and the user prefers $A_i$ over $A_j$ up to time t. Then we can rewrite $m_i(t)$ in terms of $q_{i,j}(t)$:

$$m_i(t) = \sum_{(i,j) \in J_k} q_{i,j}(t).$$

(3)
Based on the definition of $C_1$, we know $J_1 = \{(i,j), \forall i < j\}$ and $m_1(t) = \sum_{i<j} q_{i,j}(t)$. Denote $N_{i,j}(t)$ as the number of times that the system forwards arm $A_i$ and $A_j$ to the user. The next lemma shows $E[N_{i,j}(t)]$ is bounded by a constant for $1 < i < j$.

**Lemma 2.** For $1 < i < j$, we have $E[N_{i,j}(t)] \leq M \frac{p}{(2p-1)^2}$, where $M$ is the total number of non-empty cells.

**Proof.** For $1 < i < j$, we can only pull arm $A_j$ when there is cell $C_s$ which believes $A_j$ is the best arm and its corresponding $m_s(t)$ is greater or equal to $m_1(t)$. Thus we have

$$N_{i,j}(t) \leq \sum_{k=1}^{t} \mathbb{1}\{\max_{s:B(s)=j} m_s(k) \geq m_1(k)\}$$

$$\leq \sum_{s:B(s)=j} \sum_{k=1}^{t} \mathbb{1}\{m_s(k) \geq m_1(k)\}$$

$$\leq \sum_{s:B(s)=j} \sum_{k=1}^{t} \mathbb{1}\{\sum_{(i',j') \in J_s} q_{i',j'}(k) \geq \sum_{(i',j') \in J_1} q_{i',j'}(k)\}$$

$$= \sum_{s:B(s)=j} \sum_{k=1}^{t} \mathbb{1}\{\sum_{(i',j') \in J_s \setminus J_1} q_{i',j'}(k) \geq \sum_{(i',j') \in J_1 \setminus J_s} q_{i',j'}(k)\}$$

$$= \sum_{s:B(s)=j} \sum_{k=1}^{t} \mathbb{1}\{\sum_{(i',j') \in J_s \setminus J_1} [q_{i',j'}(k) - q_{j',i'}(k)] \geq 0\},$$

where the last equation is because $J_1 = \{(i',j') : i' < j'\}$ and thus $(i',j') \in J_s \setminus J_1 \iff (j',i') \in J_1 \setminus J_s$.

Denote $Z(k) = \sum_{(i',j') \in J_s \setminus J_1} [q_{i',j'}(k) - q_{j',i'}(k)]$. Then $Z(k)$ is very “close” to a random walk:

- If we forward arm $A_{i_0}$ and $A_{j_0}$ to the user where $i_0 < j_0$ and $(j_0,i_0) \in J_s \setminus J_1$, then $Z(k+1) = Z(k) + 1$ with probability $1 - p_{i_0,j_0}$ and $Z(k+1) = Z(k) - 1$ with probability $p_{i_0,j_0} \geq p$, which is independent of previous history. Further, since $(j,i) \in J_s \setminus J_1$, every time we forward $A_i$ and $A_j$ to the user, $Z(k)$ will not remain the same.

- If we forward arm $A_{i_0}$ and $A_{j_0}$ to the user where $i_0 < j_0$ and $(j_0,i_0) \notin J_s \setminus J_1$, then $Z(k+1) = Z(k)$.

Define $\tau_0 = 0$, $\tau_t = \min_k \{k > \tau_{t-1}, Z(k) \neq Z(k-1)\}$, for $t = 1, 2, \ldots$. Because $\tau_t$ is a non-decreasing right continuous stopping time, we know it is a valid random change of time [12]. Define $\zeta = \inf\{t : \tau_t = \infty\}$, which is the life time of the random change of time. Here we consider a new stochastic process $W(t)$ defined as $W(t) = Z(\tau_t)$ for $t < \zeta$. For all sample path that $\zeta < \infty$, we define $P(W(t) = W(t-1) + 1|W(1), \cdots, W(t-2)) = 1 - p$ and $P(W(t) = W(t-1) - 1|W(1), \cdots, W(t-2)) = p$ for all $t > \zeta$. Denote the filtration for $W(t)$ up to time $t$ as $\mathcal{F}_t$. Then:

- For $t \leq \zeta$, we have $P(W(t) = W(t-1) - 1|\mathcal{F}_{t-1}) = P(Z(\tau_t) = Z(\tau_{t-1}) - 1|\mathcal{F}_{t-1}) = P(Z(\tau_t) = Z(\tau_{t-1} - 1|\mathcal{F}_{t-1}) \geq p$.

- For $t > \zeta$, based on its definition, we know $P(W(t) = W(t-1) - 1|\mathcal{F}_{t-1}) = p$.

Thus, we know $W(t)$ is well defined and $P(W(t) = W(t-1) + 1|\mathcal{F}_{t-1}) \leq 1 - p$, $P(W(t) = W(t-1) - 1|\mathcal{F}_{t-1}) \geq p$ and we know $p > 0.5$. Since every time we forward arm $A_i$ and $A_j$ to the
user, $W(k) \geq 0$ must hold, thus

$$E[N_{i,j}(t)] \leq E \left[ \sum_{B(s) = j} \sum_{k=1}^{t} \sum_{(i',j') \in J_s \setminus J_t} [q_{i',j'}(k) - q_{j',i}(k)] \geq 0 \right],$$

$$\leq \sum_{s:B(s) = j} \left[ E \left[ \sum_{k=1}^{\infty} 1\{W(k) \geq 0\} \right] \right] \leq M \frac{p}{(2p - 1)^2}.$$  

Finally, denote $\Lambda = u(\theta, A_1) - u(\theta, A_N)$. Based on Lemma 2 and a union bound, we obtain our main theorem, which is

**Theorem 3.** The expected cumulative regret for CTB is bounded by $\frac{(N-1)(N-2)}{2} M \frac{p}{(2p-1)^2} \Lambda$.

Since each cell assigns a ranking for arms and different cells give us different rankings, we can bound $M$ by $N!$, which is the number of all permutations. When the utility function is linear, then based on the results in [9], we know $M$ is $O(N^{2d})$ for sufficiently large $N$.

6 Scalable-CTB (SCTB)

CTB achieves a constant expected cumulative regret. However, it requires a great deal of memory to store $m_i(t)$ for each cell, which makes it computationally infeasible for problems with many arms. Instead of saving $m_i(t)$, Equation 3 suggests that we can save $q_{i,j}(t)$ and use $q_{i,j}(t)$ to reconstruct $m_i(t)$. Based on this idea, we propose a new algorithm, SCTB that that has performance comparable to CTB, and that can work on much larger-scale problems.

In SCTB, we are going to set up Step 1 and Step 3 in CTB as a optimization problem in terms of $q_{i,j}(t)$. Denote $e_{i,j}$ as a binary variable where $e_{i,j} = 1$ means the cell is in $H_{i,j}$ and 0 otherwise, then based on Equation 3, maximizing $m_i(t)$ is equivalent to maximizing $\sum_{i,j: i \neq j} e_{i,j} \times q_{i,j}(t)$ subject to the constraint that the intersection of the winning spaces is not empty. In general, these constraints are not easy to describe in a way that admits optimization. Instead, in SCTB, we introduce imaginary cells that simplify the constraints. “Imaginary cells” are cells that correspond to a valid collection of binary variables, and a valid intersection of winning spaces, but that are empty.

**Definition 6.1.** Cell $C(V)$ is an imaginary cell if $\cap_{i<j} H_{i,j}^{(2(N-i)(i-1)+j-i)} = \emptyset$.

To find the best arm suggested by $\arg \max_i m_i(t)$, in SCTB, we first find $\max_{i:B(i) = A_k} m_i(t)$ for each arm $A_k$ after introducing the imaginary cells, which is the cell with largest $m_i(t)$ such that this cell believes $A_k$ is the best. The $k^{th}$ problem is defined as follows:

$$\text{maximize} \sum_{i,j: i \neq j} e_{i,j} \times q_{i,j}(t)$$

subject to $e_{k,j} = 1$, $\forall j \neq k$

$$e_{i,j} + e_{j,i} = 1, i, j = 1, ..., N, i \neq j$$

$$e_{i,j} \in \{0, 1\}, \forall i \neq j$$

For the $k^{th}$ problem, there are three conditions in Equation 4. The first condition says $e_{k,j} = 1$ $\forall j \neq k$, which means cell $C_t$ that satisfies the first condition must lie in the winning space $H_{k,j}$, $\forall j \neq k$. In other words, $C_t$ ranks arm $A_k$ better than any others and thus $B(t) = k$. The second and third condition together guarantees that cell $C_t$ either belongs to $H_{i,j}$ or $H_{j,i}$. These three constraints will introduce some imaginary cells into the feasible solutions. However, this will not hurt our algorithm since $\theta$ lies in the cell $C_t$ and eventually $m_1(t)$ will be the largest.

Though Equation 4 looks like an integer linear programming, it is in fact very easy to solve: the maximum value of this problem is reached when $e_{i,j} = 1$ if $q_{i,j}(t) \geq q_{j,i}(t)$ for all $i \neq j$, $e_{i,j} = 0$ otherwise. Denote the maximum value of the $k^{th}$ problem at time $t$ as $f(k,t)$.

After knowing $f(k,t) = \max_{B(i) = k} m_i(t)$, then finding the best arm with largest $m_i(t)$ is equivalent to finding $\arg \max_k f(k,t)$. Now we are ready to introduce the Scalable-CTB (SCTB). It is the same
as CTB except we have introduced imaginary cells, simplifying the constraints in Equation 4. The
SCTB algorithm is as follows.

for \( t \leq T \) do
\( \)
Step 1: Calculate \( f(k, t) \) for \( k = 1, 2, \ldots, N \);
Step 2: Denote \( i_t = \arg \max_k f(k, t) \), break ties arbitrarily;
Step 3: Pick \( X_{t, 1} = i_t \);
Step 4: Denote \( j_t = \arg \max_k \neq i_t f(k, t) \), break ties arbitrarily;
Step 5: Pick \( X_{t, 2} = j_t \);
Step 6: Observe the noisy feedback \( Y_t \) and update \( q_{i, j}(t) \) based on Equation 2;
Step 7: \( t = t + 1 \);
end

Algorithm 2: Scalable CTB (SCTB)

**Theorem 4.** The expected cumulative regret for SCTB is bounded by

\[
\frac{(N-1)(N-2)}{2} 2^\frac{N(N-1)}{2} \frac{p}{(2p-1)^p} \Lambda.
\]

The proof is similar to Theorem 3 which is in the supplement.

7 Numerical Experiments

In this section, we compare CTB and SCTB with three benchmarks: Thompson Sampling, Relative Upper Confidence Bound (RUCB) and the Query Selection Algorithm (QSA) [9].

Thompson sampling uses a posterior distribution over the cell containing theta, and this posterior is computed by beginning with a prior distribution on the location of \( \theta \), and updating this prior using Bayes rule and knowledge of \( p_{i, j} \). At time \( t \), we generate \( \theta_t \) from the posterior distribution \( p_t \) and forward the two arms that \( \theta_t \) ranks the top two to the user and receives the user’s feedback. In our implementation, we track the prior/posterior explicitly by storing a probability for each cell.

An exact implementation of Thompson sampling scales poorly with the number of cells and so we only compare against Thompson sampling in the setting with fewer arms (see Section 7.1 where the prior on theta is uniform over the unit sphere). It may also be possible to implement Thompson sampling using Gibbs sampling although this would be substantially more complicated than SCTB. We also emphasized that Thompson sampling as we consider it here requires knowledge of \( p_{i, j} \) which is not typically not available.

We choose RUCB as our benchmark because it works well when a Condorcet winner exists, and existence of a Condorcet winner is a consequence of our modeling assumptions. Though there are algorithms that claim to outperform RUCB such as CCB and SCB [9], they mainly work better in the case where a Condorcet winner does not exist.

Throughout this section, we assume the utility function is \( u(\theta, A_i) = \theta \cdot A_i \). In the first two examples with simulated data, both the arms and the user’s preference vector \( \theta \) are uniformly generated from the unit sphere. Also, we assume \( p_{i, j} = 0.8 \) for all \( i < j \). In the Yelp academic dataset, we use the probit model to model the user’s choice.

7.1 Simulated Dataset with Small Number of Arms in Low Dimensional Space

In this section, we compare CTB and SCTB with Thompson Sampling, RUCB and QSA using 20 arms in 2-dimensional space. Figure 2 shows that CTB is the best algorithm, which means that if we can store \( m_i(t) \) for all cells, then CTB typically beats Thompson Sampling. Though SCTB did not perform as well as CTB and Thompson Sampling, it outperforms QSA and RUCB. SCTB has an advantage over Thompson Sampling and CTB in that it can work on a much larger-scale problem.

7.2 Simulated Dataset with Large Number of Arms in High Dimensional Space

In this section, we compare SCTB with RUCB and QSA on a problem with 50 arms in 20-dimensional space. As can be seen from the Figure 3a, RUCB almost has a linear regret when the time is short. This is in general true in the dueling bandits literature, which typically requires \( 10^4 \) iterations before it starts to achieve \( \log(T) \) cumulative regret for 50 arms. Our algorithm finds the optimal arm after roughly 500 iterations and reaches constant expected cumulative regret.
Figure 2: Compare CTB and SCTB with Thompson Sampling, RUCB and QSA when the number of arms is small and dimension is low. In this case, CTB outperforms Thompson Sampling and both of them outperform SCTB. SCTB outperforms RUCB and QSA, and can be applied to a much larger-scale problem, which is difficult for CTB and (our implementation of) Thompson Sampling.

Figure 3: Compare the performance of SCTB, RUCB and QSA using simulated dataset as well as Yelp academic dataset. In both simulations, SCTB outperforms RUCB and QSA. Further, the expected cumulative regret of SCTB is a constant.

7.3 Yelp Academic Dataset

In this section, we compare SCTB with RUCB and QSA using the Yelp academic dataset [14]. In this experiment, we choose 100 restaurants from Las Vegas as our arms. For each restaurant, we represent it by a 20-dim feature vector, calculated using doc2vec [15] on its review. We select 49 users who have reviewed at least 20 of these 100 restaurants and calculate their preference vectors using linear regression. Denote the estimated variance of the residuals from the linear regression as $\hat{\sigma}^2$.

Since for each user, there are some restaurants that he/she hasn’t reviewed, we rank the preference for restaurants based on the inner product between the user’s preference vector and the restaurant’s feature vector. For restaurant i and restaurant j with feature vector $A_i$ and $A_j$, the user with preference vector $\theta$ will choose restaurant i with probability $\Phi(\theta \cdot A_i - \theta \cdot A_j)$, where $\Phi(\cdot)$ is the cdf for the normal distribution with mean 0 and variance $2\hat{\sigma}^2$.

The result is summarized in Figure 3(b). SCTB outperforms both RUCB and QSA significantly. RUCB has a linear regret when the time is short.

8 Conclusion

In this paper, we consider dueling bandits for online content recommendation. We formulate a new setting which differs from the traditional dueling bandits where arms are dependent, and regret depends only on the best arm shown. We propose two algorithms CTB and SCTB which are proved to have constant expected cumulative regret. Numerical experiments suggest our algorithms outperform all existing algorithms considered, in both simulated and real dataset.
References

[1] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, Filip Radlinski & Geri Gay, Evaluating The Accuracy Of Implicit Feedback From Clicks And Query Reformulations In Web Search, ACM Transactions on Information Systems (TOIS), 2007

[2] Yisong Yue & Thorsten Joachims, Interactively Optimizing Information Retrieval Systems As A Dueling Bandits Problem, ICML, 2009

[3] Katja Hofmann, Shimon Whiteson & Maarten de Rijke. Fidelity, soundness, and efficiency of interleaved comparison methods. ACM Transactions on Information Systems (TOIS), 31(4), 2013

[4] Filip Radlinski, Madhu Kurup & Thorsten Joachims, How Does Clickthrough Data Reflect Retrieval Quality? ACM Conference on Information and Knowledge Management (CIKM) (pp. 43–52), 2008.

[5] Yisong Yue, Josef Broder, Robert Kleinberg & Thorsten Joachims, The K-armed Dueling Bandits Problem, Journal Of Computer And System Sciences, 2012

[6] Yisong Yue & Thorsten Joachims, Beat The Mean Bandit, ICML, 2011

[7] Masrour Zoghi, Shimon Whiteson, Remi Munos & Maarten De Rijke, Relative Upper Confidence Bound For The K-Armed Dueling Bandit Problem, ICML, 2014

[8] Masrour Zoghi, Zohar Karnin, Shimon Whiteson & Maarten De Rijke, Copeland Dueling Bandits, NIPS, 2015

[9] Kevin Jamieson & Robert Nowak, Active Ranking using Pairwise Comparisons, NIPS, 2011

[10] David Revelt & Kenneth Train, Mixed logit with repeated choices: households’ choices of appliance efficiency level, The Review of Economics and Statistics, 1998.

[11] Philip Hans Franses & A. L. Montgomery, Econometric Models in Marketing, 2002.

[12] Ole Eiler Barndorff-Nielsen & Albert Shiryaev, Change of time and change of measure, World Scientific Publishing Co Pte Ltd, 2015. 326 p (Advanced Series on Statistical Science and Applied Probability, Vol. 21).

[13] Rolf Waebler, Peter Frazier & Shane Henderson, Bisection Search With Noisy Responses, SIAM Journal on Control and Optimization, 2013.

[14] Yelp Academic Dataset, https://www.yelp.com/dataset_challenge

[15] Radim Rehůrek and Petr Sojka, Software Framework for Topic Modelling with Large Corpora, Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, p45-p50, 2010.
9 Supplement

9.1 Proof of Lemma 5

First we need to prove another lemma.

Lemma 5. Suppose \( Z(k) \) is a random walk starts with \( Z(0) = 0 \). \( Z(k + 1) = Z(k) + 1 \) with probability \( p > 0.5 \) and \( Z(k + 1) = Z(k) - 1 \) with probability \( 1-p \), then

\[
E\left[ \sum_{t=0}^{\infty} \mathbb{1}\{Z(t) \leq 0\} \right] = \frac{2p(1-p)}{(2p-1)^2}. \tag{5}
\]

Proof. Denote \( A = E[t : \min_{t>1} Z(t) = 0 | Z(1) = -1] \) and \( B = P(\exists t, Z(t) = 0 | Z(1) = 1) \), then we know

\[
E\left[ \sum_{t=0}^{\infty} \mathbb{1}\{Z(t) \leq 0\} \right] = 1 + (1-p)(A + E\left[ \sum_{t=0}^{\infty} \mathbb{1}\{Z(t) \leq 0\} \right]) + pB(E\left[ \sum_{t=0}^{\infty} \mathbb{1}\{Z(t) \leq 0\} \right]).
\]

Now we need to calculate the expression for \( A \) and \( B \) respectively.

Based on the definition of \( A \), we can rewrite \( A \) as \( E[t : \min_{t>1} Z(t) = 1 | Z(t) = 0] \). It is easy to show that \( Y(t) := Z(t) - (2p-1)t \) is a martingale. Here we define a stopping time \( \tau \) as \( \min\{t > 1 : Z(1) = 1\} \). Then we know \( Y(t) \) stops at \( \tau \) is a martingale and thus \( E[Y(\tau)] = E[Z(\tau)] - (2p-1)E[\tau] = 0 \). Thus \( A = \frac{1}{2p-1} \).

For \( B \), based on the first step analysis, we know

\[
B = (1-p) + p \times B^2.
\]

Solving this equation, we get \( B = \frac{1-p}{p} \).

Plus in \( A \) and \( B \)'s expression, we have

\[
E\left[ \sum_{t=0}^{\infty} \mathbb{1}\{Z(t) \leq 0\} \right] = \frac{p}{(2p-1)^2},
\]

which is what we need.

Now we can get back to the proof of Lemma 1.

Proof. Suppose \( W(t) \) is a random walk and \( W(t+1) = W(t) + 1 \) with probability \( p \) and \( W(t+1) = W(t) - 1 \) with probability \( 1-p \). Based on the previous Lemma, we just need to show

\[
E\left[ \sum_{t=0}^{\infty} \mathbb{1}\{Z(t) \leq 0\} \right] \leq E\left[ \sum_{t=0}^{\infty} \mathbb{1}\{W(t) \leq 0\} \right]. \tag{6}
\]

Because \( E\left[ \sum_{t=0}^{\infty} \mathbb{1}\{W(t) \leq 0\} \right] = \sum_{t=0}^{\infty} P(W(t) \leq 0) \) and

\[
P(W(t) \leq 0) = \sum_{m \leq 0.5t} \binom{t}{m} p^{-m}(1-p)^m \geq P(Z(t) \leq 0),
\]

thus, we know Equation (6) holds true.

9.2 Proof of Theorem 4

For the \( k_{th} \) integer programming, denote \( J_{k,i} = \{(p,q) : e_{p,q} = 1 \text{ in the } i_{th} \text{ solution}\} \). Here we denote \( m_{k,i}(t) = \sum_{(i_0,j_0) \in J_{k,i}} q_{i_0,j_0}(t) \). We still use \( C_1 \) as the cell that contains \( \theta \) and \( m_{1}(t) \) is its corresponding counter.

Proof. In this case, we just need to show that \( E[N_{i,j}(t)] \) is again bounded by a constant. The proof is similar to the previous proof. Without loss of generality, we still assume that \( i < j \).
\[ N_{i,j}(t) \leq \sum_{k=1}^{t} \mathbb{1}\{\max_{r} m_{j,r}(k) \geq m_1(k)\} \]
\[ \leq \sum_{r} \sum_{k=1}^{t} \mathbb{1}\{m_{j,r}(k) \geq m_1(k)\} \]
\[ = \sum_{r} \sum_{k=1}^{t} \mathbb{1}\{ \sum_{(i_0,j_0) \in J_{j,r}} q_{i_0,j_0}(k) \geq \sum_{(i_0,j_0) \in J_i} q_{i_0,j_0}(k)\} \]
\[ = \sum_{r} \sum_{k=1}^{t} \mathbb{1}\{ \sum_{(i_0,j_0) \in J_{j,r} \backslash J_1} q_{i_0,j_0}(k) \geq \sum_{(i_0,j_0) \in J_1 \backslash J_{j,r}} q_{i_0,j_0}(k)\} \]
\[ = \sum_{r} \sum_{k=1}^{t} \mathbb{1}\{ \sum_{(i_0,j_0) \in J_{j,r} \backslash J_1} [q_{i_0,j_0}(k) - q_{j_0,i_0}(k)] \geq 0\}, \]

There are two key arguments to the proof:

- \( C_1 \) is contained in the intersection of all the winning spaces that are supposed to win the duel.
- \((j, i) \in J_{j,r} \backslash J_1\), which is because \( A_1 \in H_{i,j} \) and \( A_j \in H_{j,i} \).

Based on the same proof, we can show that

\[ E[N_{i,j}(t)] \leq \sum_{r} E \left[ \sum_{k=1}^{t} \mathbb{1}\{ \sum_{(i_0,j_0) \in J_{j,r} \backslash J_1} [q_{i_0,j_0}(k) - q_{j_0,i_0}(k)] \geq 0\} \right] \]
\[ \leq 2^{N(N-1)/2} \frac{p}{(2p - 1)^2}. \]

Thus, we know our theorem holds true. \( \blacksquare \)