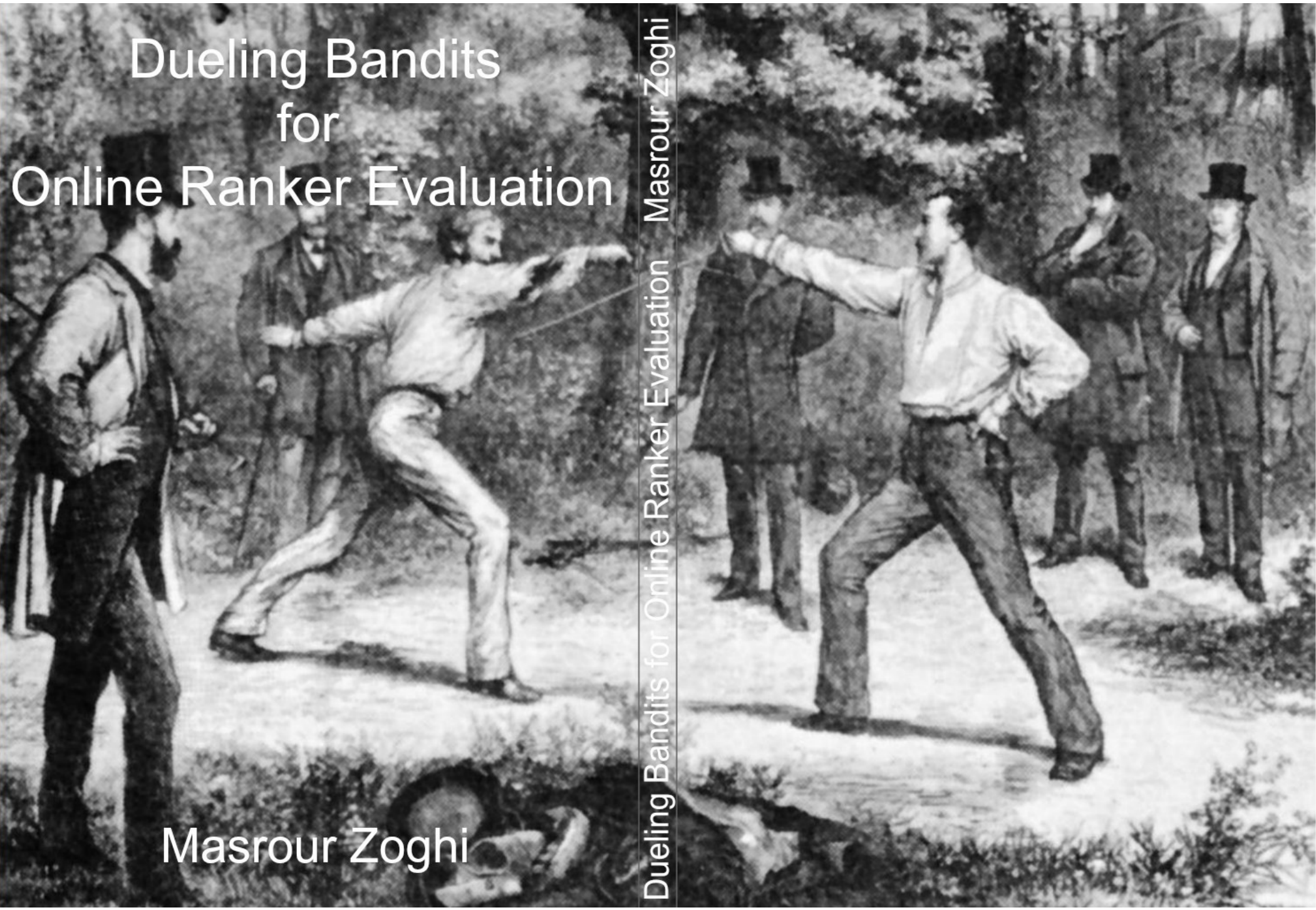
Dueling Bandits
for
Online Ranker Evaluation
Masrour Zoghi
Dueling Bandits for Online Ranker Evaluation Masrour Zoghi

Dueling Bandits for Online Ranker Evaluation

Masrour Zoghi

Graduation committee:

Chairmen:	Prof.dr. P.M.G. Apers	Universiteit Twente
	Prof.dr.ir. A. Rensink	Universiteit Twente
Supervisors:	Prof.dr. P.M.G. Apers	Universiteit Twente
	Prof.dr. M. de Rijke	Universiteit van Amsterdam
Co-supervisor:	Dr.ir. D. Hiemstra	Universiteit Twente
Members:	Prof. N. de Freitas	University of Oxford
	Prof.dr.ir. B.R.H.M. Haverkort	Universiteit Twente
	Dr. R. Munos	DeepMind
	Prof.dr. M.J. Uetz	Universiteit Twente
	Prof.dr.ir. A.P. de Vries	Radboud Universiteit Nijmegen

CTIT Ph.D. Thesis Series No. 17-427
Centre for Telematics and Information Technology
University of Twente
P.O. Box 217
7500 AE Enschede, The Netherlands

Copyright © 2017 Masrour Zoghi, Enschede, The Netherlands
ISBN: 978-90-365-4287-6
ISSN: 1381-3617 (CTIT Ph.D. thesis Series No. 17-427)
DOI: 10.3990/1.9789036543026
https://doi.org/10.3990/1.9789036543026

DUELING BANDITS FOR ONLINE RANKER EVALUATION

DISSERTATION

to obtain
the degree of doctor at the University of Twente,
on the authority of the rector magnificus
Prof.dr. T.T.M. Palstra
on account of the decision of the graduation committee,
to be publicly defended
on Friday February 24, 2017 at 14:45

by

Masrour Zoghi

born on September 19, 1978
in Karaj, Iran

This dissertation has been approved by:

Prof.dr. P.M.G. Apers (supervisor)

Dr.ir. D. Hiemstra (co-supervisor)

Prof.dr. M. de Rijke (supervisor)

Acknowledgments

I would like to first thank my advisor, Maarten, for his patient guidance over the years. In addition to that, I benefited from conversations and collaborations with a long list of researchers, including Akshay Balsubramani, Bogdan Cautis, Chun Ming Chin, Fernando Diaz, Miro Dudik, Nando de Freitas, Mohammad Ghavamzadeh, Katja Hofmann, Frank Hutter, Thorsten Joachims, Damien Jose, Satyen Kale, Evangelos Kanoulas, Zohar Karnin, Akshay Krishnamurthy, Brano Kveton, Damien Lefortier, Lihong Li, Ilya Markov, Rémi Munos, David Pal, Filip Radlinski, Rob Schapire, Anne Schuth, Milad Shokouhi, Alex Slivkins, Adith Swaminathan, Csaba Szepesvari, Tomas Tunys, Ziyu Wang, Zheng Wen and Shimon Whiteson. I would also like to extend my gratitude to my colleagues at ILPS for their support, not to mention an endless supply of interesting conversations. In particular, I would like to thank Petra for her indispensable help in navigating the bureaucracy. Furthermore, a special thanks goes to Djoerd and the rest of my committee for patiently reading through this thesis.

On a more social level, I would like to thank my friends Jean, Spyros and Tony for many enjoyable conversations that made my stay in Amsterdam memorable, as well my cousin Amin and his friends, who made my numerous trips to Berlin the highlight of my half-decade excursion to Europe.

Masrour Zoghi
Jan 2017

Contents

1 Introduction

In every domain where a service or a product is provided, an important question is that of evaluation: given a set of possible choices for deployment, what is the best choice? An important example, which is considered in this work, is that of *ranker evaluation* from the field of *information retrieval* (IR). The goal of IR is to satisfy the information need of a user in response to a query issued by them, where this information need is typically satisfied by a document (or a small set of documents) contained in what is often a large collection of documents [51]. This goal is often attained by ranking the documents according to their usefulness to the issued query using an algorithm, called a *ranker*, a procedure that takes as an input a query and a set of documents and specifies how the documents need to be ordered [51].

Let us illustrate this. The typical scenario, familiar to anyone with internet access, is that of web search: suppose you happen to be reading a thesis and run into a cited paper that has piqued your interest and you would like to inspect it more closely; then, if you are trapped in the 1990s, you could spend a substantial amount of time guessing the URL of the publisher and search through their archives for the issue that contains the article, or you could do what every person living in 2016 would do, which is to type the title of the article into a popular search engine and get a link to the article. In this case, the collection is all documents and pages on the web and the information need of the user is satisfied by the sought after article. There is, however, the issue that there might be several articles with similar titles or there might even be different versions of the same article and the user might be looking for a very specific version. The remedy used to address this difficulty is often to present a list of documents, rather than a single document, to the user, hoping that one or more of them satisfy the user's need. This gives rise to the problem of ranking, whose goal is to place the more useful documents at the top.

This thesis is concerned with *ranker evaluation* [39, 45, 60]. The goal of ranker evaluation is to determine the quality of rankers to allow us to use the best option: given a finite set of possible rankers, which one of them leads to the highest level of user satisfaction? There are two main methods for carrying this out:

Absolute metrics: The idea here is to use a *metric* that assigns an *absolute* measure of the quality of each ranker in the form of a real number and picks the ranker of the highest quality according to our metric. This could be either an *offline* metric (e.g., NDCG [41], MAP [32], etc.), which is calculated using annotated relevance judgments for the documents being ranked, or an online metric (e.g.,

time to success [27], which is calculated based on the feedback provided by the users. The latter is often carried out using A/B tests [46, 47]. This is carried out by applying each ranker to a different portion of the traffic and using a measure of the performance of the rankers to compare them against each other.

Relative comparisons: Alternatively, one could directly compare each ranker to the other rankers under consideration using interleaved comparisons [43]: This carried out by merging the results produced by a pair of rankers and using the feedback provided by the user on the resulting list of documents to decide which of the two rankers was preferred to the other. These *relative* comparisons could then be used to decide which ranker is preferred to the rest by the users of the system.

This thesis is concerned with the second, relative form of ranker evaluation because it is more efficient at distinguishing between rankers of different quality [20]: for instance interleaved comparisons take a fraction of the time required by A/B testing, but they produce the same outcome [62]. The reason for this improved efficiency is that absolute metrics calculate average performance across the whole population of queries and users and so the estimated quantities tend to have rather large variance; an relative comparison, on the other hand, takes place between the results produced by two rankers for a single query and based on the feedback of a single user, so the comparison tends to be better indicator of the relative quality of the two rankers. More precisely, the problem of *online ranker evaluation from relative feedback* can be described as follows: given a finite set of rankers, choose the best using only pairwise comparisons between the rankers under consideration.

More generally, in the above description, we could replace the word "ranker" with any object that yields itself to relative comparisons, such as images [76] or animations [11], where the task might be a subjective one such as "find the photo with the happiest face." What makes relative comparisons more suitable for such a task is the fact that it is much easier to decide which of two photos look happier than to assign a "happiness score" to a single image. More importantly, when faced with such a task, a population of users is more likely to express consistent preferences for one image over the other than it is for them to assign similar scores to individual images.

1.1 Research Outline and Questions

Here, we describe the research questions addressed in this thesis, each of which is a variation on the following question:

> *suppose we are given a finite set of objects (called "arms" for historical reasons [59]) such that we can only compare two of them at a time; then, can we find the best arm efficiently without imposing prohibitively restrictive assumptions?*

Examples of arms include ads, images and animations. We will be especially interested in rankers and often read "ranker" for "arm."

There are three components of the above question in italics that need to be made more precise for the research questions to make sense:

1. What does it mean to be the *best arm*?
2. What constitutes a *prohibitively restrictive assumption*?
3. How is *efficiency* measured?

These questions are going to be addressed in greater detail in Chapter 2; however, in order for the research questions to make sense, we provide here a brief and high level discussion of how these questions were answered in the literature that preceded this thesis. The first paper to investigate the question in italics was that of Yue et al. [74], where the authors formulated the *dueling bandit* problem, whose goal is to find the best arm as quickly as possible using only pairwise comparisons (cf. §2.1.2 for the precise definition). They also proposed an algorithm, called Interleaved Filter (IF), which required the arms to satisfy a *total ordering* assumption, which precluded any situation where the arms are in a cyclical preference relationship, i.e., if we happen to have three arms A, B and C such that A is preferred to B, B is preferred to C and C is preferred to A, then IF would not be guaranteed to find the "best" arm. In this situation, the best arm is simply the arm at the top of the hierarchy dictated by the total ordering assumption.

It turns out that cyclical preference relationships occur regularly in applications [24, 78]. This is because even if each individual user interacting with the system is rational in their choices, a population of users could easily be irrational, in the sense that they might have cycles in their preferences. Moreover, even when comparing pairs of real valued random variables, one can come across cyclical relationships, as pointed out by Gardner [29].

Now, given that assuming a total ordering among the arms is not a safe assumption in practice, the first natural question is: what is the most natural choice for the "best arm"? To answer this, it helps to bear in mind the intended application, which is to find an option that is preferred over the rest, so a natural solution is to adopt the notion of a Condorcet winner from the field of Social Choice Theory [25]: a *Condorcet winner* is an arm that is preferred to every other arm on average, i.e., given a comparison between the Condorcet winner and any other arm, the former is more likely to win than the latter.

The next natural question is if one can devise an algorithm that is guaranteed to work in the absence of a total ordering, but just assuming the existence of a Condorcet winner. Moreover, can such an algorithm be as efficient as IF? This leads us to the issue of what we mean by efficiency: Yue et al. [74] define a measure of the performance of a dueling bandit algorithm, called *cumulative regret*, which is the sum of the "regret" or missed opportunity incurred by the dueling bandit algorithm as it compares different pairs of arms in each time-step to find the best one. More precisely, the regret accumulated by the algorithm when it chooses to compare two arms is measured in terms of the probability with which each of the two arms loses to the Condorcet winner in a one-on-one comparison. The reader is referred to §2.1.2 for the precise definition, but for now let us point out that lower cumulative regret means that the algorithm is performing better, so we prove upper bounds on the cumulative regrets of our algorithms to show that they do not perform too poorly, while a lower bound on cumulative regret means that no algorithm can perform better than what the bound prescribes.

Using this measure of the quality of the algorithm, Yue et al. [74] provide a lower bound on how low the cumulative regret of any dueling bandit algorithm has to be: given

K arms and an experiment of length T,[1] they prove that the cumulative regret of the algorithm has to be higher than $\Omega(K \log T)$. Also, they show that the cumulative regret of IF is bounded by $\mathcal{O}(K \log T)$, which matches the lower bound.

One can divide the results that existed in the literature before the beginning of the research that gave rise to this thesis into two groups:

1. Algorithms with similar regret results as IF's, i.e., $\mathcal{O}(K \log T)$, but proven under similarly or even more restrictive assumption, e.g., Beat the Mean (BTM) [73], Doubler and MultiSBM [3].

2. Algorithms that were proven under more general assumptions but with regret bounds of the form $\mathcal{O}(K^2 \log T)$, e.g., the different variants of Sensitivity Analysis of VAriables for Generic Exploration (SAVAGE) [65].

Given the above dichotomy, one might wonder whether the same could be shown under the more general Condorcet assumption, which is the purpose of our first research question:

RQ1 Can the lower bound for dueling bandits be met without the total ordering assumption?

As we will see in Chapter 4, this question is answered in the affirmative using the first algorithm proposed in this thesis, called Relative Upper Confidence Bound (RUCB). We prove an upper bound on the cumulative regret of RUCB that takes the form $\mathcal{O}(K \log T)$ under the Condorcet assumption, which breaks the dichotomy that existed in the results preceding the publication of RUCB.

Despite the improvements that RUCB introduced over existing work, it has a number of flaws that the rest of this thesis attempts to address, as outlined in the research questions that follow. Let us begin by listing these shortcomings:

Exploration. As discussed in greater detail in Chapter 4, RUCB tends to be rather conservative when it comes to balancing exploration and exploitation. For instance, RUCB refuses to compare an arm against itself unless it is very confident that the arm is the Condorcet winner. Let us point out what this means in practice, using the ranker evaluation example: suppose we are given K rankers to evaluate in an online fashion, which means that rather than fixing a budget for our evaluation, by the end of which we need to produce a winner, we keep carrying out interleaved comparisons hoping that the best arm is chosen by the algorithm more and more frequently. In this scenario, if the dueling bandit algorithm proposes the same arm to be compared against itself, we can simply display the ranked list produced by the ranker proposed by the algorithm, without conducting an interleaved comparison. If the algorithm required feedback in this case, we could simply flip a fair coin and return the result to the algorithm, since we know that each arm beats itself with probability 0.5. Therefore, the sooner the dueling bandit algorithm starts proposing the Condorcet winner to be compared against itself, the better

[1] By "length of the experiment" we mean the number of pairwise comparisons carried out by the dueling bandit algorithm. We think of each comparison as occurring in a single *time-step*, which corresponds to a full *iteration* of the algorithm. Given this interpretation, we use the words "time-step" and "iteration" interchangeably and use "length" or "duration" for the number of comparisons conducted by the algorithm.

its performance. We call such a behavior on the part of the algorithm as being *more exploitative*, and it is a desirable property for a dueling bandit algorithm to have because the sooner the algorithm begins to compare the Condorcet winner against itself, the sooner it stops accumulating regret. Indeed, we use the cumulative regret of the algorithm as evidence for how exploitative it is. This forms the motivation behind our second research question:

RQ2 Can we devise an algorithm that is more exploitative than RUCB?

We answer this question in the affirmative by introducing Relative Confidence Sampling (RCS) in Chapter 5 and carrying out an experimental comparison between RCS and RUCB.

Dependence on K. Another difficulty with RUCB is that its regret bound takes the form $\mathcal{O}(K^2 + K \log T)$: note that the additive term is quadratic in K. This is simply because RUCB has to compare all pairs of arms before it can discover the Condorcet winner. This means that RUCB would have difficulty scaling up to dueling bandit problems with larger numbers of arms: note that for all practical purposes the $\log T$ term can be thought of as a constant; suppose that we can run 10^{10} iterations of RUCB (which is a sequential algorithm) per second and we run the algorithm for the life-time of the universe, then $\log T$ would still be bounded by 1000, whereas K can easily be in the thousands, in which case the K^2 term would dominate the $K \log T$ term. So, for large-scale problems, it is essential to eliminate the quadratic dependence on K. This is formulated in our next research question:

RQ3 Can the quadratic dependence on the number of arms in the regret bound for RUCB be eliminated?

We address this question by introducing an algorithm called mergeRUCB and prove that its cumulative regret grows linearly in the number of arms rather than quadratically under the Condorcet assumption.

Condorcet assumption. Finally, there remains the Condorcet assumption, which RUCB requires from the dueling bandit problem under consideration for its proper functioning. Even though experiments show that it is less likely for a Condorcet winner to fail to exist than for the total ordering assumption to be violated, dueling bandit problems without Condorcet winners do arise in practice [79]. This gives rise to the following research question:

RQ4 Can the lower bound for the dueling bandit problem be met under practically general assumptions?

We address this by introducing the Copeland Confidence Bound (CCB) algorithm and proving a regret upper bound that resembles that of RUCB. The only restriction CCB imposes upon the dueling bandit problem under consideration for this guarantee is that no two arms should be completely tied with each other.

Even though the above requirement might seem stringent, let us examine what it means in practice. For instance, in the ranker evaluation application, this means that

for every pair of rankers under consideration, one of them should be preferred over the other. Now, given that each of these rankers is in practice the outcome of the arduous labour of a team of engineers whose goal is to beat the state of the art and that under normal circumstances these teams would be in close contact with each other during the development process, it is rather difficult to imagine two teams proposing exactly the same ranker twice. Similarly, in other application domains, unless two images or animations are identical, it is impossible for a population of users to be completely ambivalent as far as preference for one item over the other is concerned.

Given these observations, we consider the requirement for the non-existence of ties to pose little hindrance in practice and so we consider the result for CCB to be "practically general."

1.2 Main Contributions

In this section, we provide an overview of the main contributions of this thesis, which can be divided into two groups:

Algorithmic contributions. The *main* algorithmic contribution of this thesis is devising the CCB algorithm, which solves the dueling bandit problem under practically general assumptions. Additionally, this thesis makes the following contributions:

- A simple algorithm (i.e., RUCB) that adapts a popular algorithm for the multi-armed bandit (MAB) problem, called Upper Confidence Bound (UCB), to the dueling bandit setting under the Condorcet assumption.
- A simple modification of RUCB (i.e., RCS) that combines the ideas of two well-known MAB algorithms, namely UCB and Thompson sampling, and which allows for a more efficient algorithm.
- A scalable version of RUCB (i.e., mergeRUCB) whose regret grows linearly in the number of arms, rather than quadratically.

Theoretical contributions The *main* theoretical contribution of this thesis is providing the first theoretical analysis of an algorithm that holds under practically general assumptions, but also its (temporally) asymptotic dependence on the number of arms is linear, i.e., it takes the form $\mathcal{O}(K \log T)$, where K is the number of arms and T is the number of time-steps the algorithm has been run. Additionally, we would like to point out the following contributions:

- A novel proof technique for a UCB-style algorithm that allows for high probability regret bounds without the need to specify the probability of failure as a parameter to the algorithm.
- The first theoretical analysis of a dueling bandit algorithm that has no quadratic dependence on the number of arms and does not assume a total ordering on the arms.

1.3 Thesis Overview

This section provides an overview of the remainder of this thesis. In Chapter 2, we give a precise definition of the problem setting and provide the necessary background for the reader to comprehend the results in the subsequent chapters.

In Chapter 3 we describe the experimental setup used in later chapters, which includes datasets, metrics, and significance tests.

Chapters 4-7 provide the details of the four proposed algorithms, as well as theoretical and experimental results comparing them against the state of the art at the time they were proposed.

More precisely, we have the following break down:

- Chapter 4 presents the RUCB algorithm and its theoretical guarantees. It also provides an experimental comparison between RUCB, Beat the Mean (BTM) and Condorcet SAVAGE, using the ranker evaluation problem.
- Chapter 5 presents the RCS algorithm and an experimental comparison between RCS, RUCB, Condorcet SAVAGE and BTM. Due to technical difficulties a theoretical analysis of RCS remains elusive.
- Chapter 6 presents the mergeRUCB algorithm, which is the state of the art scalable dueling bandit algorithm under the Condorcet assumption. We provide both a theoretical analysis of mergeRUCB, as well as a comprehensive experimental comparison.
- Chapter 7 discusses the CCB algorithm, which is the first practically general and efficient algorithm with theoretical guarantees. We compare CCB against numerous algorithms that preceded it, demonstrating its good performance in practice.

In Chapter 8, we summarize the main results obtained in the thesis and provide suggestions for future work.

All research chapters build on background introduced in Chapter 2 and alll use the experimental setup detailed in Chapter 3, even though several chapters introduce additional experimental details required to answer their specific research question. Assuming knowledge of the background material provided in Chapter 2 and 3, every chapter is self-contained. Despite this, the preferred reading order is the natural order, Chapter 4, 5, 6, and 7.

1.4 Origins

The material in this thesis first appeared in the following publications:

- Chapter 4 is based on Masrour Zoghi, Shimon Whiteson, Remi Munos, and Maarten de Rijke, Relative upper confidence bound for the k-armed dueling bandits problem, which appeared in *ICML*, 2014 [78].
- Chapter 5 is based on Masrour Zoghi, Shimon Whiteson, Maarten de Rijke, and Remi Munos, Relative confidence sampling for efficient on-line ranker evaluation, which appeared in *WSDM*, 2014 [77].

- Chapter 6 is based on Masrour Zoghi, Shimon Whiteson, and Maarten de Rijke, MergeRUCB: A method for large-scale online ranker evaluation, which appeared in *WSDM*, 2015 [80].
- Chapter 7 is based on Masrour Zoghi, Zohar Karnin, Shimon Whiteson, and Maarten de Rijke, Copeland dueling bandits, which appeared in *NIPS*, 2015 [79].

Work on the thesis also benefitted from insights gained through research that led to the following publications:

- Miroslav Dudík, Katja Hofmann, Robert E. Schapire, Aleksandrs Slivkins, and Masrour Zoghi, Contextual dueling bandits, which appeared in *Conference on Learning Theory (COLT)*, 2015 [24].
- Ziyu Wang, Masrour Zoghi, Frank Hutter, David Matheson, and Nando de Freitas, Bayesian optimization in high dimensions via random embeddings, which appeared in *IJCAI*, 2013 [67].
- Masrour Zoghi, Tomáš Tunys, Lihong Li, Damien Jose, Junyan Chen, Chun Ming Chin, and Maarten de Rijke, Click-based hot fixes for underperforming torso queries, which appeared in *SIGIR*, 2016 [81].
- Akshay Balsubramani, Zohar Karnin, Robert Schapire, and Masrour Zoghi, Instance-dependent regret bounds for dueling bandits, which appeared in *Conference on Learning Theory (COLT)*, 2016 [9].

2 Background

This chapter provides the reader with the necessary background to read the following chapters. In the following two sections, we provide the precise problem statement and discuss the related work.

2.1 Problem Setting

The problem addressed in this thesis is the *K-armed dueling bandit* problem [75] which is a modification of the *K-armed bandit* problem [64]. We begin by discussing the latter in the following subsection.

2.1.1 The K-armed bandit problem

The K-armed bandit problem (also called multi-armed bandits or MAB for short) is specified by K real-valued random variables $X_1, \ldots, X_K$ (called "arms"), whose distributions are unknown to the us, but from which we can draw samples. Loosely speaking, the goal of the problem is identify the random variable with the highest mean as quickly as possible. More precisely, for each i, let us denote the mean of X_i by μ_i and every time a sample is drawn from X_i, we define the *regret* incurred by this action to be

$$r = \max_k \mu_k - \mu_i.$$

To lessen the notational burden, we will assume in the following that $\max_k \mu_k = \mu_1$: in other words, the first arm is the one with the highest mean and so we can write regret as $r = \mu_1 - \mu_i$. However, the algorithm solving the problem is assumed not to be aware of the fact arm 1 is the best arm.

Now, let us consider the following iterative process: in each time-step, we get to select one of the arms and observe the random sample from the corresponding random variable. Note that we do not observe regret because we do not know which arm has the highest mean. Our goal then is to minimize our *cumulative regret* over time, defined to be the sum of the regret we incurred by our choice of arm at each time-step. More precisely, letting r_t be the regret incurred at time t, then cumulative regret after T time-steps is defined to be

$$R_T = \sum_{t=1}^{T} r_t = \sum_{t=1}^{T} \left(\mu_1 - \mu_{i_t}\right), \tag{2.1}$$

where i_t is the arm chosen at time t.

A simple, yet effective algorithm for solving the K-armed bandit problem is the Upper Confidence Bound (UCB) algorithm, which we describe in the following, since the key idea behind it is heavily used by the algorithms discussed in this thesis. The pseudo-code for UCB is provided in Algorithm 1: the algorithm keeps track of the number of times each arm has been pulled (i.e., $\mathbf{N}$ on Line 2) and the sum of the values returned by each arm (i.e., $\mathbf{W}$ on Line 1) and uses these numbers to estimate the mean of each arm (i.e., $\mathbf{W}/\mathbf{N}$ on Line 4).

Algorithm 1 Upper Confidence Bound (UCB)

Require: K arms $a_1, \ldots, a_K$ corresponding to K independent real-valued random variables and $\alpha > \frac{1}{2}$,

1: $\mathbf{W} = [w_i] = \mathbf{0}_K$ // 1D array of the sum of the values returned by each arm
2: $\mathbf{N} = [n_i] = \mathbf{0}_K$ // 1D array of the number of times each arm has been pulled
3: **for** $t = 1, 2, \ldots$ **do**
4: $\quad \mathbf{U}(t) = [u_i(t)] = \frac{\mathbf{W}}{\mathbf{N}} + \sqrt{\frac{\alpha \ln t}{\mathbf{N}}}$ // The UCB for each arm: all operations are element-wise; $\frac{x}{0} = 1$ for any x.
5: $\quad \widehat{\imath} = \arg\max_i u_i(t)$, with ties broken randomly.
6: $\quad$ Pull arm $a_{\widehat{\imath}}$, increment $n_{\widehat{\imath}}$ and add the value returned by $a_{\widehat{\imath}}$ to $w_{\widehat{\imath}}$.

At this point, the first idea that might come to mind is to use these estimates to decide which arm to play, however the danger of such a simple approach is that if by sheer bad luck one were to underestimate the mean of the best arm as being below the true mean of the second best arm, then one might never be able to dig oneself out of this "false optimum." The purpose of the second summand in Line 4 of Algorithm 1 is to prevent such a catastrophic outcome from occurring. More precisely, the term $\sqrt{\alpha \ln t / \mathbf{N}}$ gives each arm an optimistic boost that has two important properties: first of all, the boost grows with time, so even if UCB falls in the trap of mistaking a suboptimal arm for the winner, we know that the UCB of the best arm will keep growing and eventually overtake the UCB of the impostor. The second important property of $\sqrt{\alpha \ln t / \mathbf{N}}$ is that because the denominator under the square root is equal to the number of times each arm has been pulled, the optimistic boost is larger for arms that have been pulled less frequently. In other words, if we are less confident about our estimate of the mean of an arm we give it a bigger chance to be picked. So, each arm is pulled "enough" times for us to be relatively certain that we are not mistakenly disqualifying a good arm.

Speaking in more precise terms, the fact that renders the u_i useful is that we can show the following key intermediate result:

$$l_i(t) \leq \mu_i \leq u_i(t) \quad \text{for large enough } t \text{ with high probability,} \tag{2.2}$$

where μ_i is the mean of arm a_i as before, and $u_i(t)$ is the UCB for the same arm after t time-steps and we define

$$l_i(t) = \frac{w_i(t)}{n_i(t)} - \sqrt{\frac{\alpha \ln t}{n_i(t)}}.$$

To the reader familiar with concentration inequalities from probability theory [68], this might seem, upon first inspection, like a direct application of the Chernoff-Hoeffding

bound [34]. However, it turns out to be more complicated due to the fact that our estimates of the μ_i are not unbiased. We will discuss these subtleties when we prove a similar result in the dueling bandit context in Chapter 4.

For now, let us note that if one were to assume the veracity of inequality (2.2), then it is easy to see why after T time-steps each suboptimal arm is pulled at most $\mathcal{O}(\log T)$ many times: indeed, assume as before that $\mu_1 = \max_i \mu_i$ and denote the sub-optimality gap of each arm by

$$\Delta_i = \mu_1 - \mu_i.$$

Now, for each arm a_i with strictly positive Δ_i, define T_i be the last time until time T when arm a_i was pulled, and note that the following facts hold at time T_i:

$$\begin{aligned} \mu_1 &\leq u_1(T_i) \quad \text{by (2.2)} \\ u_1(T_i) &\leq u_i(T_i) \quad \text{by Line 5 of Algorithm 1 and that } a_i \text{ was chosen by UCB at time } T_i, \end{aligned}$$

which means that together with inequality (2.2) applied to arm a_i at time T_i we have

$$l_i(T_i) \leq \mu_i < \mu_1 \leq u_i(T_i).$$

Since the gap between μ_i and μ_1 is equal to Δ_i, we can conclude that

$$\Delta_i = \mu_1 - \mu_i \leq u_i(T_i) - l_i(T_i) = 2\sqrt{\frac{\alpha \ln T_i}{n_i(T_i) - 1}} \leq 2\sqrt{\frac{\alpha \ln T}{n_i(T) - 1}},$$

where the last inequality is due to the fact that time T_i was the last time before T when a_i was pulled and so $n_i(T) = n_i(T_i)$ and $\ln$ is a monotonic function, so $\ln T_i \leq \ln T$. Now, we can use the above inequality to bound $n_i(t)$ by an expression in terms of Δ_i as follows:

$$n_i(t) \leq \frac{4\alpha \ln T}{\Delta_i^2} + 1.$$

This in turn allows us to bound the regret accumulated by UCB with the important caveat that we have not specified how large t needs to be for inequality (2.2) to hold: this is pinned down in Chapter 4 and specifies the non-asymptotic (in T) component of the regret bound.

2.1.2 The K-armed dueling bandit problem

The K-armed *dueling* bandit problem [75] is a variation on the K-armed bandit problem, where instead of pulling a single arm at each time-step, we choose a pair of arms (a_i, a_j) to be compared against each other and receive either a_i as the better choice with some unknown probability p_{ij} or a_j with probability $p_{ji} = 1 - p_{ij}$. We define the *preference matrix* $\mathbf{P} = [p_{ij}]$, whose ij entry is equal to p_{ij}. Note that in the dueling bandit setting the quality of each arm is only defined in relation to other arms, so unlike the K-armed bandit problem, there are no absolute quantities μ_i to dictate which arm is the best. Indeed, deciding what constitutes a winner in this setting is a problem that has kept social choice theorists occupied for decades [61].

We address the problem of defining the best arm in the dueling bandit setting in two ways:

- by assuming the existence of an arm that on average beats all other arms, the so-called *Condorcet winner* [65]: formally, the Codorcet winner is an arm a_c such that for all $j \neq c$ we have $p_{cj} > 0.5$;
- by using the *Copeland winner* [65], which is guaranteed to exist: a Copeland winner is defined with the highest *Copeland score*, which is the number of other arms that a given arm beats; more precisely, we define

$$\text{Cpld}(a_i) = \#\{j \,|\, p_{ij} > 0.5\},$$

and arm a_c is said to be a *Copeland winner* is $\text{Cpld}(a_c) \geq \text{Cpld}(a_i)$ for all i.

Note that the Copeland winner is a generalization of the Condorcet winner, since in situations where the Condorcet winner exists, it is the unique Copeland winner, since the Condorcet winner by definition beats all other arms.

Moreover, we define the following two notions of regret:

Condorcet: If we know a priori that the Condorcet winner exists, we define the regret incurred by comparing a pair of arms a_i and a_j to be

$$r = \frac{p_{1i} + p_{1j} - 1}{2}, \tag{2.3}$$

where arm a_1 is assumed to be the Condorcet winner as before. Note that we accumulate zero regret from the comparison if and only if a_i and a_j are both the Condorcet winner, otherwise the regret is strictly positive since by assumption we have $p_{1i} > 0.5$ for all $i \neq 1$.

Copeland: If the goal is to find a Copeland winner and the existence of a Condorcet winner is not guaranteed, we define the regret incurred by comparing arms a_i and a_j to be

$$r = \frac{2\text{Cpld}(a_1) - \text{Cpld}(a_i) - \text{Cpld}(a_j)}{K}, \tag{2.4}$$

where arm a_1 is assumed to be a Copeland winner.

Moreover, in either case, we define cumulative regret in a similar fashion as with Equation (2.1):

$$R_T = \sum_{t=1}^{T} r_t \tag{2.5}$$

where r_t is the regret incurred as a result of the comparison carried out in time-step t.

A few general comments are in order at this point. Let us first point out that from a preference learning point of view the Condorcet winner is a desirable notion of a winner because, by its very definition, the Condorcet winner is preferred to all other arms by the users whose preferences we are trying to accommodate. So, if the Condorcet winner exists, our dueling bandit algorithm should converge to it. Indeed, an important property of the Copeland winner is that in the presence of a Condorcet winner, the two definitions coincide. However, let us clarify at this point that we are not claiming that the Copeland winner is the best generalization of the Condorcet winner one can envisage. Indeed, this

question is a very difficult one to answer in general because one often has to deal with multiple conflicting criteria for what constitutes a good notion of a winner, which is why there is a proliferation of such notions in social choice theory [61] without a clear contender for the best answer. We have chosen the Copeland winner here mostly because of its precedence in the dueling bandit literature [14].

Let us also take a few minutes to explain by way of an example some difficulties that one needs to overcome when attempting to solve a dueling bandit problem. Consider the following preference matrix:

$$\mathbf{P} = \begin{bmatrix} .5 & .51 & .51 & .51 \\ .49 & .5 & 1 & .4 \\ .49 & 0 & .5 & .75 \\ .49 & .6 & .25 & .5 \end{bmatrix}.$$

Note that the first arm (corresponding to the first row and column) is the Condorcet winner even though it beats the other arms by very thin margins, while the other three arms (corresponding to rows and columns 2–4) are in a cyclical relationship, with each of them beating one of the the other two, while losing to the third one. Moreover, note that the second arm is what is known in the literature as the *Borda winner*, i.e. an arm that beats the "average arm" by the widest margin [65]. More precisely, a_b is a Borda winner if we have

$$b = \arg\max_i \sum_{k=1}^{K} p_{ik}.$$

Indeed, one of the main challenges when designing a dueling bandit algorithm is to make sure the algorithm does not fall into the trap of mistaking the Borda winner for the Condorcet winner and prematurely start comparing the Borda winner against itself. This is because one of the major difficulties with the dueling bandit problem is that it is much more difficult to recover from such a mistake than it is when solving the multi-armed bandit (MAB) problem: when dealing with an MAB, pulling an arm always gives us a feedback, so if we mistake an inferior arm for the optimal one and keep pulling it, we get better and better estimates of its mean; on the other hand, in the dueling bandit setting, comparing an arm against itself yields no additional information because we know in advance that each arm is tied with itself. Given this, if our algorithm were to stop exploration and to only compare the Borda winner against itself, it would never realize its own folly and so would never recover from this "local optimum," hence accumulating linear regret. So, the urge to start exploiting what we believe to be the best arm should be carefully balanced with the need to perform adequate exploration.

The above preference matrix illustrates another reason for why the dueling bandit problem is more challenging than the regular bandit problem: in the case of the latter, even though we might not know which arm is the best one, we can estimate the regret of each arm because we can estimate the means of all of the arms: for instance, if we know with high probability the mean reward of each arm with an accuracy of ϵ, then we can estimate the regret of each arm with an accuracy of 2ϵ. In the case of the dueling bandit problem, however, there is a discontinuity in the definition of regret that prevents us from getting a similar estimate of the regret of each arm. For instance, in the above example, if we know the entries of the matrix with an accuracy that is greater than 0.1, then both the

first arm and the second one could be the Condorcet winner, but that would mean that the regret associated with playing the third arm might be either 0.01 or 0.5, which is not a very good estimate by any stretch of imagination.

Finally, another property of the dueling bandit problem defined by the above preference matrix is that each suboptimal arm is beaten by the widest margin by an arm other than the Condorcet winner, so if the algorithm were to adopt a "hill climbing" strategy [71], it would forever cycle among the sub-optimal arms. By "hill climbing" in this setting we mean the following scheme: we can start by choosing a random arm and try to find another arm that beats it by the widest margin and replace the former by the latter and repeat this process.

2.2 Related Work

In this section, we discuss other algorithms that have appeared in the literature for the K-armed dueling bandit problem.

2.2.1 IF and BTM

The first two methods proposed for the K-armed dueling bandit problem are Interleaved Filter (IF) [75] and Beat the Mean (BTM) [73], both of which were designed for a finite-horizon scenario. These methods work under the following restrictions:

1. a total ordering of the arms, i.e. we can relabel the arms as $a_1, \ldots, a_K$ such that $p_{ij} > 0.5$ for all $i < j$.

2. Stochastic Triangle Inequality (STI): for any pair (j, k), with $1 < j < k$, the following condition is satisfied:

$$\Delta_{1k} \leq \Delta_{1j} + \Delta_{jk},$$

   where $\Delta_{ij} := p_{ij} - 0.5$.

3. IF and BTM require two slightly different conditions:

   IF: Strong Stochastic Transitivity (SST): for any triple (i, j, k), with $i < j < k$, the following condition is satisfied:

$$\Delta_{ik} \geq \max\{\Delta_{ij}, \Delta_{jk}\}.$$

   BTM: Relaxed Stochastic Transitivity (RST): there exists a number $\gamma \geq 1$ such that for all pairs (j, k) with $1 < j < k$, we have

$$\gamma\Delta_{1k} \geq \max\{\Delta_{1j}, \Delta_{jk}\}.$$

In the case of BTM, the constant γ, which measures the degree to which SST fails to hold, needs to be passed to the algorithm explicitly: the higher the γ, the more challenging the

problem, with SST holding when $\gamma = 1$. Given these assumptions, the following regret bounds have been proven for IF [75] and BTM [73]. For large T, we have

$$\mathbb{E}\left[R_T^{\mathrm{IF}_T}\right] \leq C\frac{K\log T}{\Delta_{\min}}, \text{ and}$$
$$R_T^{\mathrm{BTM}_T} \leq C'\frac{\gamma^7 K\log T}{\Delta_{\min}} \text{ with high probability,}$$

where R_T is cumulative regret in the Condorcet setting, defined by Equations (2.5) and (2.3). Moreover, IF_T means that IF is run with the exploration horizon set to T and similarly for BTM_T; $\Delta_{\min}$ is the smallest gap $\Delta_{1j} := p_{1j} - 0.5$, assuming that a_1 is the best arm; and C and C' are universal constants that do not depend on the specific dueling bandit problem.

The first bound holds only when $\gamma = 1$ but matches the lower bound in [75, Theorem 2]. The second bound holds for $\gamma \geq 1$ and is sharp when $\gamma = 1$. This lower bound was proven for certain instances of the K-armed dueling bandit problem that satisfy $\Delta_{1i} = \Delta_{1j}$ for all $i, j \neq 1$.

On the one hand, BTM permits a broader class of K-armed dueling bandit problems than IF; however, it requires γ to be explicitly passed to it as a parameter, which poses substantial difficulties in practice. If γ is underestimated, the algorithm can in certain circumstances be misled with high probability into choosing the Borda winner instead of the Condorcet winner. On the other hand, though overestimating γ does not cause the algorithm to choose the wrong arm, it nonetheless results in a severe penalty, since it makes the algorithm much more exploratory, yielding the γ^7 term in the upper bound on the cumulative regret.

We will now give a description of IF and BTM in order to explain the key ideas that gave rise to these algorithms as well as their weaknesses.

IF: As mentioned before, IF assumes the existence of a total ordering of the arms, in the sense that if arm a_i is preferred to arm a_j and arm a_j is preferred to arm a_k, then we can conclude that a_i is preferred to a_k. Given this rather strong assumption on the dueling bandit problem at hand, a very sensible idea would be to do a form of "hill climbing." More specifically, IF begins by choosing a random arm $\widehat{a}$ as the point of reference and compares it against the other arms until we realize with high probability that $\widehat{a}$ loses to another arm, at which point the algorithm pivots to the latter arm as the point of reference and starts comparing it against the remaining arms. Additionally, the algorithm keeps track of the arms that are beaten by $\widehat{a}$ and eliminates them from consideration, hence reducing the number of comparisons needed to find the best arm.

Let us point out that as long as we assume the existence of a Condorcet winner, this algorithm will eventually converge to it: this is because every arm loses to the Condorcet winner and so they will be eliminated eventually through this process. However, the main flaw of IF is that it can stumble upon an arm $\widehat{a}$ that loses to all other arms by a very tiny margin, while the remaining arms lose to the Condorcet winner by a wide margin, so that the regret accumulated by comparing $\widehat{a}$ against the other arms is large. Recall that the regret incurred by a comparison between

a pair of arms is determined by the extent to which each of them loses to the Condorcet winner, so even if $\widehat{a}$ loses to the Condorcet winner by a small margin, comparing it against an arm other than the Condorcet winner can be costly. Indeed, the main purpose of the Strong Stochastic Transitivity assumption is to rule out such a scenario, hence obtaining a near optimal regret bound for IF.

BTM: In order to explain the idea behind BTM, let us begin by defining the following quantity: given a $K \times K$ preference matrix $\mathbf{P} = [p_{ij}]$, define the *Borda score* of arm a_i as the quantity $\frac{1}{K}\sum_j p_{ij}$, which is the probability with which arm a_i beats a uniformly randomly chosen arm a_j. Now, the key observations behind BTM are the following:

1. First of all, the Borda score of the Condorcet winner is always greater than or equal to 0.5 because by definition the Condorcet winner beats all other arms with probability greater than 0.5, so the Condorcet winner is not a "Borda loser" in the sense that it does not lose against the "average arm" and so as long as we eliminate Borda losers, the Condorcet winner would not be eliminated.
2. Secondly, the other important property of the Condorcet winner of a dueling bandit problem is that it remains the Condorcet winner of any dueling bandit problem obtained by removing any arm other than the Condorcet winner: this is simply because in the smaller dueling bandit problem the Condorcet winner of the larger problem still wins against every other arm with probability greater than 0.5.

Putting these two observations together, we see that as long as we keep eliminating Borda losers, we will eventually be left with nothing but the Condorcet winner by this process of elimination. This is precisely how BTM operates.

2.2.2 SAVAGE

Sensitivity Analysis of VAriables for Generic Exploration (SAVAGE) [65] is an algorithm that outperforms both IF and BTM by a wide margin when the number of arms is of moderate size. Moreover, one version of SAVAGE, called *Condorcet SAVAGE*, makes the Condorcet assumption and has the best theoretical results among the algorithms studied by Urvoy et al. [65, Theorem 3]. However, the regret bounds provided for Condorcet SAVAGE are of the form $\mathcal{O}(K^2 \log T)$, and so are not as tight as those of IF, BTM or our algorithms, presented in subsequent chapters.

Here, we provide a brief description of the SAVAGE family of algorithms, at the core of which is a general scheme that can be applied to a broad class of bandit problems to decide which arms can be safely eliminated from consideration, given a probability δ with which the algorithm is allowed to fail. Rather than speaking in general terms, we will describe two particular instances of this scheme that are relevant to the discussion here, namely Condorcet SAVAGE and Copeland SAVAGE. Both variants of the algorithm compare pairs of arms in a round robin fashion and drop pairs of arms from consideration as soon as it transpires that it is safe to do so, according to the following rules in each case.

Condorcet SAVAGE: If we know that the dueling bandit problem has a Condorcet winner, then any arm that loses with high probability to another arm cannot be a Condorcet winner and so can be eliminated from further consideration. Proceeding in this fashion, we will eventually be left with nothing but the Condorcet winner, which is precisely how Condorcet SAVAGE finds the Condorcet winner.

Copeland SAVAGE: If the goal is to find a Copeland winner, then we can remove an arm from consideration if the most optimistic estimate of its Copeland score is lower than the most pessimistic Copeland score of another arm. In this way, Copeland SAVAGE eliminates arms until all that is left is a collection of Copeland winners. However, in addition to this, Copeland SAVAGE utilizes the following strategy to avoid unnecessary comparisons: for any pair of arms, one of whom beats the other with high probability, we can discontinue comparisons between them, since carrying out more comparisons between such pairs of arms is unlikely to change the Copeland scores of either arms.

Let us point out that the regret bounds for all of IF, BTM and SAVAGE bound only R_T, where T is the predetermined horizon of the experiment. In particular, the horizon T needs to be passed to the algorithm in advance. By contrast, in subsequent chapters, we bound the cumulative regret of our proposed algorithms for *all* time-steps.

2.2.3 Doubler

Doubler, which was proposed by Ailon et al [3], is a method for converting K-armed bandit algorithms into dueling bandit algorithms, under the assumption that the preferences among the arms arise from underlying utilities associated with the arms. More specifically, there are K real numbers $\{u_1, \ldots, u_K\}$, each quantifying the intrinsic quality of the corresponding arm, together with a link function that takes as input a pair of utilities and outputs the probability that one arm beats the other, satisfying the property that the arm with the higher utility beats the one with the lower utility with probability greater than 0.5. This is the so-called utility-based dueling bandit problem [3].

Given this setup, Doubler employs a hill climbing strategy to converge to the best arm, i.e., the one with the highest utility, which is also the Condorcet winner. More precisely, Doubler proceeds in epochs of increasing size, in each of which the left arm is chosen in an i.i.d. manner from the distribution of arms that were chosen for the right arm in the last epoch, while the right arm is chosen using a K-armed bandit algorithm (e.g., UCB); the feedback received by the K-armed bandit algorithm is the wins and losses the right arm encounters when compared against the left arm. In other words, the goal of the right arm is to beat the distribution from which the left arm is sampled from. For a more detailed explanation of the algorithm, the interested reader is referred to [3].

Since the utility assumption induces a total ordering on the arms, and in each epoch the K-armed bandit algorithm tries to do better than its old self in the last epoch, the algorithm eventually converges to the best arm. Indeed, without the total ordering assumption and in the presence of strong cyclical relationships among the arms, Doubler could very easily get stuck in a loop and never converge to the Condorcet winner.

2.2.4 Sparring

Sparring, as proposed by [3], is another, more elegant, method for converting K-armed bandit algorithms into dueling bandit ones, although unlike Doubler there are no known optimal theoretical analyses of Sparring. The key insight is the realization that the dueling bandit problem is a special example of a so-called *symmetric game* [55]. More precisely, we can think of the two arms being chosen to be compared against each other as two opponents engaged in a contest governed by the underlying preference matrix, with the winner of the comparison gaining a reward and the loser incurring a loss. This is related to the so-called *adversarial bandit* problem [7], where in each time-step an adversary chooses the reward of each arm and the goal of the algorithm is to choose arms in such a way that the reward it accumulates is not too much smaller than the reward that it would have accumulated had it chosen any single arm in all of the time-steps. There is a rather extensive body of work on adversarial bandits, spanning multiples decades: the reader is referred to Bubeck and Cesa-Bianchi [12] for a comprehensive survey.

Now, given an algorithm, $\mathcal{A}$, that solves the adversarial bandit problem, we can use it to solve the dueling bandit problem in the following fashion, called Sparring-$\mathcal{A}$: initiate a "row" copy of the algorithm, called $\mathcal{A}_r$, and a "column" copy, called $\mathcal{A}_c$; in each time-step, $\mathcal{A}_r$ proposes a "row" arm, which we denote by a_r, and $\mathcal{A}_c$ proposes a "column" arm, which we call a_c, and the two arms are compared against each other, with the probability of the row arm a_r beating the column arms a_c being p_{rc}; once the comparison has been carried out, the algorithm that proposed the arm that won the comparison receives a reward of 1 and the other side receives a reward of 0. In this setup, each copy of the algorithm plays the role of an adversary for the other, and so if algorithm $\mathcal{A}$ does well against any arbitrary adversary, then both algorithms will converge to the Condorcet winner (if it exists) because the Condorcet winner loses to no other arm on average and so the player who consistently chooses the Condorcet winner will incur the smallest loss against an omniscient adversary, who knows the preference matrix $\mathbf{P}$ precisely; given that, it would also incur small loss against a non-omniscient adversary. In the absence of a Condorcet winner, both players $\mathcal{A}_r$ and $\mathcal{A}_c$ will converge to what is called the von Neumann winner [24].

The astute reader might notice a discrepancy between the last two paragraphs: indeed, the theory of adversarial bandits guarantees that if we make use of an algorithm $\mathcal{A}$ that is designed to function in the adversarial setting, then Sparring-$\mathcal{A}$ will incur small regret, however the regret guarantees obtained in this way take the form $\mathcal{O}(\sqrt{T})$, whereas the regret bounds proven for all of the algorithms discussed so far take the form $\mathcal{O}(\log T)$. What is intriguing, as far as the Sparring style of algorithms are concerned, is that extensive experimentation by various researchers has demonstrated that setting $\mathcal{A}$ to be an algorithm like UCB, produces results that attain logarithmic regret rate [3]. Let us point out that UCB is emphatically not guaranteed to work against an omniscient adversary because UCB is deterministic and so the adversary can simply modify the rewards it assigned to various arms such that the arm that UCB is going to choose in the next round is suboptimal. Therefore, the theory of adversarial bandits does not provide us with any guarantees regarding the performance of Sparring-UCB and indeed, as of the writing of this thesis, no such guarantees have been proven and it remains an interesting, albeit non-trivial, open problem.

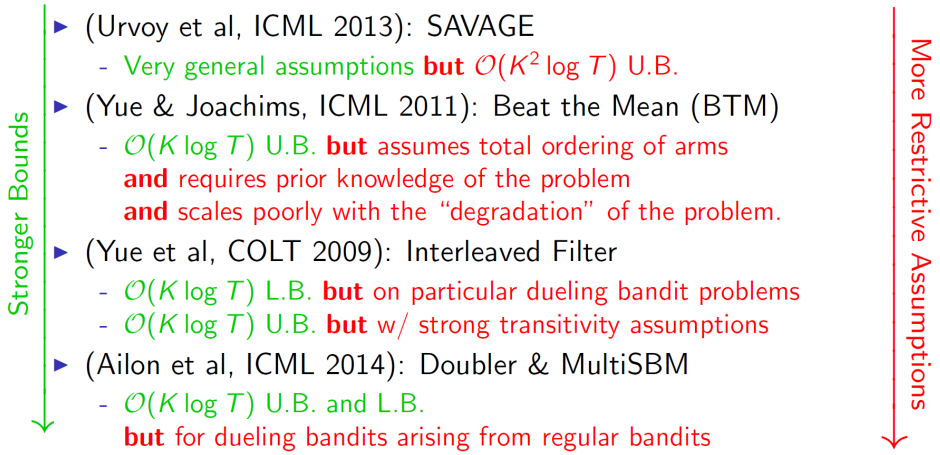

Figure 2.1: A comparison of the assumptions and the results associated with the algorithms discussed so far. In the above description, "U.B." and "L.B." are short for "regret upper-bound" and "regret lower-bound."

2.2.5 Assumptions vs. Results

Let us pause for a moment to insert the following interjection: the algorithms discussed in Sections 2.2.1–2.2.4 were proposed and analyzed before the work presented in this thesis. Furthermore, these results roughly fall into two categories: those with more restrictive assumptions and stronger bounds and those with more general assumptions and weaker bounds. In fact, a more complete picture of the restrictions and the results is provided in Figure 2.1.

Indeed, RUCB [78] and CCB [79], to be presented in Chapters 4 and 7, respectively were the first algorithms to break this dichotomy in the Condorcet and Copeland setting, respectively, in the sense that they are both applicable to a large class of K-armed dueling bandit problems and they come with theoretical guarantees of the form $\mathcal{O}(K^2 + K \log T)$. Furthermore, mergeRUCB, to be presented in Chapter 6, improves upon this in the Condorcet setting by eradicating the quadratic dependence on the number of arm, K, in the additive constant.

In the Copeland setting, the solution was provided by Zohar Karnin using the Scalable Copeland Bandit (SCB) algorithm [79], although SCB has the drawback that it has poor dependence on the gaps of the dueling bandit problem, so the problem of devising a practical algorithm for the Copeland dueling bandit problem that has no quadratic dependence on the number of arms remains an open problem. See Chapter 7.

2.2.6 RMED

More recently, the Relative Minimum Empirical Divergence (RMED) algorithm has been proposed by Komiyama et al. [49] as an algorithm with an optimal asymptotic regret bound, which improves upon the results for RUCB. The authors prove a lower bound on

the cumulative regret of any dueling bandit algorithm, which takes the form

$$R_T \geq \sum_{k=2}^{K} \min_{\{j|p_{ij}<.5\}} \frac{(\Delta_{1i} + \Delta_{1j}) \log T}{2d(p_{ij}, .5)},$$

where $d(p, q) := p \log \frac{p}{q} + (1 - q) \log \frac{1-p}{1-q}$. The upper bound for RMED matches this lower bound asymptotically. Indeed, the algorithm is directly inspired by the lower bound, in the sense that the main quantity that RMED keeps track of measures how far an arm is from RMED's estimate of its optimal number of pulls. Furthermore, despite its asymptotic optimality, the regret bound for RMED has a quadratic dependence on the number of arms. As discussed in Chapter 6, the mergeRUCB algorithm remedies this shortcoming.

2.2.7 Other solution concepts

In addition to the above, bounds have been proven for other notions of winners, including Borda [15, 16, 65], Random Walk [15, 54], and very recently von Neumann [24]. These bounds either rely on restrictive assumptions to obtain a linear dependence on K, the number of arms, or are more broadly applicable, at the expense of a quadratic dependence on K.

A related setting is that of *partial monitoring games* [56], in which an agent chooses at each round an action from a finite set and receives a reward based on an unknown function chosen by an oblivious process. The observed information is a known function of the chosen action and the current oblivious process. One extreme setting in which the observed information equals the reward captures the multi-armed bandit problem. In the other extreme, the observed information equals the entire vector of rewards (for all actions), giving rise to the so-called *full information* game. Our setting is a strict case of partial monitoring as it falls in neither extremes. While a dueling bandit problem can be modeled as a partial monitoring problem, doing so yields weaker results. In particular, most partial monitoring results consider either non-stochastic settings or present problem-independent results. In both cases the regret is lower bounded by $\sqrt{T}$, which is inapplicable to our setting (see [5] for a characterization of partial monitoring problems). Bartók et al. [10] do present problem-dependent bounds from which a logarithmic (in T) bound can be deduced for the dueling bandit problem. However, the dependence on the number of arms K is quadratic, whereas our work achieves a linear dependence in K.

Now that we have presented related work for the K-armed dueling bandit problem, we are ready to present our own solutions. Before doing so, in the next chapter we first present the experimental setup that we will be using in the remainder of the thesis.

3
Experimental Setup

In our experiments, we follow Hofmann [35] and use a setup built on three large-scale learning to rank datasets: the Microsoft Learning to Rank (MSLR), the Yahoo! Learning to Rank Challenge (YLR) and Yandex datasets. The Yahoo! dataset consists of two distinct subsets, Set 1 and Set 2, both of which we use in our experiments. These datasets consist of query-document pairs, each represented by a query id and a feature vector, whose coordinates correspond to features such as BM25, TF.IDF, etc. Additionally, the dataset specifies the relevance of the document to the query using the numbers 0, 1, 2, 3 and 4, where 0 indicates a completely irrelevant document and 4 a highly relevant one. The numerical specifics of these datasets are provided in Table 3.1.

Table 3.1: The specifics of the datasets used.

Datasets	Queries	URLs	Features	Reference
MSLR-WEB30K	31,531	3,771,125	136	[52]
Yandex	9,124	97,290	245	[70]
YLR Set 1	19,944	473,134	519	[18]
YLR Set 2	6,330	172,870	596	[18]

Using these datasets, we create a finite set of rankers, each of which corresponds to a ranking feature provided in the dataset, e.g., PageRank or BM25, and from this set we choose a subset to test our algorithms on. The ranker evaluation task thus corresponds to determining which single feature constitutes the "best" ranker: in the Condorcet case, this corresponds to a ranker that is preferred to all other rankers across the query population, while in the absence of a Condorcet winner, the goal is to find a ranker that is preferred to the highest number of other rankers, i.e., the Copeland winner.

To compare a pair of rankers, we use *Probabilistic Interleave* (PI) [36], though any other interleaved comparison method could be used instead. In broad strokes, the idea of interleaved comparisons is to use the feedback obtained from the users' interaction with the system to compare two rankers using the following procedure: given a query, a set of documents to be ranked, and two rankers r_1 and r_2, apply each ranker to the document set to obtain two lists of documents l_1 and l_2 and then merge these two lists to obtained an *interleaved* list and present this list to the user and use a "credit assignment" method [57] for deciding which ranker it was whose results the user found more relevant.

Numerous methods for carrying out interleaved comparisons have been proposed in the literature, including Balanced Interleave [42, 43], Team-Draft Interleave [57] and Document Constraints [33]. However, as shown in [37], Probabilistic Interleave has a number of desirable properties that make it preferable to the remaining methods for theoretical reasons; however, these advantages are obtained through the introduction of a larger dose of randomness in the interleaving process, which might make PI less suitable in practice.

The astute reader would have noticed while reading the last paragraph that one step in the process of interleaved comparison involves obtaining feedback from a user, which we do not have access to in the academic environment in which this thesis was written. To remedy this issue, we model the user's click behavior on the resulting interleaved lists by employing a probabilistic user model [22, 36] that uses as input the manual labels (classifying documents as relevant or not for given queries) provided with each learning to rank dataset. Queries are sampled randomly and clicks are generated probabilistically by conditioning on these assessments using a user model that resembles the behavior of an actual user [30, 31]. This approach follows an experimental paradigm that has previously been used for assessing the performance of rankers [33, 36–38]. Indeed, by now, there is an extensive literature on such *click* models [21].

We use *cumulative regret* as our main metric for evaluating the performance of our algorithms. Cumulative regret is the total amount of regret encountered by the algorithm until a given time, where the regret incurred by comparing arms a_i and a_j is defined as follows depending on whether or not there exists a Condorcet winner:

$$r = \begin{cases} \frac{\Delta_i + \Delta_j}{2} & \text{if there exists a Condorcet winner} \\ \frac{2\text{Cpld}(a_1) - \text{Cpld}(a_i) - \text{Cpld}(a_j)}{K-1} & \text{if arm } a_1 \text{ is a Copeland winner, but not Condorcet.} \end{cases}$$

Here, the "gap" Δ_k for an arm a_k is defined to be $p_{1k} - .5$, while $\text{Cpld}(a_k)$ denotes its Copeland score (i.e., the number of arms to which a_k is preferred).

Note that a ranker evaluation algorithm accumulates regret whenever it makes a suboptimal choice, meaning that it does not interleave the best ranker with itself. The more suboptimal the rankers in the interleaved comparisons, the higher is the accumulated regret. Thus, according to the cumulative regret minimization objective, the goal of the ranker evaluation algorithm is to increase the frequency with which it chooses the best ranker as soon as possible; doing so results in lower regret curves: the flatter the curve, the lower the frequency of picking poor rankers.

4

Relative Upper Confidence Bound

In this chapter, we will describe our first propsed algorithm, called **Relative Upper Confidence Bounds (RUCB)** [78], which adapts a well-known multi-armed bandit algorithm called Upper Confidence Bounds (UCB) algorithm [8] to the dueling bandit setting. The sections of this chapter are organized as follows: in §4.1, we provide the pseudocode for the algorithm and offer some intuition for its sensibility; in §4.2, we state our theoretical results, bounding the regret accumulated by RUCB; in §4.2, we give detailed proofs of the results stated in the previous section; in §4.4, we provide some experimental results demonstrating the effectiveness of the algorithm; and finally §4.5 contains a summary of the findings discussed in this chapter.

4.1 The Algorithm

We now introduce Relative Upper Confidence Bound (RUCB), which is applicable to any K-armed dueling bandit problem with a Condorcet winner, as defined in Section 2.1.2. In each time-step, RUCB, shown in Algorithm 2, goes through the following three stages:

I. RUCB puts all arms in a pool of potential champions. Then, it compares each arm a_i against all other arms optimistically: for all $i \neq j$, it computes the upper bound $u_{ij}(t) = \mu_{ij}(t) + c_{ij}(t)$, where $\mu_{ij}(t)$ is the frequentist estimate of p_{ij} at time t and $c_{ij}(t)$ is an optimism bonus that increases with t and decreases with the number of comparisons between i and j (Line 4). If $u_{ij} < \frac{1}{2}$ for any j, then a_i is removed from the pool: the set of remaining arms is called $\mathcal{C}$. If we are left with a single potential champion at the end of this process, we let a_c be that arm and put it in the set $\mathcal{B}$ of the hypothesized best arm (Line 9). Note that $\mathcal{B}$ is always either empty or contains one arm; moreover, an arm is demoted from its status as the hypothesized best arm as soon as it optimistically loses to another arm (Line 8). Next, from the remaining potential champions, a champion arm a_c is chosen in one of two ways: if $\mathcal{B}$ is empty, we sample an arm from $\mathcal{C}$ uniformly randomly; if $\mathcal{B}$ is non-empty, the probability of picking the arm in $\mathcal{B}$ is set to $\frac{1}{2}$ and the remaining arms are given equal probability of being chosen (Line 11).

II. Regular UCB is performed using a_c as a benchmark (Line 13), i.e., UCB is performed on the set of arms $a_{1c} \dots a_{Kc}$. Specifically, we select the arm $d = \arg\max_j u_{jc}$.

Algorithm 2 Relative Upper Confidence Bound

Require: $\alpha > \frac{1}{2}, T \in \{1, 2, \ldots\} \cup \{\infty\}$

1: $\mathbf{W} = [w_{ij}] \leftarrow \mathbf{0}_{K \times K}$ // 2D array of wins: w_{ij} is the number of times a_i beat a_j
2: $\mathcal{B} = \varnothing$
3: **for** $t = 1, \ldots, T$ **do**
4: // I: Run an optimistic simulated "tournament":
5: $\mathbf{U} := [u_{ij}] = \frac{\mathbf{W}}{\mathbf{W}+\mathbf{W}^T} + \sqrt{\frac{\alpha \ln t}{\mathbf{W}+\mathbf{W}^T}}$ // All operations are element-wise; $\frac{x}{0} := 1$ for any x.
6: $u_{ii} \leftarrow \frac{1}{2}$ for each $i = 1, \ldots, K$.
7: $\mathcal{C} \leftarrow \left\{a_c \,|\, \forall\, j : u_{cj} \geq \frac{1}{2}\right\}$.
8: If $\mathcal{C} = \varnothing$, then pick c randomly from $\{1, \ldots, K\}$.
9: $\mathcal{B} \leftarrow \mathcal{B} \cap \mathcal{C}$.
10: If $|\mathcal{C}| = 1$, then $\mathcal{B} \leftarrow \mathcal{C}$ and let a_c be the unique element in $\mathcal{C}$.
11: **if** $|\mathcal{C}| > 1$ **then**
12: Sample a_c from $\mathcal{C}$ using the distribution:

$$p(a_c) = \begin{cases} 0.5 & \text{if } a_c \in \mathcal{B}, \\ \frac{1}{2^{|\mathcal{B}|}\,|\mathcal{C} \setminus \mathcal{B}|} & \text{otherwise.} \end{cases}$$

13: // **II**: Run UCB in relation to c:
14: $d \leftarrow \arg\max_j u_{jc}$, with ties broken randomly. Moreover, if there is a tie, d is not allowed to be equal to c.
15: // **III**: Update W
16: Compare arms a_c and a_d and increment w_{cd} or w_{dc} depending on which arm wins.

Ensure: An arm a_c that beats the most arms, i.e., c with the largest count $\#\left\{j | \frac{w_{cj}}{w_{cj}+w_{jc}} > \frac{1}{2}\right\}$.

When $c \neq j$, u_{jc} is defined as above. When $c = j$, since $p_{cc} = \frac{1}{2}$, we set $u_{cc} = \frac{1}{2}$ (Line 5).

Note that, since u_{jc} gives a "home-court" advantage to a_j, a_d is the arm most likely to beat a_c when a_d has the "home-court" advantage. Since $u_{cc} = \frac{1}{2}$, a_c must win all its "away" games to be chosen in stage **II**, whereas it needed to win all of its home games to be chosen in stage **I**.

III. The pair (a_c, a_d) are compared against each other and the score sheet is updated as appropriate (Line 7).

Note that in stage **I** the comparisons are based on u_{cj}, i.e., a_c is compared optimistically to the other arms, making it easier for it to become the champion. By contrast, in stage **II** the comparisons are based on u_{jc}, i.e., a_c is compared to the other arms pessimistically, making it more difficult for a_c to be compared against itself. This is important because comparing an arm against itself yields no information. Thus, RUCB strives to avoid auto-comparisons until there is great certainty that a_c is indeed the Condorcet winner.

Eventually, as more comparisons are conducted, the estimates μ_{1j} tend to concentrate

above $\frac{1}{2}$ and the optimism bonuses $c_{1j}(t)$ become small. Thus, both stages of the algorithm increasingly select a_1, i.e., $a_c = a_d = a_1$, which accumulates zero regret.

Note that Algorithm 2 is a finite-horizon algorithm if $T < \infty$ and a horizonless one if $T = \infty$, in which case the for loop never terminates.

Table 4.1: List of notation used in this section

Symbol	Definition
t	Time
T	The length of time for which the algorithm is run
p_{ij}	The probability that arm i is preferred to arm j
R_T	Regret accumulated in the first T time-steps
K	Number of arms
α	The input of Algorithm 2
$N_{ij}(t)$	Number of comparisons between a_i and a_j until time t
$w_{ij}(t)$	Number of wins of a_i over a_j until time t
$u_{ij}(t)$	$\frac{w_{ij}(t)}{N_{ij}(t)} + \sqrt{\frac{\alpha \ln t}{N_{ij}(t)}}$
$l_{ij}(t)$	$1 - u_{ji}(t)$
δ	Probability of failure
$C(\delta)$	$\left(\frac{(4\alpha-1)K^2}{(2\alpha-1)\delta}\right)^{\frac{1}{2\alpha-1}}$
Δ_j	$p_{1j} - 0.5$
Δ_{ij}	$\frac{\Delta_i + \Delta_j}{2}$
$\Delta_{\max}$	$\max_i \Delta_i$
D_{ij}	$\frac{4\alpha}{\min\{\Delta_i^2, \Delta_j^2\}}$, or $\frac{4\alpha}{\Delta_j^2}$ if $i = 1$, or 0 if $i = j$
D	$\sum_{i<j} D_{ij}$
$\widehat{C}(\delta)$	$\left(4\Delta_{\max} \log \frac{2}{\delta} + 2\Delta_{\max} C\left(\frac{\delta}{2}\right) + 2D \ln 2D\right)$
$\widehat{D}_j$	$\frac{2\alpha\,(\Delta_j + 4\Delta_{\max})}{\Delta_j^2}$
$\widehat{T}_\delta$	Definition 4.3
T_δ	A time between $C(\delta/2)$ and $\widehat{T}_\delta$ when a_1 was compared against itself
$a \vee b$	$\max\{a, b\}$

4.2 Theoretical Results

In this section, we state our finite-time high-probability and expected regret bounds for RUCB: the proofs are provided in §4.3. We first state Lemma 4.1 which will be used to prove a high-probability bound on the number of comparisons for each suboptimal arm in Proposition 4.2. An immediate consequence of this result is a high probability regret bound of the form $\mathcal{O}(K^2 \log T)$, which is similar to the bound for SAVAGE [65] but for the horizonless setting. However, in Theorem 4.4 we show that this can be lowered to $\mathcal{O}(K \log T)$ and we deduce an expected regret bound in Theorem 4.5. This result is proven under conditions that are much more general than those for IF [75] and without requiring the user to specify the γ parameter as BTM does [73]. Moreover, it matches the asymptotic lower bound proven in [75, Theorem 2].

The results in Theorems 4.4 and 4.5 are surprising because a K-armed dueling bandit problem depends on roughly $\frac{K^2}{2}$ independent parameters, so one would expect a bound of the form $O(K^2 \log T)$ unless strong prior information is infused into the algorithm, as with IF and BTM. However, these theorems show that one can get asymptotic behaviour resembling that of a regular K-armed bandit algorithm on a very broad class of dueling bandit problems with very little prior knowledge. This finding is also of great practical significance because there are many situations in which one has a choice between applying a K-armed bandit algorithm to an unreliable quantity, such as Click Through Rate, or using a K-armed dueling bandit algorithm to conduct direct comparisons, which are known to be more reliable when dealing with humans [37, §2.1]. These results show that, given such a dilemma, using a dueling bandit approach does not come at the expense of the asymptotic behaviour.

Finally, note that the high probability bound proven in Theorem 4.4 does not rely on the probability of failure, δ, being passed to the algorithm. Thus, we can use it to also bound higher moments (hence also the variance) of the cumulative regret for RUCB for all times. This is in contrast to high probability bounds that require δ to be specified before the algorithm starts [1, 6, 63], from which one cannot obtain expected regret bounds for all times. While, given a time T, one can set $\delta = 1/T$ in the algorithm to get a logarithmic expected regret bound *at* time T, getting a logarithmic expected regret bound at time $T^{1+\epsilon}$ for any $\epsilon > 0$, requires rerunning the algorithm with $\delta = 1/T^{1+\epsilon}$.

As before, we assume without loss of generality that a_1 is the optimal arm. See Table 4.1 for definitions of symbols used throughout.

Let us begin by stating out main technical lemma that is crucial in all theoretical arguments presented in this thesis and will be used over and over again.

Lemma 4.1. *Let $\mathbf{P} := [p_{ij}]$ be the preference matrix of a K-armed dueling bandit problem with arms $\{a_1, \ldots, a_K\}$. Then, for any dueling bandit algorithm and any $\alpha > \frac{1}{2}$ and $\delta > 0$, we have*

$$P\Big(\forall\, t > C(\delta), i, j,\ p_{ij} \in [l_{ij}(t), u_{ij}(t)]\Big) > 1 - \delta.$$

In more plain terms, the inequality in the above lemma asserts that the probability of a certain desirable event is large: the event is that beyond a certain initial time period (i.e. $C(\delta)$ time-steps), for all pairs of arms a_i and a_j, the confidence intervals $[l_{ij}(t), u_{ij}(t)]$ are "truthful" in the sense that they contain the real preference probability p_{ij}.

Let us now turn to our first high-probability bound:

Proposition 4.2. *Given K arms $\{a_1, \dots, a_K\}$ with preference matrix $\mathbf{P} = [p_{ij}]$, such that a_1 is the Condorcet winner, and $\delta > 0$ and $\alpha > \frac{1}{2}$, then, if we apply Algorithm 2 to this K-armed dueling bandit problem, given any pair $(i, j) \neq (1, 1)$, the number of comparisons between arms a_i and a_j performed up to time t, denoted by $N_{ij}(t)$, satisfies*

$$P\Big(\exists\, t, (i,j) \neq (1,1)\colon N_{ij}(t) > C(\delta) \vee D_{ij} \ln t\Big) < \delta \tag{4.1}$$

and, $N_{ij}^{\delta}(t)$, the number of times a_i was compared against a_j between *time-steps $C(\delta)$ and t, satisfies*

$$P\Big(\exists\, t > C(\delta), (i,j) \neq (1,1)\colon N_{ij}^{\delta}(t) > D_{ij} \ln t\Big) < \delta \tag{4.2}$$

In a similar vein as in Lemma 4.1, the inequalities in Proposition 4.2 state that the probability of a certain undesirable event is small, where the event is that for any pair of arms a_i and a_j, with at least one of them being suboptimal (i.e. not the Condorcet winner a_1), the number of comparisons between them is too large. In other words, it is very likely that the number of comparisons involving suboptimal arms is small, which is what is needed to prove a regret bound because the only comparisons that do not incur regret are the ones where the Condorcet winner is compared against itself.

We use the next definition in what follows:

Definition 4.3. *Let $\widehat{T}_\delta$ be the smallest time satisfying*

$$\widehat{T}_\delta > C\left(\frac{\delta}{2}\right) + \sum_{i<j} D_{ij} \ln \widehat{T}_\delta,$$

which is guaranteed to exist since the expression on the left of the inequality grows linearly with $\widehat{T}_\delta$ and the expression on the right grows logarithmically. Note that $\widehat{T}_\delta$ is specified by the K-armed dueling bandit problem.

With this in hand, we now state our main result, which is a high probability regret bound:

Theorem 4.4. *Given the setup of Proposition 4.2, for any $\delta > 0$, we have with probability $1 - \delta$ that for all times T the following bound on the cumulative regret holds:*

$$R_T \leq \widehat{C}(\delta) + \sum_{j=2}^{K} \widehat{D}_j \ln T, \tag{4.3}$$

where

$$\widehat{C}(\delta) := \left(4 \ln \frac{2}{\delta} + 2C\left(\frac{\delta}{2}\right) + 2D \ln 2D\right) \Delta_{\max}$$
$$\widehat{D}_j := D_{1j}\left(\Delta_{1j} + 2\Delta_{\max}\right) = \frac{2\alpha\left(\Delta_j + 4\Delta_{\max}\right)}{\Delta_j^2},$$

with $C(\cdot)$ and D as in Proposition 4.2, and $\Delta_{\max}:=\max_i \Delta_i$ and $\Delta_{ij}:=\frac{\Delta_i+\Delta_j}{2}$, while R_T is the cumulative regret in the Condorcet case as defined in Section 2.1.2.

Next, we state our expected regret bound, which is a direct consequence of Theorem 4.4:

Theorem 4.5. *Given the setup of Proposition 4.2 together with the notation of Theorem 4.4, we have the following expected regret bound for RUCB, where the expectations are taken across different runs of the algorithm: if we have $\alpha > 1$, the expected regret accumulated by RUCB after T iterations is bounded by*

$$\mathbb{E}[R_T] \leq \left[8 + \left(\frac{2(4\alpha-1)K^2}{2\alpha-1}\right)^{\frac{1}{2\alpha-1}} \frac{2\alpha-1}{\alpha-1}\right]\Delta_{\max} + 2D\Delta_{\max}\ln 2D + \sum_{j=2}^{K} \frac{2\alpha\left(\Delta_j + 4\Delta_{\max}\right)}{\Delta_j^2}\ln T,$$

In the regret bound in Theorem 4.5, note that the only term that grows with T consists of K terms rather than K^2. In other words, the bound is of the form $\mathcal{O}(K\log T)$, rather than $\mathcal{O}(K^2\log T)$.

4.3 Proofs

4.3.1 Proof of Lemma 4.1

In this section, we prove Lemma 4.1, whose statement is repeated here for convenience. Recall from Section 4.2 that we assume without loss of generality that a_1 is the optimal arm. Moreover, given any K-armed dueling bandit algorithm, we define $w_{ij}(t)$ to be the number of times arm a_i has beaten a_j in the first t iterations of the algorithm. We also define

$$u_{ij}(t) := \frac{w_{ij}(t)}{w_{ij}(t)+w_{ji}(t)} + \sqrt{\frac{\alpha\ln t}{w_{ij}(t)+w_{ji}(t)}},$$

where α is any positive contant, and $l_{ij}(t) := 1 - u_{ji}(t)$. Moreover, for any $\delta > 0$, define

$$C(\delta) := \left(\frac{(4\alpha-1)K^2}{(2\alpha-1)\delta}\right)^{\frac{1}{2\alpha-1}}.$$

Lemma 4.1. *Let $\mathbf{P} := [p_{ij}]$ be the preference matrix of a K-armed dueling bandit problem with arms $\{a_1,\ldots,a_K\}$. Then, for any dueling bandit algorithm and any $\alpha > \frac{1}{2}$ and $\delta > 0$, we have*

$$P\Big(\forall\, t > C(\delta), i, j,\ p_{ij} \in [l_{ij}(t), u_{ij}(t)]\Big) > 1-\delta. \tag{4.4}$$

Proof. To decompose the lefthand side of (4.4), we introduce the notation $\mathcal{G}_{ij}(t)$ for the "good" event that at time t we have $p_{ij} \in [l_{ij}(t), u_{ij}(t)]$, which satisfies the following:

(i) $\mathcal{G}_{ij}(t) = \mathcal{G}_{ji}(t)$ because of the three equalities

$$\begin{aligned} p_{ji} &= 1 - p_{ij} \\ l_{ji}(t) &= 1 - u_{ij}(t) \\ u_{ji}(t) &= 1 - l_{ij}(t) \end{aligned}$$

(ii) $\mathcal{G}_{ii}(t)$ always holds, since $(p_{ii}, l_{ii}(t), u_{ii}(t)) = \left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)$. Together with (i), this means that we only need to consider $\mathcal{G}_{ij}(t)$ for $i < j$.

(iii) Define τ_n^{ij} to be the iteration at which arms i and j were compared against each other for the n^{th} time. If $\mathcal{G}_{ij}\left(\tau_n^{ij} + 1\right)$ holds, then the events $\mathcal{G}_{ij}(t)$ hold for all $t \in \left(\tau_n^{ij}, \tau_{n+1}^{ij}\right]$ because when $t \in \left(\tau_n^{ij}, \tau_{n+1}^{ij}\right]$, w_{ij} and w_{ji} remain constant and so in the expressions for $u_{ij}(t)$ and $u_{ji}(t)$ only the $\ln t$ changes, which is a monotonically increasing function of t. So, we have

$$l_{ij}(t) \leq l_{ij}(\tau_n^{ij} + 1) \leq p_{ij} \leq u_{ij}(\tau_n^{ij} + 1) \leq u_{ij}(t).$$

Moreover, the same statement holds with τ_n^{ij} replaced by any $T \in \left(\tau_n^{ij}, \tau_{n+1}^{ij}\right]$, i.e., if we know that $\mathcal{G}_{ij}(T)$ holds, then $\mathcal{G}_{ij}(t)$ also holds for all $t \in \left(T, \tau_{n+1}^{ij}\right]$. This is illustrated in Figure 4.1.

Now, given the above three facts, we have for any T

$$P\Big(\forall\, t \geq T, i, j,\ \mathcal{G}_{ij}(t)\Big) = P\Big(\forall\, i > j,\ \mathcal{G}_{ij}(T) \text{ and } \forall\, n\ s.t.\ \tau_n^{ij} > T,\ \mathcal{G}_{ij}(\tau_n^{ij})\Big). \quad (4.5)$$

Let us now flip things around and look at the complement of these events, i.e. the "bad" event $\mathcal{B}_{ij}(t)$ that $p_{ij} \notin [l_{ij}(t), u_{ij}(t)]$ occurs. Subtracting both sides of Equation (4.5) from 1 and using the union bound gives

$$P\Big(\exists\, t > T, i, j\ s.t.\ \mathcal{B}_{ij}(t)\Big) \leq \sum_{i<j} \left[P\Big(\mathcal{B}_{ij}(T)\Big) + P\Big(\exists\, n : \tau_n^{ij} > T \text{ and } \mathcal{B}_{ij}(\tau_n^{ij})\Big)\right].$$

Further decomposing the righthand side using union bounds and making the condition explicit, we get

$$\begin{aligned} &P\Big(\exists\, t > T, i, j\ s.t.\ \mathcal{B}_{ij}(t)\Big) \\ &\leq \sum_{i>j} \Bigg[P\left(\left|p_{ij} - \mu^{ij}_{N_{ij}(T)}\right| > \sqrt{\frac{\alpha \ln T}{N_{ij}(T)}}\right) + \\ &P\left(\exists\, n \leq T\ s.t.\ \tau_n^{ij} > T \text{ and } \left|p_{ij} - \mu_n^{ij}\right| > \sqrt{\frac{\alpha \ln \tau_n^{ij}}{n}}\right) \\ &\qquad + P\left(\exists\, n > T\ s.t.\ \left|p_{ij} - \mu_n^{ij}\right| > \sqrt{\frac{\alpha \ln \tau_n^{ij}}{n}}\right)\Bigg], \end{aligned}$$

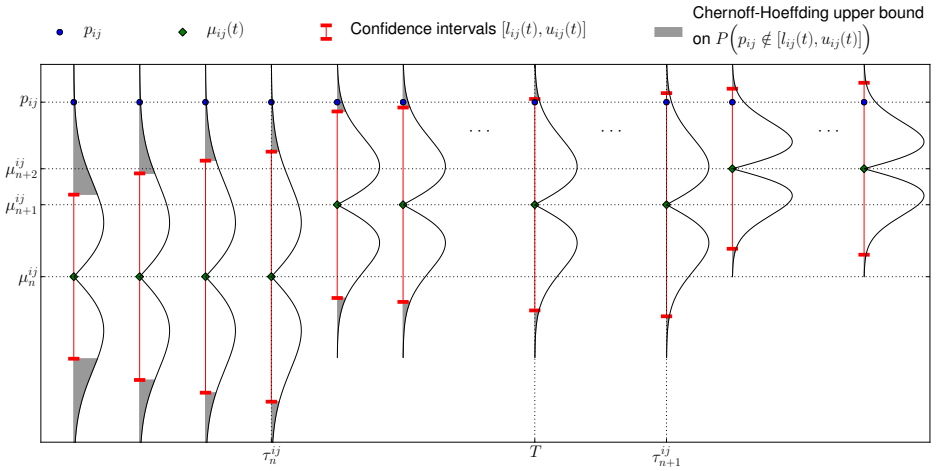

Figure 4.1: An illustrations of the idea behind Lemma 4.1 using an example of how the confidence intervals of a single pair of arms (a_i, a_j), and their relation to the comparison probability p_{ij}, might evolve over time. The time-step τ^{ij}_m denotes the m^{th} time when the arms a_i and a_j were chosen by RUCB to be compared against each other. We also define $\mu^{ij}_m := \mu_{ij}(\tau^{ij}_m)$. The time T is when the confidence intervals $[l_{ij}(t), u_{ij}(t)]$ begin to include p_{ij}. The lemma then states that with probability $1 - \delta$, we have $T \leq C(\delta)$. Moreover, for each time-step, the area of the shaded region under the vertical graphs is the bound given by the Chernoff-Hoeffding (CH) bound on the probability that the confidence interval will not contain p_{ij}. Note that the CH bound has the form $e^{-(x-\mu^{ij}_n)^2}$ and so in order for this number to be the area under a graph (hence making it easier to illustrate in a figure), we have drawn the derivative of this function, $f^{ij}_n(x) := |x - \mu^{ij}_n| e^{-(x-\mu^{ij}_n)^2}$, which is why the graphs are equal to 0 in the middle. Note that this does not mean that μ^{ij}_n has very low probability of being close to p_{ij}: the graphs drawn here are not the PDFs of the posteriors, but simply a manifestation of the bound given by the Chernoff-Hoeffding bound. More specifically, the property that they satisfy is that $P\Big(p_{ij} \notin [l_{ij}(t), u_{ij}(t)]\Big) \leq \int_{-\infty}^{l_{ij}(t)} f^{ij}_{N_{ij}(t)}(x)dx + \int_{u_{ij}(t)}^{\infty} f^{ij}_{N_{ij}(t)}(x)dx$.

since $T < n < \tau^{ij}_n$. Here, $\mu^{ij}_n := \frac{w_{ij}(\tau^{ij}_n)}{w_{ij}(\tau^{ij}_n) + w_{ji}(\tau^{ij}_n)}$ is the frequentist estimate of p_{ij} after n comparisons between arms a_i and a_j.

Now, in the above sum, we can upper-bound the first term by looking at the higher probability event that $\mathcal{B}_{ij}(T)$ happens for any possible number of comparisons between a_i and a_j, and since we know that $N_{ij}(T) \leq T$, we can replace $N_{ij}(T)$ with a variable n that can take values between 0 and T. For the second term, we know that $\tau^{ij}_n > T$, so we can replace τ^{ij}_n with T and remove the condition $\tau^{ij}_n > T$ and look at all $n \leq T$. For the third term, since we always have that $n < \tau^{ij}_n$, we can replace τ^{ij}_n with n and get a higher probability event. Putting all of this together, we get the following looser bound:

$$
\begin{aligned}
P\Big(\exists t > T, i, j \text{ s.t. } \mathcal{B}_{ij}(t)\Big) \leq \sum_{i<j} \Bigg[& P\left(\exists n \in \{0, \dots, T\} : \left|p_{ij} - \mu_n^{ij}\right| > \sqrt{\frac{\alpha \ln T}{n}}\right) \\
& + P\left(\exists n \in \{0, \dots, T\} : \left|p_{ij} - \mu_n^{ij}\right| > \sqrt{\frac{\alpha \ln T}{n}}\right) \\
& + P\left(\exists n > T \text{ s.t. } \left|p_{ij} - \mu_n^{ij}\right| > \sqrt{\frac{\alpha \ln n}{n}}\right) \Bigg] \\
\leq \sum_{i<j} \Bigg[& 2 \sum_{n=0}^{T} P\left(\left|p_{ij} - \mu_n^{ij}\right| > \sqrt{\frac{\alpha \ln T}{n}}\right) \\
& + \sum_{n=T+1}^{\infty} P\left(\left|p_{ij} - \mu_n^{ij}\right| > \sqrt{\frac{\alpha \ln n}{n}}\right) \Bigg]. \qquad (4.6)
\end{aligned}
$$

To bound the expression on line (4.6), we apply the Chernoff-Hoeffding bound, which in its simplest form states that given i.i.d. random variables $X_1, \dots, X_n$, whose support is contained in $[0, 1]$ and whose expectation satisfies $\mathbb{E}[X_k] = p$, and defining $\mu_n := \frac{X_1 + \cdots + X_n}{n}$, we have $P(|\mu_n - p| > a) \leq 2e^{-2na^2}$. This gives us

$$
\begin{aligned}
P\Big(\exists t > T, i, j \text{ s.t. } \mathcal{B}_{ij}(t)\Big) &\leq \sum_{i<j} \left[2 \sum_{n=1}^{T} 2e^{-2\cancel{n}\frac{\alpha \ln T}{\cancel{n}}} + \sum_{n=T+1}^{\infty} 2e^{-2\cancel{n}\frac{\alpha \ln n}{\cancel{n}}} \right] \\
&= \frac{K(K-1)}{2} \left[\sum_{n=1}^{T} \frac{4}{T^{2\alpha}} + \sum_{n=T+1}^{\infty} \frac{2}{n^{2\alpha}} \right] \\
&\leq \frac{2K^2}{T^{2\alpha-1}} + K^2 \int_T^{\infty} \frac{dx}{x^{2\alpha}}, \text{ since } \frac{1}{x^{2\alpha}} \text{ is decreasing.} \\
&\leq \frac{2K^2}{T^{2\alpha-1}} + K^2 \int_T^{\infty} \frac{dx}{x^{2\alpha}} \\
&= \frac{2K^2}{T^{2\alpha-1}} + \frac{K^2}{(1-2\alpha)x^{2\alpha-1}}\Bigg|_T^{\infty} \\
&= \frac{(4\alpha-1)K^2}{(2\alpha-1)T^{2\alpha-1}}. \qquad (4.7)
\end{aligned}
$$

Now, since $C(\delta) = \left(\frac{(4\alpha-1)K^2}{(2\alpha-1)\delta}\right)^{\frac{1}{2\alpha-1}}$ for each $\delta > 0$, the bound in (4.7) gives us (4.4).

□

4.3.2 Proof of Proposition 4.2

Proof. Given Lemma 4.1, we know with probability $1 - \delta$ that $p_{ij} \in [l_{ij}(t), u_{ij}(t)]$ for all $t > C(\delta)$. Let us first deal with the easy case when $i = j \neq 1$: when $t > C(\delta)$ holds,

a_i cannot be played against itself, since if we get $c = i$ in Algorithm 2, then by Lemma 4.1 and the fact that a_1 is the Condorcet winner we have $d \neq i$ since $u_{ii}(t) = \frac{1}{2} < p_{1i} \leq u_{1i}(t)$, where a_d is the second arm chosen by Algorithm 2 on Line 14.

Now, let us assume that distinct arms a_i and a_j have been compared against each other more than $D_{ij} \ln t$ times and that $t > C(\delta)$. If $s \leq t$ is the last time a_i and a_j were compared against each other, we must have

$$\begin{aligned} u_{ij}(s) - l_{ij}(s) &= 2\sqrt{\frac{\alpha \ln s}{N_{ij}(t)}} \\ &\leq 2\sqrt{\frac{\alpha \ln t}{N_{ij}(t)}} < 2\sqrt{\frac{\alpha \ln t}{\frac{4\alpha \ln t}{\min\{\Delta_i^2, \Delta_j^2\}}}} = \min\{\Delta_i, \Delta_j\}. \end{aligned} \tag{4.8}$$

On the other hand, for a_i to have been compared against a_j at time s, one of the following two scenarios must have happened:

I. In Algorithm 2, we had $c = i$ and $d = j$, in which case both of the following inequalities must hold:

   a. $u_{ij}(s) \geq \frac{1}{2}$, since otherwise c could not have been set to i by Line 5 of Algorithm 2, and

   b. $l_{ij}(s) = 1 - u_{ji}(s) \leq 1 - p_{1i} = p_{i1}$, since we know that $p_{1i} \leq u_{1i}(t)$, by Lemma 4.1 and the fact that $t > C(\delta)$, and for $d = j$ to be satisfied, we must have $u_{1i}(t) \leq u_{ji}(t)$ by Line 6 of Algorithm 2.

   From these two inequalities, we can conclude

$$u_{ij}(s) - l_{ij}(s) \geq \frac{1}{2} - p_{i1} = \Delta_i. \tag{4.9}$$

   This inequality is illustrated using the lower right confidence interval in the (a_i, a_j) block of Figure 4.2, where the interval shows $[l_{ij}(s), u_{ij}(s)]$ and the distance between the dotted lines is $\frac{1}{2} - p_{i1}$.

II. In Algorithm 2, we had $c = j$ and $d = i$, in which case swapping i and j in the above argument gives

$$u_{ji}(s) - l_{ji}(s) \geq \frac{1}{2} - p_{j1} = \Delta_j. \tag{4.10}$$

   Similarly, this is illustrated using the lower left confidence interval in the (a_j, a_i) block of Figure 4.2, where the interval shows $[l_{ji}(s), u_{ji}(s)]$ and the distance between the dotted lines is $\frac{1}{2} - p_{j1}$.

Putting (4.9) and (4.10) together with (4.8) yields a contradiction, so with probability $1 - \delta$ we cannot have N_{ij} be larger than both $C(\delta)$ and $D_{ij} \ln t$. This gives us both (4.1) and (4.2).

□

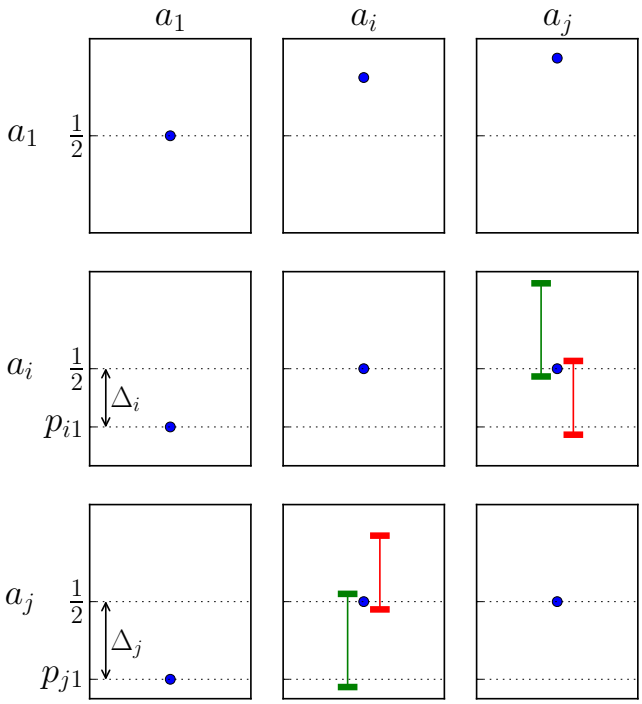

Figure 4.2: An illustration of the proof of Proposition 4.2. The figure shows an example of the internal state of RUCB at time s. The height of the dot in the block in row a_m and column a_n represents the comparisons probability p_{mn}, while the interval, where present, represents the confidence interval $[l_{mn}, u_{mn}]$: we have only included them in the (a_i, a_j) and the (a_j, a_i) blocks of the figure because those are the ones that are discussed in the proof. Moreover, in those blocks, we have included the outcomes of two different runs: one drawn to the left of the dots representing p_{ij} and p_{ji}, and the other to the right (the horizontal axis in these plots has no other significance). These two outcomes are included to address the dichotomy present in the proof. Note that for a given run, we must have $[l_{ji}(s), u_{ji}(s)] = [1 - u_{ij}(s), 1 - l_{ij}(s)]$ for any time s, hence the symmetry present in this figure.

4.3.3 Proof of Theorem 4.4

Proof. If we apply Inequality (4.2) in Proposition 4.2 with $t = \widehat{T}_\delta$ (as in Definition 4.3), we know that with probability $1 - \frac{\delta}{2}$ there is a time $T_\delta \in \left(C\left(\frac{\delta}{2}\right), \widehat{T}_\delta\right]$ when arm a_1 was compared against itself, which means that at that time we had $u_{j1}(T_\delta) < \frac{1}{2}$. This in turn implies that $\mathcal{B} = \{a_1\}$ from that point on, since by Lemma 4.1 we have that $\frac{1}{2} < p_{1j} \leq u_{1j}(t)$ for all $t > T_\delta > C\left(\frac{\delta}{2}\right)$.

Since we have $\mathcal{B} = \{a_1\}$, we know that when choosing a_c in Algorithm 2, the probability of choosing a_1 is equal to $\frac{1}{2}$. Given this, we can expect that from T_δ onwards,

the algorithm will spend roughly half of its time comparing a_1 against other arms. In what follows, we show that this is indeed the case.

Let $\widetilde{N}_{ij}(T)$ denote the number of times arm a_i was compared against a_j between times T_δ and T. Proposition 4.2 shows that, again with probability $1 - \frac{\delta}{2}$, we have $\widetilde{N}_{ij}(T) \leq D_{ij} \ln T$ for all $i < j$: note that this $1 - \frac{\delta}{2}$ is the same as the one used above. In particular, this means that $\widetilde{N}_1(T)$, the number of times between times T_δ and T when we had $c = 1 \neq d$, is bounded by

$$\widetilde{N}_1(T) \leq \sum_{j=2}^{K} \widetilde{N}_{1j}(T) \leq \sum_{j=2}^{K} D_{1j} \ln T =: \widehat{N}_1(T). \tag{4.11}$$

Let us introduce here two sets of random variables:

- $\tau_0, \tau_1, \tau_2, \ldots$, where $\tau_0 := T_\delta$ and τ_l is the l^{th} time arm a_1 was compared against another arm *after* T_δ.
- $n_1, n_2, \ldots$, where n_l is the number of times in Algorithm 2 we had $c \neq 1 \neq d$ between τ_{l-1} and τ_l.

Now, note that RUCB chooses $c \neq 1$ or $d \neq 1$ in time-step t if and only if $u_{j1}(t) \geq \frac{1}{2}$ for some $j > 1$ and that we can have $u_{j1}(t+1) < u_{j1}(t)$ only if at the end of the t^{th} iteration, arm a_1 was compared against arm a_j. In other words, whenever we have $u_{j1}(T) \geq \frac{1}{2}$ for some $j > 1$, the algorithm will continue to set $(c, d) \neq (1, 1)$ until all of the u_{j1} with $j > 1$ get submerged below $\frac{1}{2}$ and that the last comparison before we get to this state must be between a_1 and another arm. With this picture in mind, with probability $1 - \frac{\delta}{2}$, we have

$$R_T \leq T_\delta \Delta_{\max} + \sum_{j=2}^{K} D_{1j} \Delta_{1j} \ln T + \sum_{l=1}^{\widehat{N}_1(T)} n_l \Delta_{\max}, \tag{4.12}$$

where $\widehat{N}_1(T)$ is as in Inequality (4.11), and so all we need to do is bound T_δ and the sum of the intervals n_l for $l = 1, \ldots, \widehat{N}_1(T)$. Let us deal with the former first: we know that $T_\delta \leq \widehat{T}_\delta$ and that the latter is defined to be the smallest time-step satisfying the inequality in Definition 4.3, so all we need to do is produce one number that, when plugged in for $\widehat{T}_\delta$, satisfies the inequality, and one such number is $2C\left(\frac{\delta}{2}\right) + 2D \ln 2D$. To see this, let us temporarily use the notation $C := C\left(\frac{\delta}{2}\right)$, and use the concavity of the log function, a first order Taylor expansion, and the fact that we have $\ln x < x$ for any x, to get

$$\begin{aligned} C + D \ln(2C + 2D \ln 2D) &\leq C + D \ln(2D \ln 2D) + \cancel{D} \frac{\cancel{2}C}{\cancel{2D} \ln 2D} \\ &\leq C + D \ln(2D)^2 + C = 2C + 2D \ln 2D, \end{aligned}$$

where we used the fact that $D > 2$ and so $\ln 2D > 1$.

Let us now return to the task of bounding the sum of the intervals n_l. To do so, we introduce the random variables $\widehat{n}_1, \widehat{n}_2, \ldots$, which are independent samples from the

geometric distribution with decay $\frac{1}{2}$. Note that $\widehat{n}_l$ bounds n_l from above since it counts the number of iterations it would take for Line 11 of Algorithm 2 to produce a_1 and once we have $c = 1$, we are guaranteed to have a comparison between a_1 and another arm, as long as $u_{j1} \geq \frac{1}{2}$ for some $j > 1$. Furthermore, the sum of independent geometric random variables has a negative binomial distribution [26, §VI.8], with the following probability mass function, cf. [26, Equation VI.8.1]:

$$f(n;r) := P\left(\sum_{l=1}^{r} \widehat{n}_l = n\right) = \frac{\binom{n+r-1}{n}}{2^{n+r}},$$

where in our case $p = \frac{1}{2}$ and so it is eliminated from the notation of the PMF. In order to bound this sum with high probability, we note that when $n \geq 2r$, then we have

$$\frac{f(n;r)}{f(n+1;r)} = \frac{\frac{\binom{n+r-1}{n}}{2^{n+r}}}{\frac{\binom{n+r}{n+1}}{2^{n+r+1}}} = \frac{\frac{(n+r-1)!}{n!(r-1)!}}{\frac{(n+r)!}{(n+1)!(r-1)!\times 2}}$$

$$= \frac{2(n+1)}{n+r} = 2\left[1 - \frac{r-1}{n+r}\right] \geq 2 - \frac{2r-2}{3r} > \frac{4}{3}.$$

Therefore, we have $f(n;r) \leq f(2r;r)\left(\frac{3}{4}\right)^{n-2r} \leq \left(\frac{3}{4}\right)^{n-2r}$ for all $n \geq 2r$, since $f(2r;r)$ is a probability and so at most equal to 1. From this we can conclude that with probability $1 - \frac{\delta}{2}$, we have $n \leq 2r + \dfrac{\ln\frac{\delta}{2}}{\ln\frac{3}{4}} < 2r - 4\ln\frac{\delta}{2}$: note that both the numerator and the denominator of the second summand are negative and so the fraction is positive. Now, setting $r = \widehat{N}_1(T) := \sum_{j=2}^{K} D_{1j} \ln T$ and plugging the resulting upper bound into the regret bound given in (4.12) give us the desired result. □

4.3.4 Proof of Theorem 4.5

Here, we provide the proof of the expected regret bound claimed in Theorem 4.5, starting by repeating the statement of the theorem:

Theorem 4.5. *Given the setup of Proposition 4.2 together with the notation of Theorem 4.4, we have the following expected regret bound for RUCB, where the expectations are taken across different runs of the algorithm: if we have $\alpha > 1$, the expected regret accumulated by RUCB after T iterations is bounded by*

$$\mathbb{E}[R_T] \leq \left[8 + \left(\frac{2(4\alpha-1)K^2}{2\alpha-1}\right)^{\frac{1}{2\alpha-1}} \frac{2\alpha-1}{\alpha-1}\right]\Delta_{\max}$$

$$+ 2D\Delta_{\max}\ln 2D + \sum_{j=2}^{K} \frac{2\alpha\left(\Delta_j + 4\Delta_{\max}\right)}{\Delta_j^2}\ln T. \tag{4.13}$$

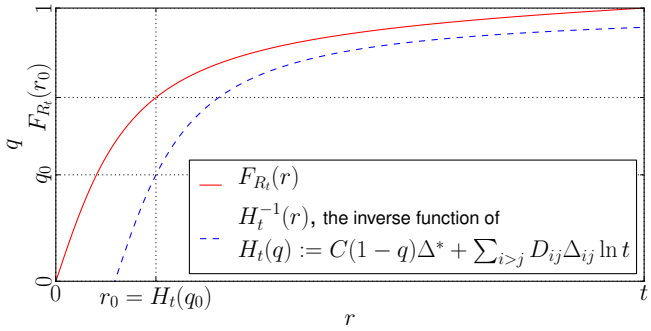

Figure 4.3: A schematic graph illustrating the proof of Theorem 4.5. Note that the expression for $H_T(q)$ is extracted from (4.3), which also implies that H_T^{-1} is necessarily below F_{R_T}: formulated in terms of CDFs, (4.3) states that $F_{R_T}\left(H_T(q_0)\right) > q_0 = H_T^{-1}\left(H_T(q_0)\right)$, where $q_0 = 1 - \delta_0$ is a quantile. From this, we can conclude that $F_{R_T}(r) > H_T^{-1}(r)$ for all r.

Proof. We can obtain the bound in (4.13) from (4.3) by integrating with respect to δ from 0 to 1. This is because given any one-dimensional random variable X with CDF F_X, we can use the identity $\mathbb{E}[X] = \int_0^1 F_X^{-1}(q)dq$. In our case, $X = R_T$ for a fixed time t and, as illustrated in Figure 4.3, we can deduce from (4.3) that $F_{R_T}(r) > H_T^{-1}(r)$, which gives the bound

$$F_{R_T}^{-1}(q) < H_T(q) = \widehat{C}(1-q) + \sum_{j=2}^{K} \widehat{D}_j \ln T.$$

Now, assume that $\alpha > 1$. To derive (4.13) from the above inequality, we need to integrate the righthand side, and since it is only the first two terms in the definition of $\widehat{C}$ that depends on δ, that is all we need to integrate. Let us deal with the first term first, using the substitution $1 - q = \delta$, $dq = -d\delta$:

$$\begin{aligned}\int_{q=0}^{1} 4\Delta_{\max} \ln \frac{2}{1-q} dq &= 4\Delta_{\max}\left[\ln 2 - \int_{\delta=1}^{0} -\ln \delta \, d\delta\right] \\ &= 4\Delta_{\max}\left[\ln 2 - \int_{\delta=0}^{1} \ln \delta \, d\delta\right] \\ &= 4\Delta_{\max}(\ln 2 + 1) < 8\Delta_{\max}\end{aligned}$$

To deal with the second term in $\widehat{C}$, recall that it is equal to

$$2\Delta_{\max} C\left(\frac{\delta}{2}\right) := 2\Delta_{\max}\left(\frac{2(4\alpha-1)K^2}{(2\alpha-1)\delta}\right)^{\frac{1}{2\alpha-1}},$$

so to simplify notation, we define

$$L := 2\Delta_{\max}\left(\frac{2(4\alpha-1)K^2}{2\alpha-1}\right)^{\frac{1}{2\alpha-1}}.$$

Now, we can carry out the integration as follows, again using the substitution $1 - q = \delta$, $dq = -d\delta$:

$$\begin{aligned}
\int_{q=0}^{1} C(1-q)dq &= \int_{\delta=1}^{0} -C(\delta)d\delta \\
&= \int_{0}^{1} 2\left(\frac{2(4\alpha-1)K^2}{(2\alpha-1)\delta}\right)^{\frac{1}{2\alpha-1}} d\delta \\
&= L\int_{0}^{1} \delta^{-\frac{1}{2\alpha-1}} d\delta \\
&= L\left[\frac{\delta^{1-\frac{1}{2\alpha-1}}}{1-\frac{1}{2\alpha-1}}\right]_{0}^{1} \\
&= \left(\frac{2(4\alpha-1)K^2}{2\alpha-1}\right)^{\frac{1}{2\alpha-1}} \frac{2\alpha-1}{\alpha-1}.
\end{aligned}$$

□

4.4 Experimental Results

To evaluate RUCB, we apply it to the problem of *ranker evaluation* from the field of *information retrieval* (IR) [51]. A ranker is a function that takes as input a user's search query and ranks the documents in a collection according to their relevance to that query. Ranker evaluation aims to determine which among a set of rankers performs best. One effective way to achieve this is to use *interleaved comparisons* [57], which interleave the documents proposed by two different rankers and presents the resulting list to the user, whose resulting click feedback is used to infer a noisy preference for one of the rankers. Given a set of K rankers, the problem of finding the best ranker can then be modeled as a K-armed dueling bandit problem, with each arm corresponding to a ranker.

We evaluated RUCB, Condorcet SAVAGE and BTM using randomly chosen subsets from the pool of 64 rankers provided by LETOR, a standard IR dataset, discussed in greater detail in Section 4.4.1, yielding K-armed dueling bandit problems with $K \in \{16, 32, 64\}$. For each set of rankers, we performed 100 independent runs of each algorithm for a maximum of 4.5 million iterations. For RUCB we set $\alpha = 0.51$, which approaches the limit set by our high-probability result. Since BTM and SAVAGE require the exploration horizon as input, we ran BTM_T and CSAVAGE_T for various horizons T ranging from 1000 to 4.5 million. In the plots in Figure 4.4, the markers on the green and the blue curves show the regret accumulated by BTM_T and CSAVAGE_T in the first T iteration of the algorithm for each of these horizons. Thus, each marker corresponds, not to the continuation of the runs that produced the previous marker, but to new runs conducted with a larger T.

Since RUCB is horizonless, we ran it for 4.5 million iterations and plotted the cumulative regret, as shown using the red curves in the plots in Figure 4.4. For all three algorithms, the middle curve shows average cumulative regret and the dotted lines show minimum and maximum cumulative regret across runs. Note that these plots are in log-linear scale,

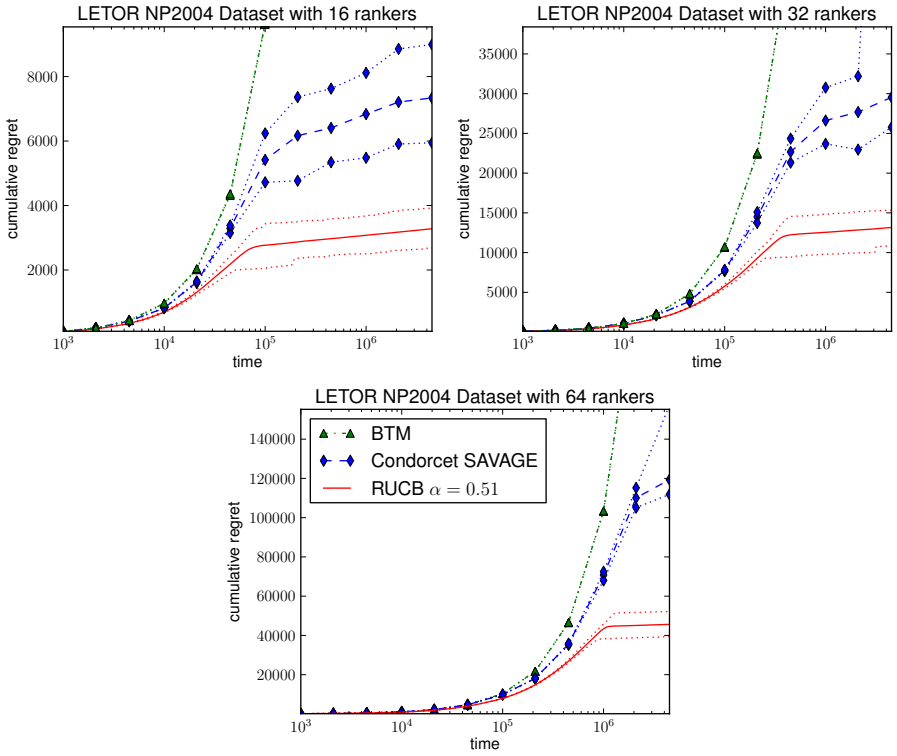

Figure 4.4: Average cumulative regret for 100 runs of BTM, Condorcet SAVAGE and RUCB with $\alpha = 0.51$ applied to three K-armed dueling bandit problems with $K = 16, 32, 64$. Note the time axis uses a log scale, so that the curves depict the relation between $\log T$ and R_T; also, the dotted curves signify best and worst regret performances across all runs.

so they depict the relation between R_T and $\log T$, which can be seen to be asymptotically linear. The regret curves for BTM are cut-off in these plots, since in all three experiments $R_T^{BTM_T}$ grew linearly with T in the first 4.5 million iterations. As can be seen from the plots in Figure 4.4, RUCB accumulates the least regret of the three algorithms: the average regret accumulated by RUCB is less than half of that of Condorcet SAVAGE by the end of each of the three experiments and even the worst performing run of RUCB accumulated considerably less regret than the best performing run of Condorcet SAVAGE.

Finally, the plots in Figure 4.5 show the accuracy of all three algorithms across 100 runs, computed at the same times as the exploration horizons used for BTM and SAVAGE in Figure 4.4. Note that RUCB reaches the 80% mark almost twice as fast as Condorcet SAVAGE, all without knowing the horizon T. The contrast is even more stark when comparing to BTM.

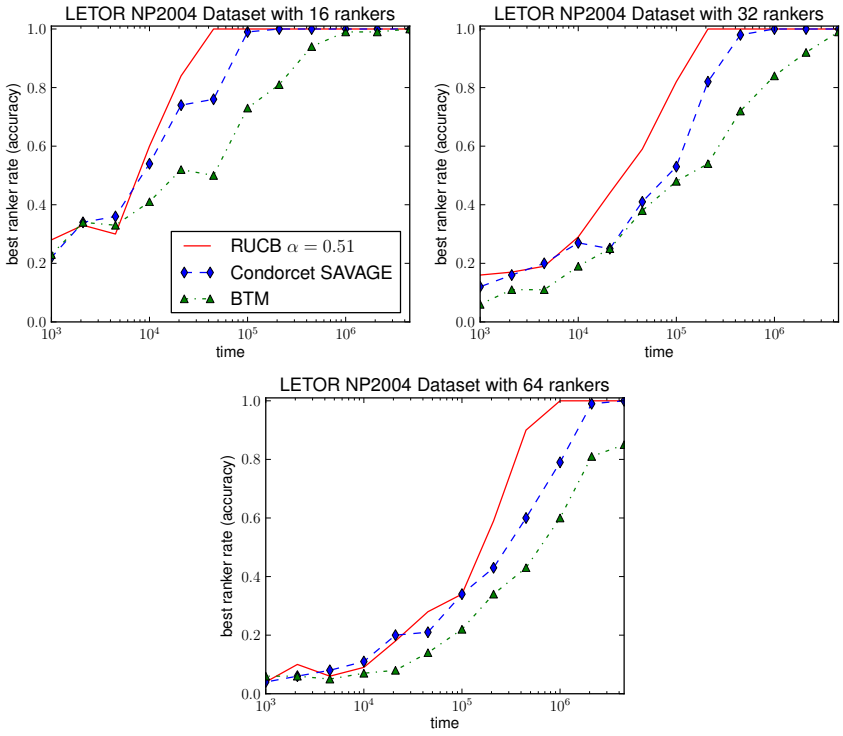

Figure 4.5: Average accuracy for 100 runs of BTM, Condorcet SAVAGE and RUCB with $\alpha = 0.51$ applied to three K-armed dueling bandit problems with $K = 16, 32, 64$. Note that the x-axes in these plots use a log scale.

4.4.1 Details of the Experimental Setup

Our experimental setup is built on real IR data, namely the LETOR NP2004 dataset [50]. This dataset is based on the TREC Web track named-page finding task, where a query is what the user believes to be a reasonable estimate of the name of the webpage she is seeking. Using this data set, we create a set of 64 rankers, each corresponding to a ranking feature provided in the data set, e.g., PageRank. The ranker evaluation task in this context corresponds to determining which single feature constitutes the best ranker [38].

To compare a pair of rankers, we use *probabilistic interleave* (PI) [36], a recently developed method for interleaved comparisons. To model the user's click behavior on the resulting interleaved lists, we employ a probabilistic user model [22, 36] that uses as input the manual labels (classifying documents as relevant or not for given queries) provided with the LETOR NP2004 dataset. Queries are sampled randomly and clicks are generated probabilistically by conditioning on these assessments in a way that resembles the behavior of an actual user [31].

Following [73], we first used the above approach to estimate the comparison probabilities p_{ij} for each pair of rankers and then used these probabilities to simulate comparisons

between rankers. More specifically, we estimated the full preference matrix by performing 4000 interleaved comparisons on each pair of the 64 feature rankers.

4.5 Summary

In this chapter, we proposed a new method called Relative Upper Confidence Bound (RUCB) for the *K-armed dueling bandit problem* that extends the Upper Confidence Bound (UCB) algorithm to the relative setting by using optimistic estimates of the pairwise probabilities to choose a potential champion and conducting regular UCB with the champion as the benchmark.

We proved finite-time high-probability and expected regret bounds for RUCB that match an existing lower bound and evaluated it empirically in an information retrieval application. Unlike existing results, our regret bounds hold for all time-steps, rather than just a specific horizon T input to the algorithm. Furthermore, they take the form $\mathcal{O}(K \log T)$ while making much less restrictive assumptions than existing algorithms with similar bounds. Finally, the empirical results showed that RUCB greatly outperforms state-of-the-art methods.

There are two natural extensions to this research that one could consider for further investigation. First, building off extensions of UCB to the continuous bandit setting [13, 23, 53, 63, 66], one could extend RUCB to the continuous dueling bandit setting, without a convexity assumption as in [40, 71]. Second, building off Thompson Sampling [2, 44, 64], an elegant and effective sampling-based alternative to UCB, one could investigate whether a sampling-based extension to RUCB would be amenable to theoretical analysis. Both these extensions involve overcoming not only the technical difficulties present in the regular bandit setting, but also those that arise from the two-stage nature of RUCB. The latter of these two ideas has been validated experimentally in [77], which we describe in Chapter 5, although a theoretical analysis is still lacking. Very recently, a modification of RUCB has been proposed in [69] that uses Thompson Sampling to choose the optimistic Condorcet winner more effectively, although the remaining choices are made using confidence bounds.

5 Relative Confidence Sampling

In this chapter, we present the **Relative Confidence Sampling** (RCS) algorithm, which aims to reduce cumulative regret by being less conservative than RUCB about eliminating arms from contention. While RCS is related to RUCB, it differs in one crucial respect: in the first phase of the algorithm, when a potential Condorcet winner is being chosen, RCS uses *sampling* to conduct the round-robin tournament. The goal in doing so is to exploit one of the key lessons that has been learned in the study of regular K-armed bandits: that much better performance can be obtained by maintaining posterior distributions over the expected value of each arm and sampling from those posteriors to determine which arm to select. This is evidenced by the superior performance of Thompson Sampling [2, 44, 64], a K-armed bandit method that employs such sampling, over various UCB-type algorithms [19, 44].

Unlike the UCB family of algorithms, which rely on surrogates for the means of the arms that are significantly different from those means themselves, sampling-based methods rely on samples drawn from posteriors that tend not to gravitate toward the extremes, leading to more appropriate choices being made more frequently. In the particular case of RUCB, the algorithm tends to be very conservative in its choice of a potential Condorcet winner, in the sense that unless it is highly confident that an arm a_i is inferior to another arm, it will go on considering a_i as a potential Condorcet winner. By contrast, RCS is less timid in its choices: the more an arm beats the rest, the greater its chances of being chosen as a Condorcet winner to be compared against the rest.

The remainder of this chapter is organized as follows: in §5.1, we provide and explain the pseudo-code for RCS; in §5.2, we describe the experiments carried out to compare RCS against other dueling bandit algorithms, present the results and offer a detailed discussion; in §5.3, we summarize the findings in this chapter.

5.1 The Algorithm

The RCS algorithm, described in Algorithm 3, takes as input a set of arms and an oracle such as an interleaved comparison method that can compare these arms and return a noisy estimate of which is the winner. As it is horizonless, RCS does not have an output: as time goes by it chooses the best arm more and more frequently. RCS has one parameter α (Line 1), which controls how exploratory the algorithm's behavior is: the higher the value of α, the more slowly the algorithm settles on a single arm. RCS maintains a

Algorithm 3 Relative Confidence Sampling (RCS)

Require: A set of arms $a_1, \ldots, a_K$, a number $\alpha > 1/2$ and an oracle that can take a pair of arms and return one as the winner (e.g., an interleaved comparison method)

1: Choose $\alpha > \frac{1}{2}$
2: $\mathbf{W} \leftarrow \mathbf{0}_{K \times K}$ // 2D array of wins: $\mathbf{W}_{ij}$ is the number of times a_i beat a_j
3: **for** $t = 1, 2, \ldots$ **do**
4: // **I**: Run a simulated "tournament":
5: $\mathbf{\Theta}(t) \leftarrow \frac{\mathbf{1}_{K \times K}}{2}$
6: **for** $i, j = 1, \ldots, K$ with $i < j$ **do**
7: $\mathbf{\Theta}_{ij}(t) \sim \text{Beta}(\mathbf{W}_{ij} + 1, \mathbf{W}_{ji} + 1)$
// Here $\text{Beta}(\alpha, \beta)$ is the Beta distribution with non-negative parameters α and β
8: $\mathbf{\Theta}_{ji}(t) = 1 - \mathbf{\Theta}_{ij}(t)$
9: Pick c such that $\mathbf{\Theta}_{cj}(t) \geq \frac{1}{2}$ for all j. If no such arm exists, pick the arm that has been chosen champion least often.
10: // **II**: Run UCB in relation to c:
11: $\mathbf{U}(t) = \frac{\mathbf{W}}{\mathbf{W}+\mathbf{W}^T} + \sqrt{\frac{\alpha \ln t}{\mathbf{W}+\mathbf{W}^T}}$
// All operations are element-wise, with $\frac{x}{0} := 1$ for any x.
12: $\mathbf{U}_{ii}(t) \leftarrow \frac{1}{2}$ for each $i = 1, \ldots, K$.
13: $d \leftarrow \arg\max_j \mathbf{U}_{jc}(t)$
14: // **III**: Update $\mathbf{W}$
15: Compare arms a_c and a_d and increment either $\mathbf{W}_{cd}$ if a_c beat a_d or $\mathbf{W}_{dc}$ otherwise.

scoresheet $\mathbf{W}$ (Line 2), in which it records the comparison results and proceeds in two phases:

I: A tournament is simulated based on the current scoresheet, i.e., samples $\mathbf{\Theta}_{ij}$ are collected for each pair of arms (i, j) with $i > j$, from the posterior Beta distribution maintained on p_{ij}; Since $p_{ji} = 1 - p_{ij}$, RCS sets $\mathbf{\Theta}_{ji} = 1 - \mathbf{\Theta}_{ij}$ (Lines 6-9). Also, RCS sets $\mathbf{\Theta}_{ii} = \frac{1}{2}$ for each arm i, since $p_{ii} = \frac{1}{2}$ and thus its posteriors are all concentrated at $\frac{1}{2}$ (Line 5). Given these sampled results, arm i beats arm j in the simulated tournament if $\mathbf{\Theta}_{ij} > \frac{1}{2}$ for $i \neq j$. There are two possibilities at this stage (Line 10):

1. There is a champion arm c that beats all other arms in this tournament, i.e., $\mathbf{\Theta}_{cj} > \frac{1}{2}$ for all $j \neq c$.
2. No arm beats all other arms, in which case RCS sets $c = \text{argmin}_i N_i$, where N_i is the number of times arm i was previously chosen as champion. In other words, c is the arm that has been the champion least often.

Eventually, once the Condorcet winner has been compared against the rest of the arms often enough, its superiority over the rest will cause their elimination in this phase of the algorithm. So, as times goes by, c will be the Condorcet winner more and more often.

II: The UCB algorithm is applied to the K-armed bandit problem with means $\{p_{1c}, \ldots, p_{Kc}\}$ (Lines 12-14). In other words, for each $j \in \{1, \ldots, K\}$, we calculate the optimistic

estimate

$$u_{jc} := \frac{\mathbf{W}_{jc}}{\mathbf{W}_{jc} + \mathbf{W}_{cj}} + \sqrt{\frac{\alpha \ln t}{\mathbf{W}_{jc} + \mathbf{W}_{cj}}},$$

where the first term is our estimate of the comparison probability p_{jc} and the second term is a confidence radius that is added to ensure adequate exploration by allowing the other arms to compare themselves against c optimistically. RCS picks the arm d for which u_{dc} is higher than all other u_{jc}.

The astute reader would have noticed that this aspect of RCS closely resembles the corresponding component in RUCB and indeed the reason for that similarity is simply that now that arm a_c has been chosen as a potential Condorcet winner, the arms can be thought of as forming a multi-armed bandit problem, where the mean of each arm is its probability of beating a_c; given this, a natural choice is to apply UCB to this problem, which is what this step does.

Finally, arms c and d are compared against each other using a real interleaved comparison and $\mathbf{W}$ is updated accordingly (Line 16).

To better understand the rationale behind RCS, consider the following example from the world of tennis: suppose we wish to use a K-armed dueling bandit method to efficiently identify the world's best tennis player. In addition, suppose that Rafael Nadal is likely to win, say with probability 0.55, a match against the other top players since, according to The New York Times (June 9, 2013) "Nadal is also the only member of the so-called Big Four to have a head-to-head edge over all the other members of that club: [Novak] Djokovic, Roger Federer and Andy Murray." Finally, suppose also that, though Federer loses in expectation to Nadal, he has a 0.75 probability of beating Djokavic and Murray. Thus, Nadal is the Condorcet winner but Federer is the Borda winner.

In this example, a key danger is that a K-armed dueling bandits algorithm may mistakenly conclude that Federer is the champion and stop comparing him against other players, most importantly Nadal. RCS avoids this pitfall in two ways. First, if Nadal has not been compared against the others enough times, then $\mathbf{\Theta}_{NF}$, where N is Nadal and F is Federer, will have high variance so that in Phase **I**, Federer will have difficulty beating the others in the simulated tournament in order to be chosen as the champion. In cases where no player beats everyone in the simulated tournament, RCS ensures that everyone gets a chance at being the champion, including Nadal. This ensures that our posterior belief in Nadal's chance of beating all other players will further concentrate above 0.5, making it even more likely that he will win future simulated tournaments. Second, even in cases where Federer does win the simulated tournament in Phase **I**, the fact that Nadal and Federer have not been compared often means that the upper bound $\mathbf{U}_{NF}$ will very likely be above $\mathbf{U}_{FF} = \frac{1}{2}$, hence preventing a fruitless comparison between Federer and himself. Instead, Federer will be compared to Nadal, ensuring that Nadal's superiority is eventually discovered.

5.2 Experiments

In this section we present the results of four sets of experiments that we designed to answer the following four questions:

Q1 How do SAVAGE and RCS perform at ranker evaluation on large-scale learning to rank datasets in terms of the accuracy of the predicted Condorcet winner?

Q2 How do RUCB and RCS perform on these datasets in terms of cumulative regret?

Q3 How does RUCB perform when the parameter α is too small for its theoretical guarantees to hold? Moreover, how does RCS perform in the same range?

Q4 How do RUCB and RCS scale as the number of arms grows?

We perform these experiments by performing interleaved comparisons between pairs of arms using Probabilistic Interleave (PI) on the MSLR and Yahoo! Learning to Rank Challenge datasets, as described in Chapter 3. The answers to these questions are provided in the following four subsections.

5.2.1 Accuracy Results

The performance of RCS in terms of accuracy was compared against Condorcet SAVAGE, which was the state of the art in according to this metric [65]. The left plots in Figure 5.1 compare the accuracy of RCS and Condorcet SAVAGE at 10 different horizons on three 10-armed bandits obtained from the three different datasets by considering 10 of the feature arms. Note that the horizontal axis is in log scale and accuracy is the percentage of the runs that correctly produced the best arm as the winner at the given horizon.

Note that, because Condorcet SAVAGE requires the horizon as input, the algorithm must be rerun from scratch for each horizon. Thus, to obtain the plotted results, we conducted independent runs for each of the 10 horizons considered. By contrast, since RCS is a horizonless algorithm, we simply ran it until the longest horizon and then measured its accuracy at each of the 10 horizons.

Concerning **Q1**, these results show that RCS has consistently higher accuracy than Condorcet SAVAGE on all datasets. This is a particularly striking result because RCS does not have any parameters optimized for the specific horizons in these experiments. Condorcet SAVAGE, by contrast, has the advantage that it receives the horizon as input and can thus adapt its behavior accordingly. Nonetheless, RCS outperforms Condorcet SAVAGE according to the metric for which Condorcet SAVAGE was designed. Though the lines appear close in the graph due to the log scale, the learning speed actually differs substantially: RCS reaches the same level of accuracy almost twice as quickly as Condorcet SAVAGE. For example, on the Yahoo! Set 1, RCS achieves an accuracy of 0.8 after 1000 steps while SAVAGE achieves it only after 2000 steps.

We also provide the regret accumulated by each algorithm in the right column of plots in Figure 5.1. Note that these plots are in log-log scale. As with the accuracy plots, these plots select the performance of separate runs conducted at each horizon for Condorcet SAVAGE, while each RCS run was used to measure regret at all horizons. As is clear from these plots, RCS dramatically outperforms Condorcet SAVAGE despite being ignorant of the predetermined horizon. In fact, in each experiment, by the time Condorcet SAVAGE reaches 100% accuracy, it has accumulated at least 3 times as much regret as when RCS achieves that same accuracy.

The superior performance of RCS over Condorcet SAVAGE according to both accuracy and regret is due to related phenomena: the main advantage of RCS over Condorcet

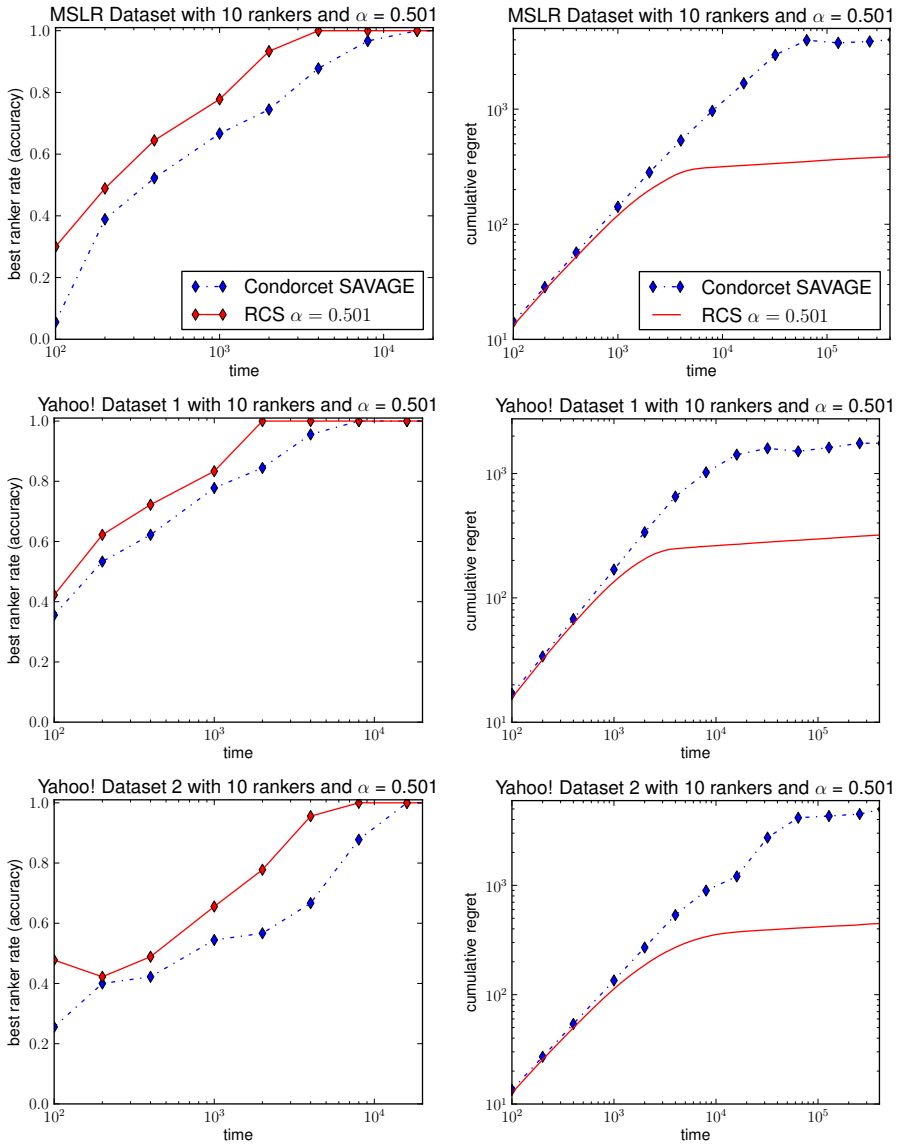

Figure 5.1: Accuracy (best arm rate) and average cumulative regret over 90 runs; in the plots in the left column, the x-axis uses a log scale; in the plots in the right column, both axes use log scales. Log scales were used for the x-axes to make the accuracy plots easier to read and in the y-axes of the regret plots since the regret accumulated by Condorcet SAVAGE is an order of magnitude higher than that of RCS.

SAVAGE is that, instead of comparing pairs of arms uniformly randomly, as is the case with Condorcet SAVAGE, it rapidly focuses on comparing the Condorcet winner against

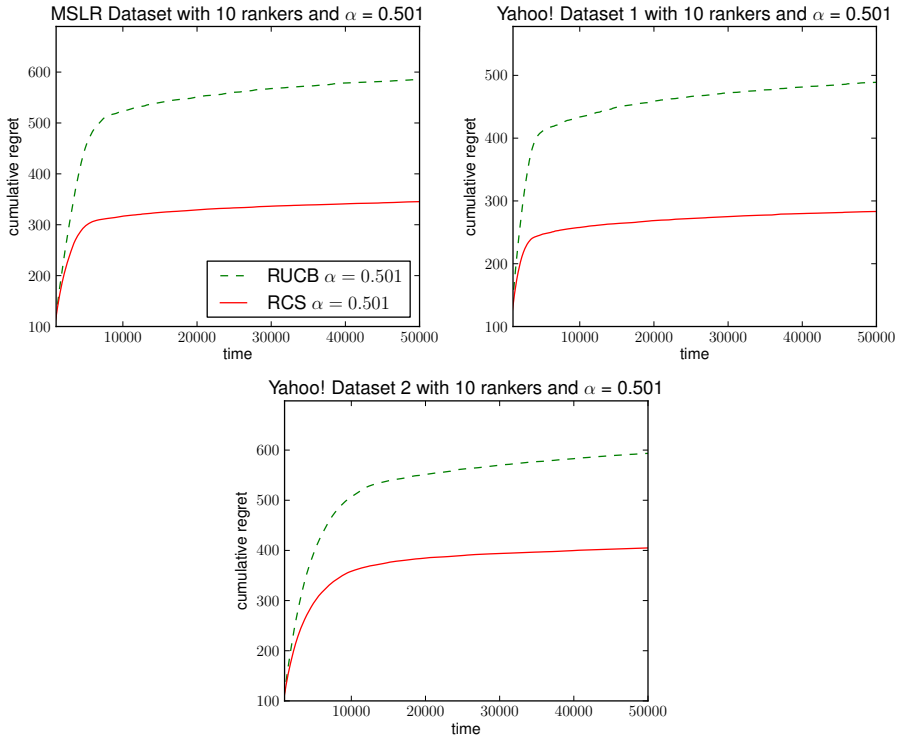

Figure 5.2: Cumulative regret averaged over 90 runs on the three datasets. All plots use axes with linear scale, since the two curves are much closer to each other than the ones in the regret plots in Figure 5.1.

the rest, thereby reducing regret, because one of the summands in the definition of regret (cf. (2.3)) is now zero and, more importantly, doing so leads to much better estimates of the probabilities p_{1j}, and thus greater confidence in the supremacy of the best arm. This in turn leads to more frequent comparisons between the best arm and itself, further reducing regret and increasing accuracy.

We also tested the accuracy of RUCB and found higher accuracy for RCS than RUCB, though the difference was relatively small. There was, however, a substantial difference in the regret performances of RUCB and RCS, as demonstrated in the next section. We omitted the accuracy curves for RUCB in order to improve the readability of the plots.

5.2.2 Cumulative Regret Results

Turning now to the other method for evaluating K-armed dueling bandits algorithms, Figure 5.2 shows the expected cumulative regret obtained when applying RUCB and RCS to three 10-armed dueling bandits problems obtained from the three different datasets. For both RCS and RUCB, the parameter α is set to be 0.501 so that RUCB's theoretical guarantees [78] hold. The curves in the plots show the mean cumulative regret over 90

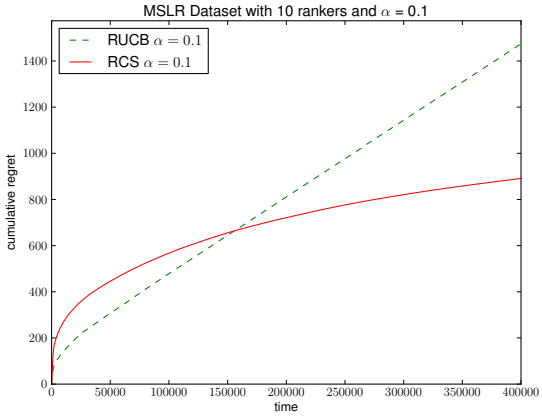

Figure 5.3: Cumulative regret averaged over 30 runs.

independent runs of each algorithm. The plots show results on the first 50,000 time steps, using linear scales on both axes.

Regarding **Q2**, the plots in Figure 5.2 clearly demonstrate that RCS accumulates substantially less regret than RUCB, with the former accumulating roughly a third less regret than latter. In other words, not only does RCS find the best arm more quickly than RUCB, it also makes less severe errors in the process of doing so. In more concrete terms, the difference in the regret levels at which the two algorithms plateau is on the order of 200 in these three datasets, which roughly translates to an extra 2000 interleaved comparisons involving suboptimal arms: this is because the probability with which the Condorcet winner beats the remaining arms is around 0.6. Needless to say, this performance difference can have a great impact on user satisfaction and engagement. Thus, these results highlight the benefits of a sampling-based approach to exploration.

Moreover, given the qualitative similarity between the performances of RUCB and RCS when $\alpha > 0.5$, we strongly suspect that similar regret bounds as those proven for RUCB [78] also hold for RCS. However, the use of sampling would necessitate a more intricate theoretical argument that we leave as future work.

Finally, in these experiments both algorithms had similar variance across runs; the best performing run of RUCB had a higher regret curve than the average regret curve of RCS.

5.2.3 Stability of RUCB and RCS

The improved performance of RCS over RUCB in the previous section can be attributed to the fact that RCS engages in less unnecessary exploration. Since lowering α makes both RCS and RUCB less exploratory, an important question, as posed in **Q3**, is whether regret can be even further reduced by setting α to values below 0.5.

To address this question, we investigate the stability of RUCB and RCS by setting $\alpha = 0.1$, which lies outside the range permitted by RUCB's theoretical results. Figure 5.3 shows the cumulative regret results averaged over 30 runs of both RUCB and RCS on

the MSLR dataset: fewer runs were used in this case to illustrate how easily RUCB can misbehave when α is below 0.5. These results show that the average cumulative regret for RUCB grows linearly, which was due to two of the runs never reaching the point where they keep interleaving the best arm with itself. By contrast, though RCS accumulates almost twice as much regret at $\alpha = 0.1$ than $\alpha = 0.501$, the performance degradation is much less severe than for RUCB, with RCS's cumulative regret curve flattening much more. Similar results were also observed with the other datasets.

The performance difference is due to the fact that reducing α results in shrinking the confidence intervals maintained by RUCB, which results in the tournament phase of the algorithm not being exploratory enough. Hence, RUCB focuses prematurely on a single arm that has a temporary advantage over the others, preventing it from getting better estimates of the comparison probabilities between the Condorcet winner and the rest, which is necessary for the Condorcet winner to be chosen by the tournament. This, however, is not a stumbling block for RCS because there are no confidence intervals in the tournament phase, which relies on sampling instead. Figure 5.3 demonstrates that these samples ensure enough exploration to avoid getting stuck with a suboptimal arm. Thus, while lower values of α are not beneficial to either algorithm, RCS remains stable while RUCB can experience the catastrophic negative performance associated with linear regret.

5.2.4 Size of the Set of Rankers

In order to study the issue of scalability, we compare RCS to RUCB on problems with 20, 30 and 40 arms, all extracted from the MSLR dataset. See Figure 5.4.

Regarding **Q4**, these results show that the cumulative regret curve of RCS flattens much sooner than that of RUCB. Thus, RCS starts focusing on the best arm more quickly than RUCB: more specifically, where the two curves cross, RUCB was spending on average 6 to 9 times more iterations interleaving non-optimal arms than RCS. For instance, in the experiments with 20 arms, in the vicinity of the crossing point, 3.6% of RCS's comparisons involved subptimal arms, whereas 27% of RUCB's did; the same quantities for the 30 and 40 arm experiments are 1.9% vs. 17% and 3.3% vs. 21%. On the other hand, at time $T = \frac{T_0}{2}$, where T_0 is the time at which the two average regret curves cross, these differences are much smaller with the same numbers being 31% vs. 33%, 31% vs. 36% and 44% vs. 45% for the 20, 30 and 40 arm experiments, respectively.

Note that this more rapid convergence to the best arm requires more aggressive exploration early on. This can be deduced from the fact that, before plateauing, the red curves for RCS have slightly steeper slopes than the green curves for RUCB during the same period. This is due to the fact that RCS abstains from removing any arms from consideration until it became clear which arm is the best, whereas RUCB stops comparing poorer arms earlier in the process. Nevertheless, RCS starts interleaving the best arm with itself in substantially fewer iterations and thus accumulates much less regret in the long run. For instance, at time $T = 2T_0$, where T_0 is the time at which the two average regret curves cross, the average cumulative regret of RUCB is 30%, 12.3% and 12.6% higher than that of RCS for 20, 30 and 40 arm experiments, respectively.

Moreover, the regret curves for RCS are much flatter after they plateau than those of RUCB, which means that, once the best arm is identified, RCS is more likely to avoid futile interleaved comparisons with suboptimal arms. More precisely, at time $T = 2T_0$,

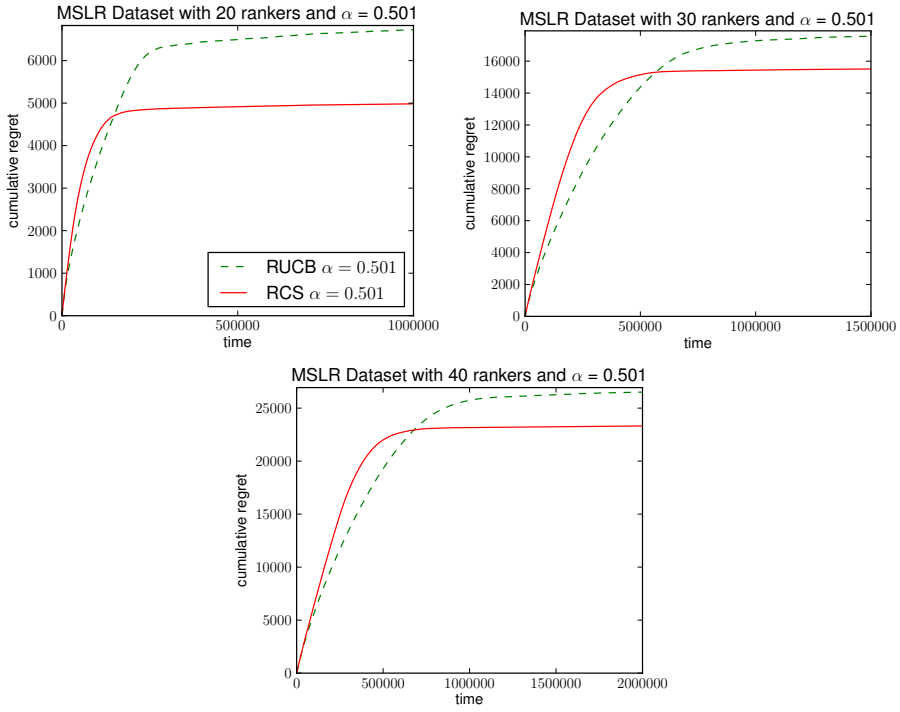

Figure 5.4: Cumulative regret averaged over 90 runs on the MSLR-WEB30K dataset, with 20, 30, 40 arms. For comparison, the plot for 10 arms is the leftmost plot in Figure 5.2; note that the scales differ between the four plots, which is necessary in order to illustrate the non-asymptotic portion of all of the results.

the percentage of comparisons RUCB devotes to suboptimal arms is still roughly 8 times higher than that of RCS, e.g., 2% and 0.3%, respectively, for the 20 arm experiment.

5.3 Summary

In this chapter, we have proposed a new method for addressing the K-armed dueling bandit problem and experimentally evaluated it using online ranker evaluation. Our method, *Relative Confidence Sampling* (RCS), was evaluated against the existing methods on large scale learning to rank datasets using two measures of performance: accuracy and cumulative regret. RCS significantly outperforms the existing state-of-the-art methods according to both measures. In particular, given the need in online ranker evaluation scenarios to identify and compare the best arm as quickly as possible, RCS has a large advantage over SAVAGE and RUCB, since when asked to return the best arm with accuracy in mind, it has a higher probability of returning the best arm, while minimizing the number of queries wasted on comparing suboptimal arms (as evidenced by the lower regret curves for RCS), without requiring prior knowledge of the length of the evaluation

or imposing restrictive assumptions such as a total ordering of the arms.

Given the results in §5.2.4, an interesting question that naturally arises is whether or not there exists an algorithm that scales well with the number of arms. This question is partially addressed in the next chapter, where we present a dueling bandit algorithm, called mergeRUCB, that asymptotically has linear dependence on the number of arms, rather than a quadratic one. Other attempts at dealing with this question are works such as [11] and [72], which deal with an infinite number of arms, with the additional assumption that the dueling bandit problem arises from underlying utilities that determine the probability with which an arm beats another. However, we pose as an interesting research question for further inquiry whether it is possible to adapt various extensions of the UCB algorithm to the case with infinitely many arms [13, 23, 53, 63, 66] to the dueling bandit setting in order to devise an algorithm that can solve continuous-armed dueling bandit problems, under no more restrictive an assumption than the existence of a Condorcet winner.

Finally, there remains the issue of our assumption that the dueling bandit problem contains a Condorcet winner, which might not hold in practice. We address this point in Chapter 7 with the introduction of the Copeland Confidence Bound algorithm that searches for a generalization of the Condorcet winner called the Copeland winner, which is guaranteed to exist.

6

MergeRUCB

In this chapter, we address the scalability issue brought up at the end of the last chapter. More specifically, a challenge that the algorithms considered so-far face is that the number of parameters that must be learned grows quadratically with K, i.e., the number of arms. This in turn results in excessive exploration. The challenge is especially relevant in the case of web search, where a large number of arms may be under consideration. This is of practical significance because as reported for instance in [48] on any given day over 200 concurrent experiments are being run at Bing, with users ending up in one of billions of possible variants of the site. Therefore, algorithms, such as SAVAGE, RUCB and RCS, would have difficulty scaling to such large values of K. Other algorithms such as IF and BTM avoid this problem by making more restrictive assumptions, such as a total ordering of the arms, that make it possible to identify the best arm without explicitly considering all pairs. However, this approach is problematic because the required assumptions often do not hold in applications such as ranker evaluation.

We remedy the above shortcomings by bridging the gap between these two approaches. Specifically, we propose and evaluate a new method for evaluating rankers, called **mergeRUCB**, that makes only weak assumptions about the K-armed dueling bandit problem, but provably requires only $\mathcal{O}(K)$ comparisons and therefore performs well when many rankers must be compared, as is typically the case in web search. As the name suggests, mergeRUCB uses a divide and conquer strategy to reduce the number of exploratory comparisons carried out by the evaluation process. It proceeds by grouping rankers into small batches so that fewer comparisons are needed before rankers can be eliminated.

The remainder of this chapter is organized as follows: in §6.1, we describe the algorithm and provide intuition for the specific choices made in its design; in §6.2, we state our main theoretical result and the necessary lemmas; in §6.3, we provide formal proofs for the claims made in the previous section; in §6.4, we conduct an experimental evaluation of mergeRUCB, comparing it against other dueling bandit algorithms using large-scale examples from ranker evaluation; and finally §6.5 gives an overview of the findings in this chapter.

6.1 The Algorithm

In this section, we present mergeRUCB, shown in Algorithm 4, to deal with K-armed dueling bandit problems involving many arms. As with sorting algorithms, most naive approaches to the K-armed dueling bandit problem suffer from quadratic dependence on K because they require every arm to be compared against every other arm. However, this quadratic dependence can be avoided by a mergesort-style algorithm that carries out comparisons only "locally," i.e., items are placed in small batches that are processed separately and then merged together.

The same principle underlies mergeRUCB. The crucial difference is that, unlike in sorting, one comparison is not sufficient to determine which of a pair of arms is better, since feedback is stochastic. Furthermore, the number of times two arms must be compared is larger if the arms are more similar. In the worst case, we have $p_{ij} = 0.5$ and the two arms cannot be distinguished. This case is problematic because ρ_i and ρ_j might be weak arms overall (i.e., lose badly to other arms), in which case comparing them to each other many times will incur large regret. MergeRUCB deals with this difficulty by using the best arm in the batch to eliminate the rest. If a batch contains only similar arms and is thus too slow in eliminating arms, it is combined with other batches that have more variety.

In the following, we explain the components of mergeRUCB, which proceeds in *stages* (Line 4). Before the first stage, the algorithm groups arms into small batches $\mathcal{B}_i$ (Line 2). Then, within each stage, mergeRUCB carries out interleaved comparisons among arms that reside in the same batch. At any given time, the choice of arms to compare against each other inside a given batch is guided by a matrix $\mathbf{U}$ of upper confidence bounds (Line 7), which is obtained by optimistically estimating the preference probabilities p_{ij}: the optimism is included to ensure sufficient exploration among the arms. The matrix $\mathbf{U}$ is used both to eliminate arms if they lose to other arms by a wide margin (Line 8) and to choose the arm ρ_d (Line 10) that is selected so as to hasten the elimination of ρ_c, which is chosen randomly. The algorithm proceeds in this fashion until the number of remaining arms becomes small (Line 12), at which point the stage is concluded by merging pairs of batches together to form bigger batches (Line 13). This initiates the next stage, and the process repeats until a single arm remains. Our theoretical results state that the probability that this remaining arm is the Condorcet winner is greater than $1 - \delta$.

6.2 Theory

In this section, we provide theoretical guarantees for the proper functioning and scalability of mergeRUCB. We begin by listing a number of reasonable assumptions that we impose upon the problem in order to guarantee the proper functioning of the algorithm:

A1. We assume that there is no repetition of rankers, i.e., any pair of rankers ρ_i and ρ_j are different and thus $p_{ij} \neq 0.5$, unless both rankers are *uninformative*: they provide no useful information and so lose to all other rankers, i.e., $p_{ki} \geq 0.5$ and $p_{kj} \geq 0.5$ for all k.

A2. We assume that at most a third of the rankers are uninformative.

Algorithm 4 mergeRUCB(δ)

Require: A set of arms $\rho_1, \ldots, \rho_K$;
an oracle that can take a pair of arms and return one as the winner (e.g., an interleaved comparison method);
the size of each partition, $p \geq 4$;
the maximum probability of failure, δ;
$\alpha > \frac{1}{2}$.

$\mathbf{W} \leftarrow \mathbf{0}_{K \times K}$ {2D array of wins: $\mathbf{W}_{ij}$ is the number of times ρ_i has beaten ρ_j}

$\mathcal{B}_1 = \left\{ \underbrace{\{\rho_1, \ldots, \rho_p\}}_{B_1}, \ldots, \underbrace{\{\rho_{(b_1-1)p+1}, \ldots, \rho_K\}}_{B_{b_1}} \right\}$, a set of disjoint batches of arms, with $b_1 = \lfloor \frac{K}{p} \rfloor$

$C(\delta) = \left\lceil \left(\frac{(4\alpha-1)K^2}{(2\alpha-1)\delta} \right)^{\frac{1}{2\alpha-1}} \right\rceil$

$S = 1$ {The stage that the algorithm is in.}

for $t = 1, 2, \ldots$ **do**

  $i = t \mod b_S$

  $\mathbf{U} = \frac{\mathbf{W}}{\mathbf{W}+\mathbf{W}^T} + \sqrt{\frac{\alpha \ln(t+C(\delta))}{\mathbf{W}+\mathbf{W}^T}}$, where all operations are element-wise.

  For each $\rho_k \in B_i$ if $\mathbf{U}_{kl} < \frac{1}{2}$ for any $\rho_l \in B_i$, remove ρ_k from B_i.

  Select $\rho_c \in B_i$ randomly.

  Set $d := \arg\max_{\{l | \rho_l \in B_i \setminus \{\rho_c\}\}} \mathbf{U}_{lc}$.

  Compare ρ_c against ρ_d and increment $\mathbf{W}_{cd}$ if c won and $\mathbf{W}_{dc}$ otherwise.

  if $\sum_i |B_i| \leq \frac{K}{2^S}$ **then**

    Combine pairs of batches of arms so that each new batch has between $p/2$ and $3p/2$ arms in it, pairing the smallest batches with the largest ones, making sure that each batch contains at least two arms. Update the sets B_i, putting them all in the set $\mathcal{B}_S$, and define $b_S := |\mathcal{B}_S|$.

    $S = S + 1$

In online ranker evaluation in web search settings, assumption **A1** is reasonable because the rankers ρ_i under evaluation are typically the result of substantial deliberation and research and so the chances of the same informative ranker appearing twice are slim. The second assumption is motivated by the Yahoo! Learning to Rank challenge dataset, in which either 104 or 181 (depending on the dataset) out of 700 feature rankers always return zero. More generally, it is plausible that some uninformative rankers are inadvertently included in the evaluation process. However, if there are too many of them, the evaluation task will be lengthened.

Here, we provide a high probability bound on the regret accumulated by mergeRUCB; Table 6.1 lists our notation.

Theorem 6.1. *Given a K-armed dueling bandit problem with rankers $\rho_1, \ldots, \rho_K$ with ρ_1 the Condorcet winner, then if we apply mergeRUCB(δ), with probability $1 - \delta$ we have*

Table 6.1: List of notation used in Section 6.2.

Symbol	Definition
K	Number of rankers
α	Exploration parameter in Algorithm 4
δ	Probability of failure
p	Initial size of the batches
S	Stage of the algorithm
$\mathcal{B}_S$	Set of batches in stage S
b_S	Number of batches in stage S
R_T	Cumulative regret at time T
$w_{ij}(t)$	Number of times ρ_i beat ρ_j in the first t time-steps
$N_{ij}(t)$	$w_{ij}(t) + w_{ji}(t)$
$u_{ij}(t)$	$\mathbf{U}_{ij} := \frac{w_{ij}(t)}{N_{ij}(t)} + \sqrt{\frac{\alpha \ln t}{N_{ij}(t)}}$
$l_{ij}(t)$	$1 - u_{ji}(t)$
$C(\delta)$	$\left(\frac{(4\alpha-1)K^2}{(2\alpha-1)\delta}\right)^{\frac{1}{2\alpha-1}}$
Δ_{ij}	$p_{ij} - 0.5$
$\Delta_{B,\min}$	$\min_{i,j \in B} \Delta_{ij}$
T_B	$\frac{4\alpha\binom{q-1}{2}\log(T + C(\delta))}{\Delta^2_{B,\min}}$
T_i	T_{B_i}
$\widehat{\Delta}_S$	$\left(\frac{2b_S}{3} + 1\right)^{th}$ largest element of $\{\Delta_{B,\min} \mid B \in \mathcal{B}_S\}$
$\widehat{T}_S$	$\frac{8\alpha pK \ln(T + C(\delta))}{\widehat{\Delta}^2_S}$
$\ln t$	Natural logarithm of t

the following bound on cumulative regret at time T:

$$R_T \leq \frac{16\alpha pK \ln(T + C(\delta)) \max_j \Delta_{1j}}{\min_{S=1,\ldots,\lceil \log_2 K \rceil} \widehat{\Delta}^2_S} \leq \frac{8\alpha pK \ln(T + C(\delta))}{\min_{\{(i,j) \mid p_{ij} \neq 0.5\}} \Delta^2_{ij}}.$$

This theorem says that if mergeRUCB is run for T time-steps with probability of failure set to δ, then with probability $1 - \delta$, the total regret accumulated by the algorithm is bounded by an expression that is logarithmic in T and linear in K. This in turn tells us that the number of suboptimal interleaved comparisons grows linearly in K, since accumulating non-zero regret corresponds to suboptimal comparisons. Unlike existing results in the literature, the strongest of which take the form $\mathcal{O}(K^2) + \mathcal{O}(K \log T)$, Theorem 6.1 is the first regret bound that is completely linear in K.

Furthermore, even though as stated the above theorem is a high probability bound, by setting $\delta = 1/T$, we obtain a bound on the expected regret of mergeRUCB at time T as follows: since the maximum amount of regret that the algorithm can accumulate in the

first T time-steps is bounded by T, we have

$$\begin{aligned}\mathbf{E}R_T &\leq \delta T + (1-\delta)\frac{8\alpha pK\ln(T+C(\delta))}{\Delta^2_{\min}}\\ &\leq 1 + \frac{8\alpha pK\ln T}{\Delta^2_{\min}} + \frac{8\alpha pKC(1/T)}{T\Delta^2_{\min}},\\ &\leq 1 + \frac{8\alpha pK\ln T}{\Delta^2_{\min}} + \frac{8\alpha pK\left(\frac{T(4\alpha-1)K^2}{(2\alpha-1)}\right)^{\frac{1}{2\alpha-1}}}{T\Delta^2_{\min}},\end{aligned}$$

where $\Delta_{\min} = \min_{\{(i,j)\,|\,p_{ij}\neq 0.5\}}\Delta^2_{ij}$ and the second inequality is obtained by using a Taylor expansion of $\ln t$ at $t = T$. Now, if $\alpha \geq 1$, the last summand in the right-hand side of the above inequality is in $\mathcal{O}(1)$, and so we have a finite-horizon expected regret bound of the form $\mathcal{O}(K\ln T)$. Moreover, this finite horizon bound can be turned into an infinite horizon one (up to $\ln\ln T$ factors) using the 'squaring trick' [4]. We would like to emphasize that these results hold under very general assumptions that do not preclude the existence of cyclical relationships among the rankers.

The proof of Theorem 6.1 relies on the following lemma.

Lemma 6.2. *In mergeRUCB(δ), consider a batch B of size q, at least one of whose rankers is informative. Let $\Delta_{B,\min}$ denote the smallest nonzero gap $\Delta_{kl} := |p_{kl} - \frac{1}{2}| \neq 0$, with $\rho_k, \rho_l \in B$. Then, the number of comparisons N_B that could have happened between pairs of rankers in B before it is merged with another batch is bounded with probability $1-\delta$ as follows:*

$$N_B < T_B := \frac{4\alpha\binom{q-1}{2}\ln(T+C(\delta))}{\Delta^2_{B,\min}}.$$

The proof of this lemma follows directly from the fact that the number of comparisons between any pair of rankers in the batch is at most $\frac{4\alpha\ln(T+C(\delta))}{\Delta^2_{B,\min}}$, as proven in Lemma 6.3 below, since there are $\binom{q-1}{2}$ distinct pairs of rankers in B.

Lemma 6.3. *Given any pair of distinct rankers $\rho_i, \rho_j \in B$, the maximum number of comparisons that could have been carried out between these two rankers in the first T time-steps of Algorithm 4 before a merger between B and another batch occurs, is bounded by $\frac{4\alpha\ln(T+C(\delta))}{\Delta^2_{B,\min}}$.*

The proof of Lemma 6.3 considers the two possible cases: either at least one of the rankers under consideration is informative or both are uninformative. In the first case, if the two rankers have been compared more times than the above number, we show that one of the two must have eliminated the other, while in the second case, if the two rankers have been compared too many times, then a third, informative ranker (whose existence is guaranteed by the assumption of the lemma) must have eliminated one of them.

The proof of the above lemma relies on Lemma 4.1, which we repeat here for the reader's convenience:

Lemma 6.4. *Let* $\mathbf{P} := [p_{ij}]$ *be the preference matrix of a* K*-armed dueling bandit problem with arms* $\{a_1, \ldots, a_K\}$*. Then, for any dueling bandit algorithm and any* $\alpha > \frac{1}{2}$ *and* $\delta > 0$*, we have*

$$P\Big(\forall\, t > C(\delta), i, j,\ p_{ij} \in [l_{ij}(t), u_{ij}(t)]\Big) > 1 - \delta.$$

Given the above facts, we can describe the main idea of the proof of Theorem 6.1 as follows. The central difficulty in the proof is that there may exist batches that consist entirely of uninformative rankers. This is problematic because in these batches no rankers are eliminated, since for each pair of rankers ρ_i, ρ_j in such a batch, we have $p_{ij} = 0.5$ and so with high probability we have neither $u_{ij} < 0.5$ nor $u_{ji} < 0.5$. The proof overcomes this difficulty by showing that such fully uninformative batches all disappear at the end of the first stage of the algorithm. This occurs because of how the batches are merged at the end of each stage (cf. Line 13 of Algorithm 4): the largest batches are combined with the smallest ones. Since uninformative batches inevitably fail to eliminate any rankers, they have the largest number of rankers, while the smallest batches are guaranteed to contain informative rankers. Therefore, from the second stage onwards, mergeRUCB is guaranteed not to compare rankers in a batch, none of whose elements will be eliminated.

6.3 Proofs

In this section, we prove the results stated in the last section.

Proof of Lemma 6.3. Let us begin by assuming that the number of comparisons between ρ_i and ρ_j is greater than $\frac{4\alpha \ln(T+C(\delta))}{\Delta^2_{B,\min}}$, and let us distinguish between two cases:

1. *At least one of* ρ_i *and* ρ_j *is informative*: in this case, by assumption **A1**, we know that $p_{ij} \neq 0.5$, and moreover by Lemma 6.4, we know that with probability $1-\delta$ we have $p_{ij} \in [l_{ij}(t), u_{ij}(t)]$, with t being the last time that ρ_i was compared against ρ_j and $l_{ij} := 1 - u_{ji}$. However, this tells us that one of the two rankers should have been eliminated already, since we have

$$\begin{aligned} u_{ij}(t) - l_{ij}(t) &= 2\sqrt{\frac{\alpha \ln(t + C(\delta))}{N_{ij}(t)}} \\ &\leq 2\sqrt{\frac{\alpha \ln(T + C(\delta))}{N_{ij}(t)}} \\ &< 2\sqrt{\frac{\alpha \ln(t + C(\delta))}{\frac{4\alpha \ln(T + C(\delta))}{\Delta^2_{B,\min}}}} = \Delta_{B,\min} \leq \Delta_{ij}, \end{aligned} \tag{6.1}$$

   where the last inequality is due to our assumption that $N_{ij}(t) > \frac{4\alpha \ln(T+C(\delta))}{\Delta^2_{B,\min}}$. Therefore, the confidence interval $[l_{ij}(t), u_{ij}(t)]$ does not contain 0.5, which is the criterion used by Algorithm 4 to eliminate rankers.

2. *Rankers ρ_i and ρ_j are both uninformative*: by assumption **A1** and Lemma 6.4, uninformative rankers cannot eliminate informative rankers, so no matter how many rankers have been eliminated from B, there must be an uneliminated third ranker ρ_k that is informative in the batch together with ρ_i and ρ_j, and by assumption **A1**, we have $p_{ki} > 0.5$ and $p_{kj} > 0.5$. Again, applying Lemma 6.4 as in the previous case, we know that with probability $1 - \delta$ we have

$$0.5 = p_{ij} \in [l_{ij}(t), u_{ij}(t)];$$

on the other hand, using the same chain of inequalities as in (6.1), we can deduce that

$$u_{ij}(t) - l_{ij}(t) < \Delta_{B,\min} \leq \min\{\Delta_{ki}, \Delta_{kj}\}. \tag{6.2}$$

Now, in order for ρ_i to have been compared to ρ_j at time t, we must have had one of the following two scenarios:

   (a) mergeRUCB chose $c = i$ and $d = j$ at time t: this requires the satisfaction of the following two conditions:

   - $u_{ij}(t) \geq 0.5$, by Line 8 of Algorithm 4.
   - $l_{ij}(t) \leq p_{ik}$: this is because in order to have $d = j$, we must have $u_{ji}(t) \geq u_{ki}(t)$ and by Lemma 6.4, we have $u_{ki}(t) \geq p_{ki}$, and so $l_{ij}(t) := 1 - u_{ji}(t) \leq 1 - p_{ki} = p_{ik}$.

   This means that we have $u_{ij}(t) - l_{ij}(t) \geq \Delta_{ki}$. However, this contradicts inequality (6.2), so we could not have had $(c, d) = (i, j)$.

   (b) mergeRUCB chose $c = j$ and $d = i$ at time t: repeating the same argument as in the previous case with i and j swapped, we get $u_{ji}(t) - l_{ji}(t) \geq \Delta_{kj}$, which also contradicts inequality (6.2).

Therefore, our assumption that the number of comparisons between ρ_i and ρ_j is greater than $\frac{4\alpha \ln(T+C(\delta))}{\Delta^2_{B,\min}}$ cannot hold in either scenario. □

Proof of Theorem 6.1. We begin by considering the first stage of the algorithm:

$S = 1$ During the first stage, we have two types of batches: those that consist solely of uninformative rankers and those that contain at least one informative ranker. Assumption **A2** implies that at least two thirds of the batches have at least one informative ranker, so we can apply Lemma 6.2 to them.

To estimate number of time-steps mergeRUCB spends in its first stage, we introduce the following notation: recall from Algorithm 4 that b_1 is the number of partitions in the first stage of the algorithm and let $\widehat{\Delta}_1$ denote the $\left(\frac{2b_1}{3} + 1\right)^{th}$ largest number in the set $\{\Delta_{B,\min} | B \in \mathcal{B}_1\}$. Now, once all but one of the rankers in every batch B with $\Delta_{B,\min} \geq \widehat{\Delta}_1$ have been eliminated, the algorithm moves to the next stage. This occurs because at least half of the rankers have been eliminated, since there are $\frac{2b_1}{3} + 1$ batches, inside which $p - 1$ rankers are eliminated, and so the total number

of eliminated rankers is at least

$$\left(\frac{2b_1}{3}+1\right)(p-1) \geq \frac{2(b_1+1)}{3}(p-1)$$
$$\geq \frac{2K}{3p}(p-1)$$
$$\geq \frac{2K}{3}\frac{3}{4} \geq \frac{K}{2}. \qquad \text{(since } p \geq 4\text{)}$$

Therefore, Line 12 of Algorithm 4 forces the next stage to begin. Now, applying Lemma 6.2 to the $\frac{2b_1}{3}$ batches B with $\Delta_{B,\min} \geq \widehat{\Delta}_1$, and using the fact that the size of the batches is at most $2p$ (cf. Line 2 of Algorithm 4), we can conclude that with probability $1-\delta$ the number of time-steps in the first stage of mergeRUCB could not have been more than

$$\frac{K}{p} \times \frac{4\alpha\binom{2p}{2}\ln(T+C(\delta))}{\widehat{\Delta}_1^2} \leq \frac{8\alpha pK\ln(T+C(\delta))}{\widehat{\Delta}_1^2} =: \widehat{T}_1.$$

$S \geq 2$ At the end of the first stage of the algorithm, we combine the largest remaining batches with the smallest ones. The fact that exactly half of the rankers were eliminated in the first stage implies that this policy for combining batches forces every fully uninformative batch to acquire an informative ranker, since by Assumption **A1** and Lemma 6.4, the probability of an uninformative ranker eliminating an informative ranker is less than δ. Hence, from this point on, we can apply Lemma 6.2 to every batch.

We can use a similar argument as with the first stage of the algorithm to bound the number of time-steps that mergeRUCB would spend in the S^{th} stage. To that end, let $\widehat{\Delta}_S$ denote the $\left(\frac{2b_S}{3}+1\right)^{th}$ largest number in the set $\{\Delta_{B,\min}|B \in \mathcal{B}_S\}$. Now, applying the same argument as above and using the fact that in stage S we have $K/2^{S-1}$ rankers, we get that the number of comparisons in stage S of mergeRUCB is bounded by

$$\frac{8\alpha pK\ln(T+C(\delta))}{2^{S-1}\widehat{\Delta}_S^2} =: \widehat{T}_S.$$

After $\lceil \log_2 K \rceil$ stages, only a single ranker remains, beyond which point mergeRUCB goes on interleaving that ranker with itself. This ranker is the Condorcet winner with probability $1-\delta$ because the probability of the Condorcet winner being eliminated by another ranker is at most δ. Therefore, in order to estimate the total regret accumulated by mergeRUCB we can sum the $\widehat{T}_S$ for $S = 1, \ldots, \lceil \log_2 K \rceil$ and multiply the result by the maximum regret any comparison can result in, which is $\max_j \Delta_{1j}$. This gives the bound in the statement of Theorem 6.1 once we notice that

$$\sum_{S=1}^{\lceil \log_2 K \rceil} \frac{1}{2^{S-1}\widehat{\Delta}_S} \leq \frac{2}{\min\limits_{S=1,\ldots,\lceil \log_2 K \rceil} \widehat{\Delta}_S^2}$$

This concludes the proof of Theorem 6.1. □

6.4 Experiments

We use all three learning to rank datasets discussed in Chapter 3 to test the scalability of mergeRUCB in comparison to the other algorithms discussed so far. More specifically, our experiments aim to answer the following specific questions:

Q1 Does mergeRUCB outperform BTM, the state-of-the-art online ranker evaluation algorithm for large-scale evaluation problems?

Q2 How does mergeRUCB scale as the number of arms increases in comparison to existing algorithms?

Q3 How does the click model affect the scalability of the various algorithms?

Q4 How does the performance of mergeRUCB depend on the parameters α and p? In particular, how do our default parameters perform?

For the large-scale experiments in §6.4.1, aimed at answering **Q1**, we use all of the feature arms available in these datasets and perform the comparisons between arms by directly using Lerot to simulate interleaved comparisons. In this case, our assumption that there exists a Condorcet winner happens to be satisfied in the case of all four datasets. For all other experiments, for each value of K tested, we choose 10 subsets of arms of size K and apply each algorithm to each subset: this choice is made by sampling subsets of size K at random and keeping the first 10 that have Condorcet winners. As illustrated in [77], the probability that a subset has a Condorcet winner depends on K, but is generally very high. In addition, since the Lerot-based experiments for **Q1** took three months to complete,[1] we use a faster proxy setup for the other experiments: for each pair of feature arms ρ_i, ρ_j in the MSLR dataset, we estimate the probability p_{ij} that ρ_i beats ρ_j by simulating 400,000 interleaved comparisons between the two using Lerot.[2] Given these numbers p_{ij}, in the remaining experiments, we perform comparisons between arms ρ_i and ρ_j for each pair (i, j) by drawing a sample from the Bernoulli distribution with mean p_{ij}, i.e., by flipping a biased coin. This is a standard approach to evaluating dueling bandit algorithms (cf. [73, 75, 78]). We verify the accuracy of the proxy approach in §6.4.2.

In all experiments other than those in §6.4.5, we use the following parameter settings: $\alpha = 1.01$ and $p = 4$. In fact, the only constraint on α is that it should be greater than 0.5 in order for $C(\delta)$ (cf. Line 3 in Algorithm 4) to be well-defined and for our theoretical results in §6.2 to hold. However, as α approaches 0.5, the expression for $C(\delta)$ grows super-exponentially as a function of α, and so the benefits of having more slowly growing confidence intervals (cf. Line 7 of Algorithm 4) are outweighed by the added exploration caused by starting with larger confidence intervals. Indeed, as demonstrated in §6.4.5, there is little or no gain from changing these parameters from the above values. Moreover, for all of our experiments, we chose the probability of failure, δ to be 0.01. Finally, all experiments other than those in §6.4.4 used the navigational click model (cf. Table II of [37]) to simulate user click behavior.

[1] This was primarily due to shortcomings of the competing algorithms, which need to be run sequentially. By contrast, mergeRUCB can easily be parallelized across different batches.

[2] The resulting matrices can be found here (as Numpy matrices): `bit.ly/nips15data`

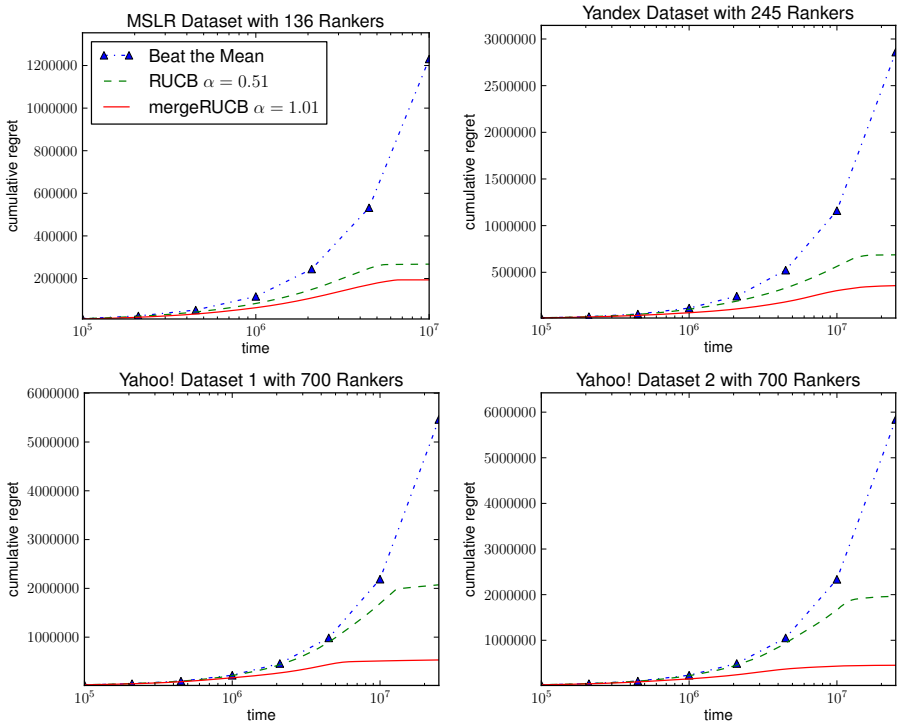

Figure 6.1: Average cumulative regret plots for four large-scale evaluation problems.

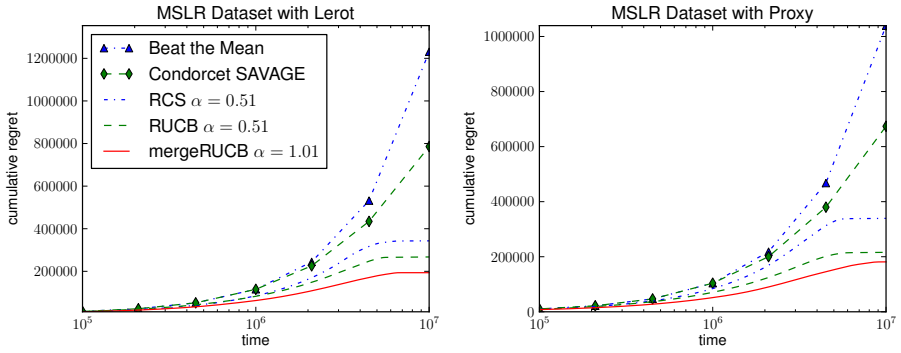

Figure 6.2: Average cumulative regret on the 136-ranker evaluation problem arising from the MSLR dataset using Lerot (left) or the proxy approach (right).

In the following subsections, we present our experimental results to answer the questions raise above.

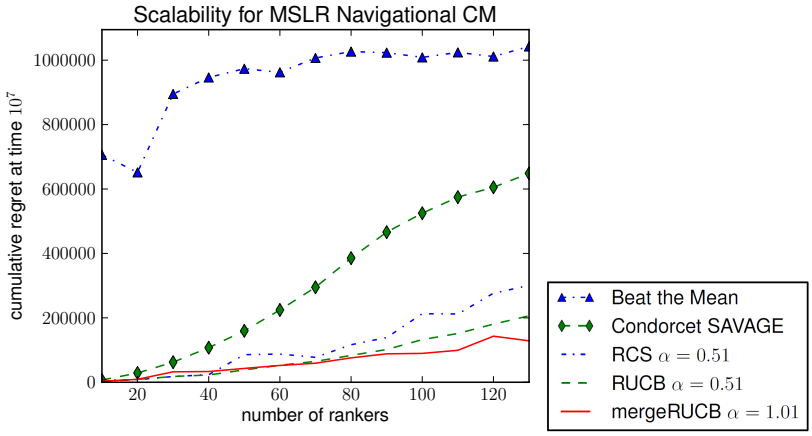

Figure 6.3: Average cumulative regret after 10^7 iterations on K-ranker evaluation problems with K ranging from 10 to 130.

6.4.1 Large scale experiments

We first address our main research question, **Q1**. We tested mergeRUCB on the full set of feature vectors of the four large learning to rank datasets described in Table 3.1, directly using Lerot instead of the proxy approach. The MSLR results, shown in Figure 6.1 (top-left), were carried out for 10 million time-steps, since two of the three algorithms converge to the Condorcet winner within that time frame. For the remaining datasets, we extended the horizon to 25 million time-steps, again to make sure two of the three algorithms converge. These results are shown in the remaining plots in Figure 6.1. Note that, in these plots and those that follow, the time axis uses a log scale, while the vertical axis uses a linear scale.

For these experiments, we tested three algorithms: mergeRUCB and RUCB, which had the best performance in the scalability experiments in §6.4.3, together with BTM, which is the state of the art K-armed dueling bandit algorithm for large K, according to [65, 73]. These plots show that, as the number of arms increases (going from 136 to 245 to 700), so does the difference between the performance of mergeRUCB and the remaining algorithms.

6.4.2 Lerot simulation vs Bernoulli samples

The remaining results presented in this work use the proxy approach described in §6.4. So, before proceeding further, we validate the proxy approach by showing that it provides qualitatively similar results to those generated with Lerot. To do so, we compare the performances of five K-armed dueling bandit algorithms on the MSLR dataset using both approaches. The results for Lerot are shown in Figure 6.2 (left), while those of the proxy approach are shown in Figure 6.2 (right). Comparing the two plots shows that there is no qualitative difference in the relative performance of the various dueling bandit algorithms under consideration here. Consequently, we use the proxy method to conduct

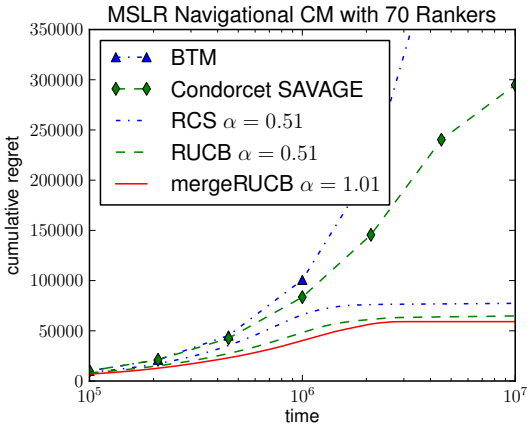

Figure 6.4: Average cumulative regret on the 70-ranker evaluation problem arising from MSLR.

the experiments described in the rest of this section.

6.4.3 Dependence on K

To address **Q2**, we compare 5 dueling-bandit algorithms on K-ranker evaluation experiments with K ranging from 10 to 130 in increments of 10 with the K arms chosen randomly from the 136 feature arms in the MSLR dataset.

Figure 6.3 shows the results: the horizontal axis measures K, the number of arms, while the vertical axis shows the regret accumulated after 10^7 iterations. As this plot demonstrates, for $K \geq 70$, mergeRUCB outperforms all other dueling bandit algorithms.

Of course, while Figure 6.3 shows performance across different values of K, it does so for only one moment in time: after 10^7 iterations. However, comparing Figure 6.3 to Figure 6.2 confirms that, for $K = 136$, the regret accumulated by the algorithm after 10^7 time-steps is a good indication of the overall performance of the algorithm over time. Figure 6.4 confirms that the same is true when $K = 70$.

6.4.4 Effect of click models

To address **Q3**, we conducted the same scalability test as in §6.4.3, using three different click models proposed in [37], namely the *perfect*, *navigational* and *informational* click models. The perfect click model represents the behavior of a persistent user, who inspects every single document in the retrieved list and clicks on each document with a probability proportional to the document's relevance to the given query. The navigational click model simulates the behavior of a user who is trying to satisfy a specific information need and is likely to stop inspecting the items in the list upon viewing a relevant document. Finally, the informational click model mimics the behavior of a user whose information need is not satisfied by a single document and is trying to gather information about a general

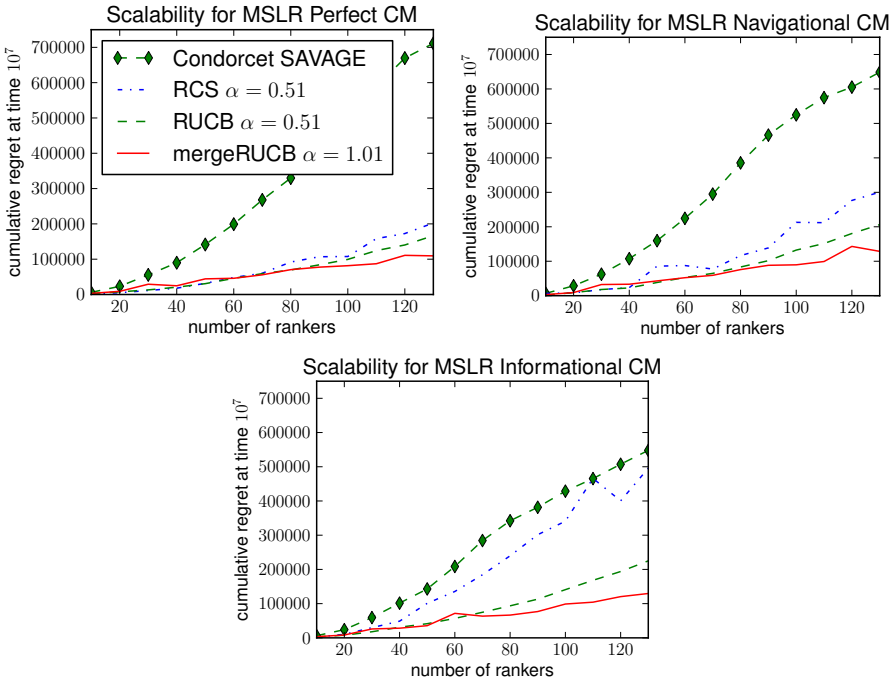

Figure 6.5: Average cumulative regret after 10^7 iterations on K-ranker evaluation problems with K ranging from 10 to 130 for the perfect (top-left), navigational (top-right), and informational (bottom) click models.

topic. Accordingly, the informational click model is more likely to continue inspecting the items retrieved by the arm even after encountering a relevant document.

The results, shown in Figure 6.5, demonstrate that RCS is affected more severely by the click model than either mergeRUCB or RUCB. This is because, in our experience, RCS tends to be sensitive to the margins by which the Condorcet winner beats the remaining arms: as these gaps shrink, the performance of RCS degrades dramatically. This is precisely what takes place when one replaces the perfect click model with the navigational one and the latter with the informational click model, since doing so increases the number of clicks in interleaved comparisons, making them noisier.

6.4.5 Parameter dependence

To address **Q4**, we repeated the experiments in §6.4.3, using the following grid of parameters:

$$(p, \alpha) \in \{4, 6, 8, 10\} \times \{0.71, 0.81, 0.91, 1.01, 1.11, 1.21\}.$$

Figure 6.6 shows, for each number of arms, the minimum and maximum cumulative regrets accumulated by mergeRUCB across the above set of parameters, as well as the regret for the default parameters used in the other experiments. Note that the vertical

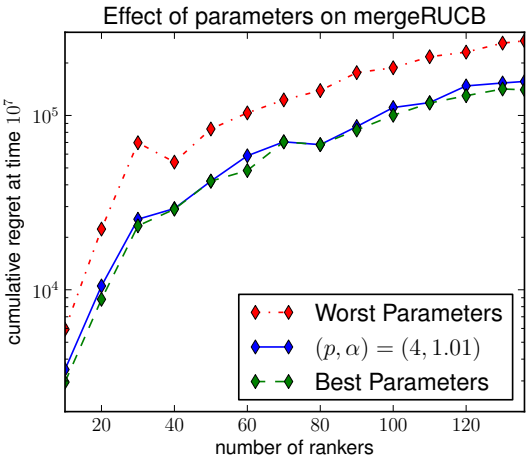

Figure 6.6: Effect of parameters on mergeRUCB's with MSLR dataset and navigational click model.

axis uses a log scale, which is chosen to facilitate comparing the three curves for small values of K, since in a linear plot they would be too close to distinguish from each other. As can be seen from the plots, the regret accumulated by mergeRUCB, using the default parameters $(p, \alpha) = (4, 1.01)$, is consistently close, if not equal, to the regret accumulated by the best choice of parameters, which validates our intuition that the default parameters are sensible.

6.5 Summary

In this chapter, we have proposed a new algorithm, called mergeRUCB, for the K-armed dueling bandit problem in situations that are of particular interest for web search, i.e., with large numbers of arms. We conducted extensive experimentation to understand the behavior of this algorithm in comparison to other online evaluation algorithms. The results of these experiments demonstrate that mergeRUCB can significantly outperform existing state of the art algorithms on large-scale evaluation problems. Moreover, we provided theoretical guarantees proving the proper functioning of mergeRUCB.

The algorithm presented in this chapter makes it feasible for search engines to perform large-scale online ranker evaluation experiments that might be too costly if other K-armed dueling bandit algorithms were used. Furthermore, our theoretical results provide the necessary assurance for undertaking such large-scale evaluation tasks. Since we care more about the worst-case performance of ranker evaluation algorithms than their average performance, our theoretical results bound the regret of the algorithm with high probability rather than proving bounds on expected regret, unlike previous work such as [75, 78].

7 Copeland Confidence Bounds

In this chapter, we present the final dueling algorithm discussed in this thesis. The algorithm, named **Copeland Confidence Bounds** (CCB), is designed to deal with dueling bandit problems that lack a Condorcet winner, which is required by all of the algorithms discussed so far. In such a scenario, there is an over-abundance of proposals for definitions of what constitutes a winner coming from social choice theory and game theory [58, 61], with each definition having its merits and shortcomings, and CCB seeks to find one such definition, called the Copeland winner, as described in Section 2.1.2.

The remainder of this chapter is organized as follows: in §7.1, we present experimental evidence for the need for an algorithm that generalizes beyond the Condorcet case; in §7.2, we present the pseudo-code for CCB and motivate some of the choices made in the design of the algorithm; in §7.3, we present our main result, bounding the regret accumulated by CCB; in §7.4, we provide proofs of the results stated in the previous section; in §7.5, we evaluate CCB experimentally, and finally §7.6 offers an overview of the findings in this chapter.

7.1 Motivation

In this section, we begin by motivating the need for an algorithm such as CCB that can deal with dueling bandit problems without Condorcet winners (cf. §2.1.2). Moreover, we offer a comparison between the Copeland winner, the notion used by CCB against other definitions of what constitutes a winner in the absence of the Condorcet winner. Finally, we end this section by investigating some quantities that will arise in our regret bounds in §7.3.

7.1.1 The Condorcet Assumption

To test how stringent the Condorcet assumption is, we use the informational preference matrix described in §7.5 to estimate for each $K = 1, \dots, 136$ the probability P_K that a given K-armed dueling bandit problem, obtained from considering K of our 136 feature rankers, would have a Condorcet winner by randomly selecting $10,000$ K-armed dueling bandit problems and counting the ones with Condorcet winners. As can be seen from Figure 7.1, as K grows, the probability that the Condorcet assumption holds decreases rapidly. We hypothesize that this is because the informational click model explores more

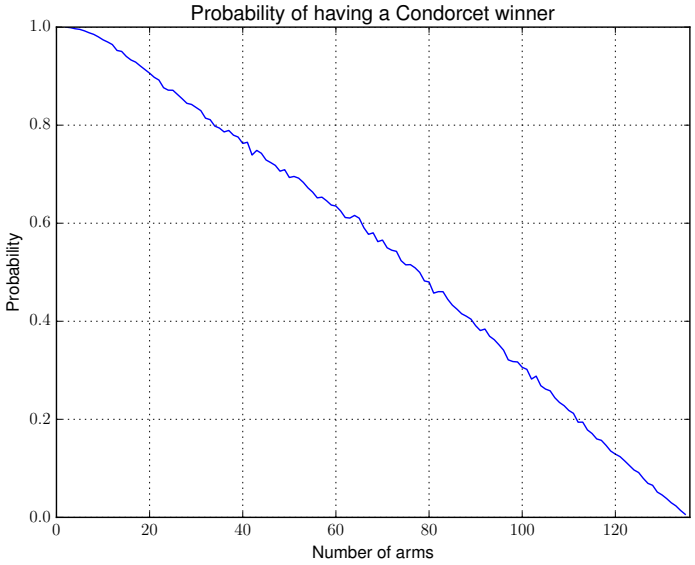

Figure 7.1: The probability that the Condorcet assumption holds for subsets of the feature rankers in the MSLR dataset. The probability is shown as a function of the size of the subset.

of the list of ranked documents than the navigational click model, which was used in Chapter 4, and so it is more likely to encounter non-transitivity phenomena of the sort described in [29].

7.1.2 Other Notions of Winners

As mentioned before, numerous other definitions of what constitutes the best arm have been proposed, some of which specialize to the Condorcet winner, when it exists. This latter property is desirable both in preference learning and social choice theory: the Condorcet winner is the choice that is preferred over all other choices, so if it exists, there is good reason to insist on selecting it. The Copeland winner, as discussed in this chapter, and the von Neumann winner [24, 58] satisfy this property, while the Borda (a.k.a. Sum of Expectations) and the Random Walk (a.k.a. PageRank) winners [14] do not. The von Neumann winner is in fact defined as a distribution over arms such that playing it will maximize the probability to beat any fixed arm. The Borda winner is defined as the arm maximizing the score $\sum_{j \neq i} p_{ij}$ and can be interpreted as the arm that beats other arms by the most, rather than beating the most arms. The Random Walk winner is defined as the arm we are most likely to visit in some Markov Chain determined by the preference matrix. In this section, we provide some numerical evidence for the similarity of these notions in practice, based on the sampled preference matrices obtained from the ranker evaluation from IR, which was described in the Section 7.1.1. Table 7.1 lists the percentage of preference matrices for which pairs of winners overlap. In the case of the von Neumann

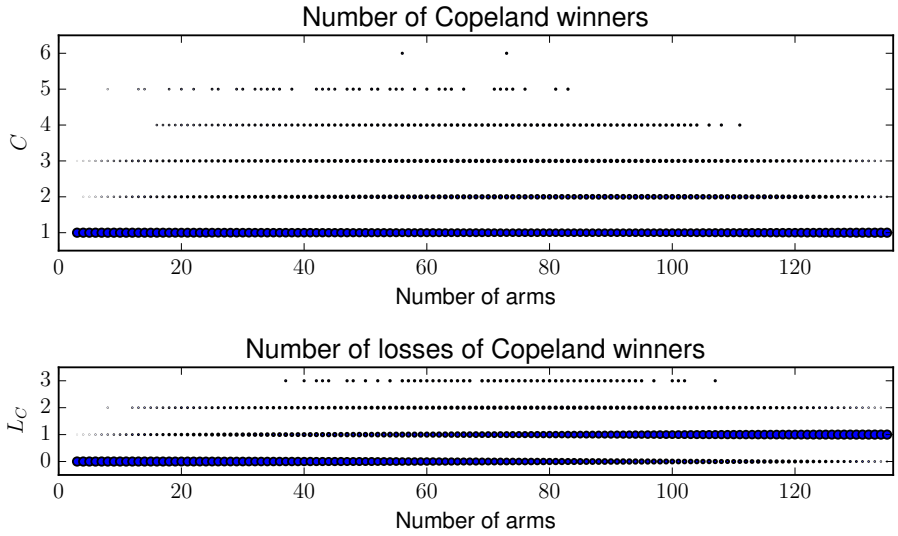

Figure 7.2: Observed values of the parameters C and L_C: the area of the circle with coordinates (x, y) is proportional to the percentage of examples with $K = x$ which satisfied $C = y$ (in the top plot) or $L_C = y$ in the bottom plot.

winner, which is defined as a probability distribution over the set of arms [24], we used the support of the distribution (i.e., the set of arms with non-zero probability) to define overlap with the other definitions.

Table 7.1: Percentage of matrices for which the different notions of winners overlap in the experimental setup described in §7.1.1.

Overlap	Copeland	von Neumann	Borda	Random Walk
Copeland	100%	99.94%	51.49%	56.15%
von Neumann	99.94%	100%	77.66%	82.11%
Borda	51.49%	77.66%	100%	94.81%
RandomWalk	56.15%	82.11%	94.81%	100%

As these numbers demonstrate, the Copeland and the von Neumann winners are very likely to overlap, as are the Borda and Random Walk winners, while the first two definitions are more likely to be incompatible with the latter two. Furthermore, in the case of 94.2% of the preference matrices, all Copeland winners were contained in the support of the von Neumann winner, suggesting that in practice the Copeland winner is a more restrictive notion of what constitutes a winner.

7.1.3 The Quantities C and L_C

We also examine additional quantities relevant to our regret bounds: the number of Copeland winners, C; the number of losses of each Copeland winner, L_C; and the range of values in which these quantities fall. Using the above randomly chosen preference sub-matrices, we counted the number of times each possible value for C and L_C was observed. The results are depicted in Figure 7.2: the area of the circle with coordinates (x, y) is proportional to the percentage of examples with $K = x$ which satisfied $C = y$ (in the top plot) or $L_C = y$ (in the bottom plot). As these plots show, the parameters C and L_C are generally much lower than K.

7.2 The CCB Algorithm

In this section, we present CCB (see Algorithm 5), which is based on the principle of *optimism followed by pessimism*: it maintains optimistic and pessimistic estimates of the preference matrix, i.e., matrices $\mathbf{U}$ and $\mathbf{L}$ (Line 6). It uses $\mathbf{U}$ to choose an *optimistic Copeland winner* a_c (Lines 7–9 and 11–12), i.e., an arm that has some chance of being a Copeland winner. Then, it uses $\mathbf{L}$ to choose an *opponent* a_d (Line 13), i.e., an arm deemed likely to discredit the hypothesis that a_c is indeed a Copeland winner.

More precisely, an optimistic estimate of the Copeland score of each arm a_i is calculated using $\mathbf{U}$ (Line 7), and a_c is selected from the set of top scorers, with preference given to those in a shortlist, $\mathcal{B}_t$ (Line 11). These are arms that have, roughly speaking, been optimistic winners throughout history. To maintain $\mathcal{B}_t$, as soon as CCB discovers that the optimistic Copeland score of an arm is lower than the pessimistic Copeland score of another arm, it purges the former from $\mathcal{B}_t$ (Line 9B).

The mechanism for choosing the opponent a_d is as follows. The matrices $\mathbf{U}$ and $\mathbf{L}$ define a confidence interval around p_{ij} for each i and j. In relation to a_c, there are three types of arms:

(1) arms a_j s.t. the confidence region of p_{cj} is strictly above 0.5,

(2) arms a_j s.t. the confidence region of p_{cj} is strictly below 0.5, and

(3) arms a_j s.t. the confidence region of p_{cj} contains 0.5.

Note that an arm of type (1) or (2) at time t' may become an arm of type (3) at time $t > t'$ even without queries to the corresponding pair as the size of the confidence intervals increases as time goes on.

CCB always chooses a_d from arms of type (3) because comparing a_c and a type (3) arm is most informative about the Copeland score of a_c. Among arms of type (3), CCB favors those that have confidently beaten arm a_c in the past (Line 13), i.e., arms that in some round $t' < t$ were of type (2). Such arms are maintained in a shortlist of "formidable" opponents ($\mathcal{B}_t^i$) that are likely to confirm that a_i is not a Copeland winner; these arms are favored when selecting a_d (Lines 10 and 13).

The sets $\mathcal{B}_t^i$ are what speed up the elimination of non-Copeland winners, enabling regret bounds that scale asymptotically with K rather than K^2. Specifically, for a non-Copeland winner a_i, the set $\mathcal{B}_t^i$ will eventually contain $L_C + 1$ strong opponents for a_i

(Line 9C), where L_C is the number of losses of each Copeland winner. Since L_C is typically small, as discussed in §7.1, asymptotically this leads to a bound of only $\mathcal{O}(\log T)$ on the number of time-steps when a_i is chosen as an optimistic Copeland winner, instead of a bound of $\mathcal{O}(K \log T)$, which a more naive algorithm would produce.

Algorithm 5 Copeland Confidence Bounds

Require: A Copeland dueling bandit problem and an exploration parameter $\alpha > \frac{1}{2}$.

1: $\mathbf{W} = [w_{ij}] \leftarrow \mathbf{0}_{K \times K}$ // 2D array of wins: w_{ij} is the number of times a_i beat a_j

2: $\mathcal{B}_1 = \{a_1, \ldots, a_K\}$ // potential best arms

3: $\mathcal{B}_1^i = \varnothing$ for each $i = 1, \ldots, K$ // potential to beat a_i

4: $\overline{L}_C = K$ // estimated max losses of a Copeland winner

5: **for** $t = 1, 2, \ldots$ **do**

6: Define the matrces

$$\mathbf{U} := [u_{ij}] = \frac{\mathbf{W}}{\mathbf{W} + \mathbf{W}^T} + \sqrt{\frac{\alpha \ln t}{\mathbf{W} + \mathbf{W}^T}}$$

$$\mathbf{L} := [l_{ij}] = \frac{\mathbf{W}}{\mathbf{W} + \mathbf{W}^T} - \sqrt{\frac{\alpha \ln t}{\mathbf{W} + \mathbf{W}^T}}$$

with $u_{ii} = l_{ii} = \frac{1}{2}, \forall i$ // All operations are element-wise; $\frac{x}{0} := 1$ for any x.

7: $\overline{\text{Cpld}}(a_i) = \#\left\{k \,|\, u_{ik} \geq \frac{1}{2}, k \neq i\right\}$ and $\underline{\text{Cpld}}(a_i) = \#\left\{k \,|\, l_{ik} \geq \frac{1}{2}, k \neq i\right\}$ // Optimistic and pessimistic Copeland score, respectively

8: $\mathcal{C}_t = \{a_i \,|\, \overline{\text{Cpld}}(a_i) = \max_j \overline{\text{Cpld}}(a_j)\}$

9: Set $\mathcal{B}_t \leftarrow \mathcal{B}_{t-1}$ and $\mathcal{B}_t^i \leftarrow \mathcal{B}_{t-1}^i$ and update as follows:

A. **Reset disproven hypotheses:** If for any i and $a_j \in \mathcal{B}_t^i$ we have $l_{ij} > 0.5$, reset $\mathcal{B}_t$, $\overline{L}_C$ and $\mathcal{B}_t^k$ for all k (i.e., set them to their original values as in Lines 2–4 above).

B. **Remove non-Copeland winners:** For each $a_i \in \mathcal{B}_t$, if $\overline{\text{Cpld}}(a_i) < \underline{\text{Cpld}}(a_j)$ holds for any j, set $\mathcal{B}_t \leftarrow \mathcal{B}_t \setminus \{a_i\}$, and if $|\mathcal{B}_t^i| \neq \overline{L}_C + 1$, then set $\mathcal{B}_t^i \leftarrow \{a_k | u_{ik} < 0.5\}$. However, if $\mathcal{B}_t = \varnothing$, reset $\mathcal{B}_t$, $\overline{L}_C$ and $\mathcal{B}_t^k$ for all k.

C. **Add Copeland winners:** For any $a_i \in \mathcal{C}_t$ with $\overline{\text{Cpld}}(a_i) = \underline{\text{Cpld}}(a_i)$, set $\mathcal{B}_t \leftarrow \mathcal{B}_t \cup \{a_i\}$, $\mathcal{B}_t^i \leftarrow \varnothing$ and $\overline{L}_C \leftarrow K - 1 - \overline{\text{Cpld}}(a_i)$. For each $j \neq i$, if we have $|\mathcal{B}_t^j| < \overline{L}_C + 1$, set $\mathcal{B}_t^j \leftarrow \varnothing$, and if $|\mathcal{B}_t^j| > \overline{L}_C + 1$, randomly choose $\overline{L}_C + 1$ elements of $\mathcal{B}_t^j$ and remove the rest.

10: With probability $1/4$, sample (c, d) uniformly from the set

$$\{(i, j) | a_j \in \mathcal{B}_t^i \text{ and } 0.5 \in [l_{ij}, u_{ij}]\}$$

(if it is non-empty) and skip to Line 14.

11: If $\mathcal{B}_t \cap \mathcal{C}_t \neq \varnothing$, then with probability $2/3$, set $\mathcal{C}_t \leftarrow \mathcal{B}_t \cap \mathcal{C}_t$.

12: Sample a_c from $\mathcal{C}_t$ uniformly at random.

13: With probability $1/2$, choose the set $\mathcal{B}^i$ to be either $\mathcal{B}_t^i$ or $\{a_1, \ldots, a_K\}$ and then set $d \leftarrow \arg\max_{\{j \in \mathcal{B}^i \,|\, l_{jc} \leq 0.5\}} u_{jc}$. If there is a tie, d is not allowed to be equal to c.

14: Compare arms a_c and a_d and increment w_{cd} or w_{dc} depending on which arm wins.

7.3 Theory

In this section, we present our main regret bound for CCB. Assuming that the number of Copeland winners and the number of losses of each Copeland winner are bounded (cf. §7.1), CCB's regret bound takes the form $\mathcal{O}(K^2 + K \log T)$.

Throughout this section we impose the following condition on the preference matrix:

A There are no ties, i.e., for all pairs (a_i, a_j) with $i \neq j$, we have $p_{ij} \neq 0.5$.

This assumption is not very restrictive in practice. For example, in the ranker evaluation setting from information retrieval, each arm corresponds to a ranker, a complex and highly engineered system, so it is unlikely that two rankers are indistinguishable. Furthermore, some of the results we present in this section actually hold under even weaker assumptions. However, for the sake of clarity, we defer a discussion of these nuanced differences to §7.4.

In this section, we provide a rough outline of our argument for the bound on the regret accumulated by Algorithm 5. For a more detailed argument, the interested reader is referred to §7.4.

Consider a K-armed Copeland bandit problem with arms $a_1, \ldots, a_K$ and preference matrix $\mathbf{P} = [p_{ij}]$, such that arms $a_1, \ldots, a_C$ are the Copeland winners, with C being the number of Copeland winners. Moreover, we define L_C to be the number of arms to which a Copeland winner loses in expectation.

Using this notation, our expected regret bound for CCB takes the form:

$$\mathcal{O}\left(\frac{K^2 + (C + L_C)K \ln T}{\Delta^2}\right) \tag{7.1}$$

Here, Δ is a notion of gap defined in §7.4, which is an improvement upon the smallest gap between any pair of arms.

This result is proven in two steps. First, we bound the number of comparisons involving non-Copeland winners, yielding a result of the form $\mathcal{O}(K^2 \ln T)$. Second, Theorem 7.1 closes the gap between this bound and the one in (7.1) by showing that, beyond a certain time horizon, CCB selects non-Copeland winning arms as the optimistic Copeland winner very infrequently.

Theorem 7.1. *Given a Copeland bandit problem satisfying Assumption* **A** *and any* $\delta > 0$ *and* $\alpha > 0.5$*, there exist constants* $A_\delta^{(1)}$ *and* $A_\delta^{(2)}$ *such that, with probability* $1 - \delta$*, the regret accumulated by CCB is bounded by the following:*

$$A_\delta^{(1)} + A_\delta^{(2)} \sqrt{\ln T} + \frac{2K(C + L_C + 1)}{\Delta^2} \ln T.$$

Using the high probability regret bound given in Theorem 7.1, we can deduce the expected regret result claimed in (7.1) for $\alpha > 1$, as a corollary by integrating δ over the interval $[0, 1]$.

Table 7.2: List of notation used in this chapter

Symbol	Definition
K	Number of arms
$[K]$	The set $\{1, \ldots, K\}$
$a_1, \ldots, a_K$	Set of arms
p_{ij}	Probability of arm a_i beating arm a_j
$\text{Cpld}(a_i)$	Copeland score: number of arms that a_i beats, i.e. $\lvert\{j \mid p_{ij} > 0.5\}\rvert$
C	Number of Copeland winners, i.e. arms a_i with $\text{Cpld}(a_i) \geq \text{Cpld}(a_j)$ for all j
$a_1, \ldots, a_C$	Copeland winner arms
α	UCB parameter of Algorithm 5
δ	Probability of failure
$C(\delta)$	$\left(\frac{(4\alpha-1)K^2}{(2\alpha-1)\delta}\right)^{\frac{1}{2\alpha-1}}$
$N_i(t)$	Number of times arm a_i was chosen as the optimistic Copeland winner until time t
$N_i^\delta(t)$	Number of times arm a_i was chosen as the optimistic Copeland winner in the interval $(C(\delta), t]$
$N_{ij}(t)$	Total number of time-steps before t when a_i was compared against a_j (notice that this definition is symmetric with respect to i and j)
$N_{ij}^\delta(t)$	Number of time-steps between times $C(\delta)$ and t when a_i was chosen as the optimistic Copeland winner and a_j as the challenger (note that, unlike $N_{ij}(t)$, this definition is not symmetric with respect to i and j)
τ_{ij}	The last time-step when a_i was chosen as the optimistic Copeland winner and a_j as the challenger (note that $\tau_{ij} \geq C(\delta)$ iff $N_{ij}^\delta(t) > 0$)
$w_{ij}(t)$	Number of wins of a_i over a_j until time t
$u_{ij}(t)$	$\frac{w_{ij}(t)}{N_{ij}(t)} + \sqrt{\frac{\alpha \ln t}{N_{ij}(t)}}$
$l_{ij}(t)$	$1 - u_{ji}(t)$
$\overline{\text{Cpld}}(a_i)$	$\#\left\{k \mid u_{ik} \geq \frac{1}{2}, k \neq i\right\}$
$\underline{\text{Cpld}}(a_i)$	$\#\left\{k \mid l_{ik} \geq \frac{1}{2}, k \neq i\right\}$
$\overline{\mathcal{C}_t}$	$\{i \mid \overline{\text{Cpld}}(a_i) = \max_j \overline{\text{Cpld}}(a_j)\}$
$\mathcal{L}_i$	the set of arms to which a_i loses, i.e. a_j such that $p_{ij} < 0.5$
L_C	The largest number of losses that any Copeland winner has, i.e. $\max_{i=1}^{C} \lvert\{j \mid p_{ij} < 0.5\}\rvert$
$\overline{L}_C$	Algorithm 5's estimate of L_C
$\mathcal{B}_t$	The potentially best arms at time t, i.e. the set of arms that according to Algorithm 5 have some chance of being Copeland winners
$\mathcal{B}_t^i$	The arms that at time t have the best chance of beating arm a_i (Cf. Line 12 in Algorithm 5)
Δ_{ij}	$\lvert p_{ij} - 0.5\rvert$
$\Delta_{\min}$	$\min\{\Delta_{ij} \mid \Delta_{ij} \neq 0\}$
i^*	the index of the $(L_C+1)^{th}$ largest element in the set $\{\Delta_{ij} \mid p_{ij} < 0.5\}$ in the case that $i > C$

Table 7.3: List of notation used in this chapter (Cont'd)

Symbol	Definition
Δ_i^*	$\begin{cases} \Delta_{ii^*} & \text{if } i > C \\ 0 & \text{otherwise} \end{cases}$
Δ_{ij}^*	$\begin{cases} \Delta_i^* + \Delta_{ij} & \text{if } p_{ij} \geq 0.5 \\ \max\{\Delta_i^*, \Delta_{ij}\} & \text{otherwise} \end{cases}$
	(See Figures 7.5 and 7.4 for a pictorial explanation.)
$\Delta_{\min}^*$	$\min_{i>C} \Delta_i^*$
$\widehat{N}_{ij}^{\delta}(T)$	$\begin{cases} \frac{4\alpha \ln T}{(\Delta_{ij}^*)^2} & \text{if } i \neq j \\ 0 & \text{if } i = j \text{ and } i > C \end{cases}$
$\widehat{N}_i^{\delta}(T)$	$\sum_{j=1}^{K} \widehat{N}_{ij}^{\delta}(T)$
$\widehat{N}^{\delta}(T)$	$\sum_{i \neq j} \widehat{N}_{ij}^{\delta}(T) + 1$
$T_\delta \geq$	$C(\frac{\delta}{2}) + 8K^2(L_C+1)^2 \ln \frac{6K^2}{\delta} + K^2 \ln \frac{6K}{\delta}$ $+ \frac{32\alpha K(L_C+1)}{\Delta_{\min}^2} \ln T_\delta + \widehat{N}^{\delta/2}(T_\delta)$ $+4K \max_{i>C} \widehat{N}_i^{\delta/2}(T_\delta)$
	T_δ is the smallest integer satisfying the above inequality (Cf. Definition 7.4).
T_0	$C(\delta/2) + \widehat{N}^{\delta/2}(T_\delta)$ $+ \frac{32\alpha K(L_C+1)\ln T_\delta}{\Delta_{\min}^2}$ $+8K^2(L_C+1)^2 \ln \frac{6K^2}{\delta}$
n_b	$2K\widehat{N}_b^{\delta/2}(\widehat{T}_\delta) + \frac{K^2 \ln(4K/\delta)}{2}$
$Binom(n,p)$	A "binomial" random variable obtained from the sum of n independent Bernoulli random variables, each of which produces 1 with probability p and 0 otherwise.
Δ_i	$\max\left\{\text{cpld}(a_1) - \text{cpld}(a_i), \frac{1}{K-1}\right\}$
H_i	$\sum_{j \neq i} \frac{1}{\Delta_{ij}^2}$
H_∞	$\max_i H_i$
Δ_i^ϵ	$\max\{\Delta_i, \epsilon(1 - \text{cpld}(a_1))\}$

7.4 Proofs

In this section, provide the proofs for the result claimed in the previous section, starting with a rough outline of the argument in §7.4.1, followed by detailed proofs of the needed lemmas.

7.4.1 An Outline of the Proof of Theorem 7.1

To analyze Algorithm 5, consider a K-armed Copeland bandit problem with arms $a_1, \ldots, a_K$ and preference matrix $\mathbf{P} = [p_{ij}]$, such that arms $a_1, \ldots, a_C$ are the Copeland winners, with C being the number of Copeland winners. Throughout this section, we assume that the parameter α in Algorithm 5 satisfies $\alpha > 0.5$, unless otherwise stated. We first define the relevant quantities:

Definition 7.2. Given the above setting we define:[1]

1. $\mathcal{L}_i := \{a_j \,|\, p_{ij} < 0.5\}$, i.e., the arms to which a_i loses, and $L_C := |\mathcal{L}_1|$.
2. $\Delta_{ij} := |p_{ij} - 0.5|$ and $\Delta_{\min} := \min_{i \neq j} \Delta_{ij}$.
3. Given $i > C$, define i^* as the index of the $(L_C + 1)^{th}$ largest element in the set $\{\Delta_{ij} \,|\, p_{ij} < 0.5\}$.
4. Define Δ_i^* to be Δ_{ii^*} if $i > C$ and 0 otherwise. Moreover, let us set $\Delta_{\min}^* := \min_{i>C} \Delta_i^*$.
5. Define Δ_{ij}^* to be $\Delta_i^* + \Delta_{ij}$ if $p_{ij} \geq 0.5$ and $\max\{\Delta_i^*, \Delta_{ij}\}$ otherwise.[2]
6. $\Delta := \min\{\min_{i \leq C < j} \Delta_{ij}, \Delta_{\min}^*\}$, where $\Delta_{\min}^*$ is defined as in item 4 above.
7. $C(\delta) := \left((4\alpha - 1)K^2/(2\alpha - 1)\delta\right)^{\frac{1}{2\alpha - 1}}$ where α is as in Algorithm 5.
8. $N_{ij}^{\delta}(t)$ is the number of time-steps between times $C(\delta)$ and t when a_i was chosen as the optimistic Copeland winner and a_j as the challenger. Also, $\widehat{N}_{ij}^{\delta}(t)$ is defined to be $(4\alpha \ln t)/\left(\Delta_{ij}^*\right)^2$ if $i \neq j$, 0 if $i = j > C$ and t if $i = j \leq C$. We also define $\widehat{N}^{\delta}(t) := \sum_{i \neq j} \widehat{N}_{ij}^{\delta}(t) + 1$.

Using this notation, our expected regret bound for CCB takes the form:

$$\mathcal{O}\left(\frac{K^2 + (C + L_C)K \ln T}{\Delta^2}\right) \tag{7.2}$$

This result is proven in two steps. First, Proposition 7.3 bounds the number of comparisons involving non-Copeland winners, yielding a result of the form $\mathcal{O}(K^2 \ln T)$. Second, Theorem 7.10 closes the gap between this bound and that of (7.2) by showing that,

[1] See Tables 7.2 and 7.3 for a summary of the definitions used in this chapter.
[2] See Figures 7.4 and 7.5 for a pictorial explanation.

beyond a certain time horizon, CCB selects non-Copeland winning arms as the optimistic Copeland winner very infrequently.

Note that we have $\Delta^*_{ij} \geq \Delta_{ij}$ for all pairs $i \neq j$. Thus, for simplicity, the analysis in this section can be read as if the bounds were given in terms of Δ_{ij}. We use Δ^*_{ij} instead because it gives tighter upper bounds. In particular, simply using the gaps Δ_{ij} would replace the denominator of the expression in (7.2) with $\Delta^2_{\min}$, which leads to a substantially worse regret bound in practice. For instance, in the ranker evaluation application used in the experiments, this change would on average increase the regret bound by a factor that is of the order of tens of thousands. See §7.4.2 for a more quantitative discussion of this point.

We can now state our first bound, proved in §7.4.4 under weaker assumptions.

Proposition 7.3. *Given any $\delta > 0$ and $\alpha > 0.5$, if we apply CCB (Algorithm 5) to a dueling bandit problem satisfying Assumption* **A***, the following holds with probability $1 - \delta$: for any $T > C(\delta)$ and any pair of arms a_i and a_j, we have $N^{\delta}_{ij}(T) \leq \widehat{N}^{\delta}_{ij}(T)$.*

One can sum the inequalities in the last proposition over pairs (i, j) to get a regret bound of the form $\mathcal{O}(K^2 \log T)$ for Algorithm 5. However, as Theorem 7.10 will show, we can use the properties of the sets $\mathcal{B}^i_t$ to obtain a tighter regret bound of the form $\mathcal{O}(K \log T)$. Before stating that theorem, we need a few definitions and lemmas. We begin by defining the key quantity:

Definition 7.4. Given a preference matrix $\mathbf{P}$ and $\delta > 0$, then T_δ is the smallest integer satisfying

$$\begin{aligned} T_\delta \geq C(\frac{\delta}{2}) &+ 8K^2(L_C + 1)^2 \ln \frac{6K^2}{\delta} \\ &+ K^2 \ln \frac{6K}{\delta} \\ &+ \frac{32\alpha K(L_C + 1)}{\Delta^2_{\min}} \ln T_\delta \\ &+ \widehat{N}^{\frac{\delta}{2}}(T_\delta) \\ &+ 4K \max_{i>C} \widehat{N}^{\frac{\delta}{2}}_i(T_\delta). \end{aligned}$$

Remark 7.5. T_δ is $\mathrm{poly}(K, \delta^{-1})$ and our regret bound below scales as $\log T_\delta$.

The following two lemmas are key to the proof of Theorem 7.10. Lemma 7.6 (proved in §7.4.5) states that, with high probability by time T_δ, each set $\mathcal{B}^i_t$ contains $L_C + 1$ arms a_j, each of which beats a_i (i.e., $p_{ij} < 0.5$). This fact then allows us to prove Lemma 7.7 (§7.4.6), which states that, after time-step T_δ, the rate of suboptimal comparisons is $\mathcal{O}(K \ln T)$ rather than $\mathcal{O}(K^2 \ln T)$.

Lemma 7.6. *Given $\delta > 0$, with probability $1 - \delta$, each set $\mathcal{B}^i_{T_\delta}$ with $i > C$ contains exactly $L_C + 1$ elements with each element a_j satisfying $p_{ij} < 0.5$. Moreover, for all $t \in [T_\delta, T]$, we have $\mathcal{B}^i_t = \mathcal{B}^i_{T_\delta}$.*

Lemma 7.7. *Given a Copeland bandit problem satisfying Assumption* **A** *and any $\delta > 0$, with probability $1 - \delta$ the following holds: the number of time-steps between $T_{\delta/2}$ and T when each non-Copeland winner a_i can be chosen as optimistic Copeland winners (i.e., times when arm a_c in Algorithm 5 satisfies $c > C$) is bounded by*

$$\widehat{N}^i := 2\widehat{N}^i_{\mathcal{B}} + 2\sqrt{\widehat{N}^i_{\mathcal{B}}} \ln \frac{2K}{\delta},$$

where $\widehat{N}^i_{\mathcal{B}} := \sum_{j \in \mathcal{B}^i_{T_{\delta/2}}} \widehat{N}^{\delta/4}_{ij}(T)$.

Remark 7.8. Due to Lemma 7.6, with high probability we have $\widehat{N}^i_{\mathcal{B}} \leq \frac{(L_C+1)\ln T}{\left(\Delta^*_{\min}\right)^2}$ for each $i > C$ and so the total number of times between T_δ and T when a non-Copeland winner is chosen as an optimistic Copeland winner is in $\mathcal{O}(KL_C \ln T)$ for a fixed minimal gap $\Delta^*_{\min}$. The only other way a suboptimal comparison can occur is if a Copeland winner is compared against a non-Copeland winner, and according to Proposition 7.3, the number of such occurrences is bounded by $\mathcal{O}(KC \ln T)$. Hence, the number of suboptimal comparisons is in $\mathcal{O}(K \ln T)$ assuming that C and L_C are bounded. In §7.1, we provide experimental evidence for this.

We now define the quantities needed to state the main theorem.

Definition 7.9. We define the following three quantities:

$$A^{(1)}_\delta := C(\delta/4) + \widehat{N}^\delta(T_{\delta/2})$$
$$A^{(2)}_\delta := \sum_{i>C} \frac{\sqrt{L_C+1}}{\Delta^*_i} \ln \frac{2K}{\delta}$$
$$A^{(3)} := \sum_{i \leq C < j} \frac{1}{\left(\Delta_{ij}\right)^2} + 2\sum_{i>C} \frac{L_C+1}{\left(\Delta^*_i\right)^2}$$

Finally, we repeat the statement of Theorem 7.1 for the reader's convenience.

Theorem 7.10. *Given a Copeland bandit problem satisfying Assumption* **A** *and any $\delta > 0$ and $\alpha > 0.5$, with probability $1 - \delta$, the regret accumulated by CCB is bounded by the following:*

$$A^{(1)}_\delta + A^{(2)}_\delta \sqrt{\ln T} + A^{(3)} \ln T \;\leq\; A^{(1)}_\delta + A^{(2)}_\delta \sqrt{\ln T} + \frac{2K(C + L_C + 1)}{\Delta^2} \ln T.$$

For a general assessment of the above quantities, assuming that L_C and C are both $\mathcal{O}(1)$, the above quantities in terms of K become $A^{(1)}_\delta = \mathcal{O}(K^2)$, $A^{(2)}_\delta = \mathcal{O}(K \log(K))$, $A^{(3)} = \mathcal{O}(K)$. Hence, the above bound boils down to the expression in (7.2). We now turn to the proof of the theorem.

Proof of Theorem 7.10. Let us consider the two disjoint time-intervals $[1, T_{\delta/2}]$ and $(T_{\delta/2}, T]$:

[1,$\mathbf{T}_{\delta/2}$]: In this case, applying Proposition 7.3 to T_δ, we get that the number of time-steps when a non-Copeland winner was compared against another arm is bounded by $A_\delta^{(1)}$. As the maximum regret such a comparison can incur is 1, this deals with the first term in the above expression.

($\mathbf{T}_{\delta/2}$,T]: In this case, applying Lemma 7.7, we get the other two terms in the above regret bound. □

Now that we have the high probability regret bound given in Theorem 7.10, we can deduce the expected regret result claimed in (7.2) for $\alpha > 1$, as a corollary by integrating δ over the interval $[0, 1]$.

7.4.2 The Gap Δ

The regret bound for CCB, given in (7.2), depends on the gap Δ defined in Definition 7.2(6), rather than the smallest gap $\Delta_{\min}$ as specified in Definition 7.2(2). The latter would result in a looser regret bound and Figure 7.3 quantifies this deterioration in the ranker evaluation example under consideration here. In particular, the plot depicts the average of the ratio between the two bounds (the one using Δ and the one using $\Delta_{\min}$) across the $10,000$ sampled preference matrices used in the analysis of the Condorcet winner for each K in the set $\{2, \ldots, 135\}$. The average ratio decreases as the number of arms approaches 136 because, as K increases, the sampled preference matrices increasingly resemble the full preference matrix and so their gaps Δ and $\Delta_{\min}$ approach those of the full 136-armed preference matrix as well. As it turns out, the ratio $\Delta^2/\Delta_{\min}^2$ for the full matrix is equal to $1,419$. Hence, the curve in Figure 7.3 approaches that number as the number of arms approaches 136.

7.4.3 Background Material

Let us begin by reminding the reader of some useful lemmas that we will make use of repeatedly in the rest of this section:

Maximal Azuma-Hoeffding Bound [17, §A.1.3]: Given random variables $X_1, \ldots, X_N$ with common range $[0, 1]$ satisfying $\mathbf{E}[X_n | X_1, \ldots, X_{n-1}] = \mu$, define the partial sums $S_n = X_1 + \cdots + X_n$. Then, for all $a > 0$, we have

$$P\Big(\max_{n \leq N} S_n > n\mu + a\Big) \leq e^{-2a^2/N}$$
$$P\Big(\min_{n \leq N} S_n < n\mu - a\Big) \leq e^{-2a^2/N}$$

We also repeat the following lemma from Chapter 4 for the reader's convenience:

Lemma 7.11 (Lemma 4.1)**.** *Let $\mathbf{P} := [p_{ij}]$ be the preference matrix of a K-armed dueling bandit problem with arms $\{a_1, \ldots, a_K\}$. Then, for any dueling bandit algorithm and any $\alpha > \frac{1}{2}$ and $\delta > 0$, we have*

$$P\Big(\forall\, t > C(\delta), i, j,\ p_{ij} \in [l_{ij}(t), u_{ij}(t)]\Big) > 1 - \delta.$$

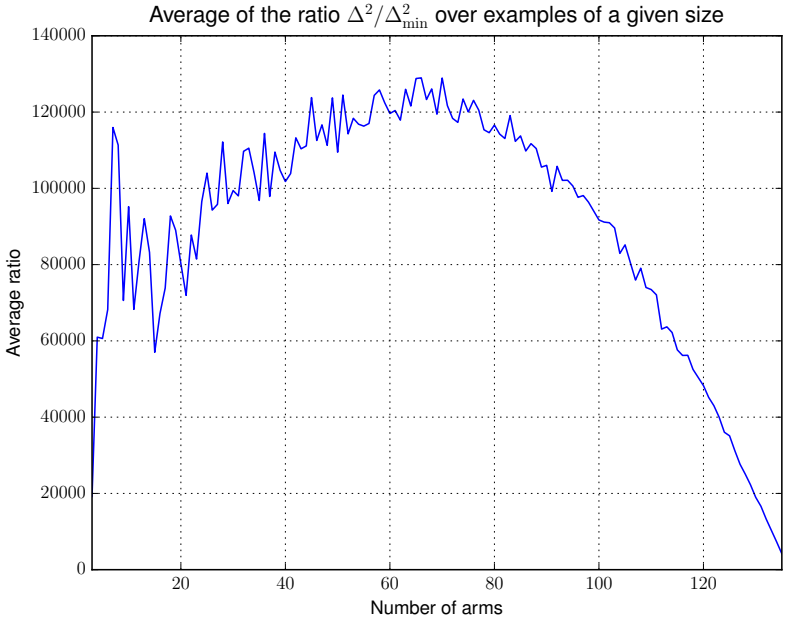

Figure 7.3: The average advantage gained by having the bound in (7.2) depend on Δ rather than $\Delta_{\min}$: for each number of arms K, the expectation is taken across the $10,000$ K-armed preference matrices obtained using the sampling procedure described in §7.4.2.

7.4.4 Proof of Proposition 7.3

Before starting with the proof, let us point out the following two properties that can be derived from Assumption **A** in Section 7.3:

P1 There are no ties involving a Copeland winner and a non-Copeland winner, i.e., for all pairs of arms (a_i, a_j) with $i \leq C < j$, we have $p_{ij} \neq 0.5$.

P2 Each non-Copeland winner has more losses than every Copeland winner, i.e., for every pair of arms (a_i, a_j), with $i \leq C < j$, we have $|\mathcal{L}_i| < |\mathcal{L}_j|$.

Even though we have assumed in the statement of Proposition 7.3 that Assumption **A** holds, it turns out that the proof provided in this section holds as long as the above two properties hold.

Proposition 7.3 *Applying CCB to a dueling bandit problem satisfying properties* **P1** *and* **P2***, we have the following bounds on the number of comparisons involving various arms for each $T > C(\delta)$: for each pair of arms a_i and a_j, such that either at least one of them is not a Copeland winner or $p_{ij} \neq 0.5$, with probability $1 - \delta$ we have*

$$N_{ij}^{\delta}(T) \leq \widehat{N}_{ij}^{\delta}(T) := \begin{cases} \dfrac{4\alpha \ln T}{\left(\Delta_{ij}^{*}\right)^2} & \text{if } i \neq j \\ 0 & \text{if } i = j > C \end{cases} \tag{7.3}$$

Proof of Proposition 7.3. We will prove these bounds by considering a number of cases separately:

1. $i \leq C$ **and** $p_{ij} \neq 0.5$: First of all, since a_i is a Copeland winner, this means that according to the definitions in Tables 7.2 and 7.3, Δ^*_{ij} is simply equal to Δ_{ij}; secondly, assuming by way of contradiction that $N^{\delta}_{ij}(t) > \frac{4\alpha \ln T}{\Delta_{ij}} > 0$, then we have $\tau_{ij} > C(\delta)$ and so by Lemma 7.11, we have with probability $1 - \delta$ that the confidence interval $[l_{ij}(\tau_{ij}), u_{ij}(\tau_{ij})]$ contains the preference probability p_{ij}. But, in order for arm a_j to have been chosen as the challenger to a_i, we must also have $0.5 \in [l_{ij}(\tau_{ij}), u_{ij}(\tau_{ij})]$; to see this, let us consider the two possible cases:

   (a) If we have $p_{ij} > 0.5$, then having
   $$0.5 \notin [l_{ij}(\tau_{ij}), u_{ij}(\tau_{ij})]$$
   implies that we have $l_{ij}(\tau_{ij}) > 0.5$, which in turn implies
   $$u_{ji}(\tau_{ij}) = 1 - l_{ij}(\tau_{ij}) < 0.5 = u_{ii}(\tau_{ij}),$$
   but this is impossible since in that case a_i would have been chosen as the challenger.

   (b) If we have $p_{ij} < 0.5$, then have
   $$0.5 \notin [l_{ij}(\tau_{ij}), u_{ij}(\tau_{ij})]$$
   implies that we have $u_{ij}(\tau_{ij}) < 0.5$, but this is impossible because it means that we had $l_{ji}(\tau_{ij}) > 0.5$, and CCB would have eliminated it from considerations in its second round.

   So, in either case, we cannot have $0.5 \notin [l_{ij}(\tau_{ij}), u_{ij}(\tau_{ij})]$. Therefore, at time τ_{ij}, we must have had $u_{ij}(\tau_{ij}) - l_{ij}(\tau_{ij}) > |p_{ij} - 0.5| =: \Delta_{ij}$. From this, we can conclude the following, using the definition of u_{ij} and l_{ij}:
   $$\begin{aligned}
   & u_{ij}(\tau_{ij}) - l_{ij}(\tau_{ij}) := 2\sqrt{\frac{\alpha \ln \tau_{ij}}{N_{ij}(\tau_{ij})}} \geq \Delta_{ij} \\
   \therefore \quad & 2\sqrt{\frac{\alpha \ln \tau_{ij}}{N^{\delta}_{ij}(\tau_{ij})}} \geq \Delta_{ij} \quad \because \; N^{\delta}_{ij}(\tau_{ij}) \leq N_{ij}(\tau_{ij}) \\
   \therefore \quad & 2\sqrt{\frac{\alpha \ln T}{N^{\delta}_{ij}(\tau_{ij})}} \geq \Delta_{ij} \quad \because \; \tau_{ij} \leq T \\
   \therefore \quad & N^{\delta}_{ij}(\tau_{ij}) \leq \frac{4\alpha \ln T}{\Delta^2_{ij}},
   \end{aligned}$$
   giving us the desired bound. The reader is referred to Figure 7.4 for an illustration of this argument.

2. $C < i$: Let us deal with the two cases included in Inequality (7.3) separately:

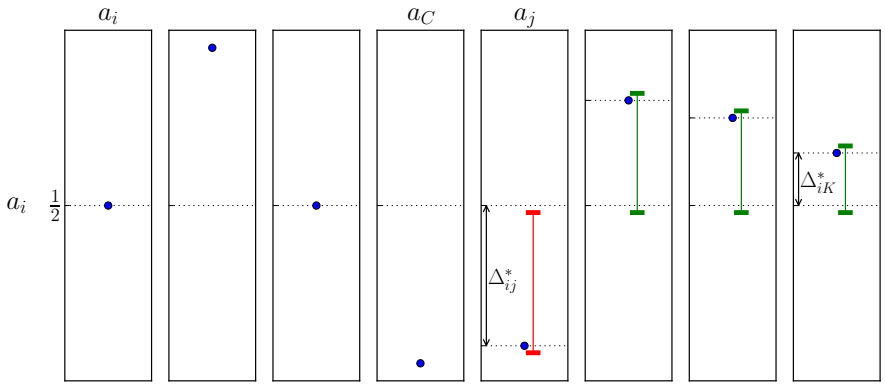

Figure 7.4: This figure illustrates the definition of the quantities Δ_i^* and Δ_{ij}^* in the case that arm a_i is a Copeland winner, as well as the idea behind Case 1 in the proof of Proposition 7.3. In this setting we have $\Delta_i^* = 0$ and $\Delta_{ij}^* = \Delta_{ij}$. On the one hand, by Lemma 7.11, we know that the confidence intervals will contain the p_{ij} (the blue dots in the plots), and on the other as soon as the confidence interval of p_{ij} stops containing 0.5 for some arm a_j, we know that it could not be chosen to be compared against a_i. In this way, the gaps Δ_{ij}^* regulate the number of times that each arm can be chosen to be played against a_i during time-steps when a_i is chosen as optimistic Copeland winner.

(a) $i = j > C$: In plain terms, this says that with probability $1 - \delta$ no non-Copeland winner will be compared against itself after time $C(\delta)$. The reason for this is the following set of facts:

- Since a_i is a non-Copeland winner, we have by Property **P1** that it loses to more arms than any Copeland winner.
- For a_i to have been chosen as an optimistic Copeland winner, it has to have (optimistically) lost to no more than L_C arms, which means that there exists an arm k such that $p_{ik} < 0.5$, but $u_{ik} \geq 0.5$.
- By Lemma 7.11, for all time steps after $C(\delta)$, we have $l_{ik} \leq p_{ik} < 0.5$, and so in the second round we have $u_{ki} > 0.5 = u_{ii}$, and so a_i could be not chosen as the challenger to itself.

(b) $i \neq j$: In the case that a_i is not a Copeland winner and a_j is different from a_i, we distinguish between the following two cases, where Δ_i^* is defined as in Tables 7.2 and 7.3:

i. $p_{ij} \leq 0.5 - \Delta_i^*$: In this case, the definition of Δ_i^* reduces to Δ_{ij}. Now, since when choosing the challenger, CCB eliminates from consideration any arm a_j that has $l_{ji} > 0.5$, the last time-step τ_{ij} after $C(\delta)$ when a_j was chosen as the challenger for a_i, we must have had $u_{ij}(\tau_{ij}) := 1 - l_{ji}(\tau_{ij}) \geq 0.5$. On the other hand, Lemma 7.11 implies that we must also have $l_{ij}(\tau_{ij}) \leq p_{ij}$, and therefore, we have $u_{ij}(\tau_{ij}) - l_{ij}(\tau_{ij}) \geq \Delta_{ij}$;

so, doing the same calculation as in part 1 of this proof, we have

$$\begin{aligned}
& u_{ij}(\tau_{ij}) - l_{ij}(\tau_{ij}) := 2\sqrt{\frac{\alpha \ln \tau_{ij}}{N_{ij}(\tau_{ij})}} \geq \Delta_{ij} \\
\therefore \quad & 2\sqrt{\frac{\alpha \ln \tau_{ij}}{N^{\delta}_{ij}(\tau_{ij})}} \geq \Delta_{ij} \quad \because \; N^{\delta}_{ij}(\tau_{ij}) \leq N_{ij}(\tau_{ij}) \\
\therefore \quad & 2\sqrt{\frac{\alpha \ln T}{N^{\delta}_{ij}(\tau_{ij})}} \geq \Delta_{ij} \quad \because \; \tau_{ij} \leq T \\
\therefore \quad & N^{\delta}_{ij}(\tau_{ij}) \leq \frac{4\alpha \ln T}{\Delta^2_{ij}},
\end{aligned}$$

ii. $p_{ij} > 0.5 - \Delta^*_i$: Repeating the above argument about $u_{ij}(\tau_{ij})$, we can deduce that $u_{ij}(\tau_{ij}) \geq 0.5$ must hold. Furthermore, Lemma 7.11 states that with probability $1 - \delta$ we have $u_{ij}(\tau_{ij}) \geq p_{ij}$. Putting these two together we get

$$u_{ij}(\tau_{ij}) \geq \max\{0.5, p_{ij}\}. \tag{7.4}$$

Moreover, we will show next that with probability $1 - \delta$, we have $l_{ij}(\tau_{ij}) \leq 0.5 - \Delta^*_i$; this is a consequence of the following facts:

- Since a_i was chosen as the optimistic Copeland winner, we can deduce that a_i had no more that L_C optimistic losses.
- Let $a_{k_1}, \ldots, a_{k_l}$ be the $l \leq L_C$ arms to which a_i lost optimistically during time-step τ_{ij}. Then, the smallest p_{ik} with $k \notin \{k_1, \ldots, k_l\}$, must be less than to equal to the $\{L_C + 1\}^{th}$ smallest element in the set $\{p_{ik} \,|\, k = 1, \ldots, K\}$.
- This, in turn, is equal to the $\{L_C + 1\}^{th}$ smallest element in the set $\{p_{ik} | p_{ik} < 0.5\}$ (since this latter set of numbers are the smallest ones in the former set). But, this is equal to $0.5 - \Delta^*_i$ by definition.

So, we have the desired bound on $l_{ij}(\tau_{ij})$ and combining this with Inequality (7.4), we have

$$u_{ij}(\tau_{ij}) - l_{ij}(\tau_{ij}) \geq \max\{0, p_{ij} - 0.5\} + \Delta^*_i = \Delta^*_{ij},$$

where the last equality follows directly from the definition of Δ^*_{ij} and the fact that $p_{ij} > 0.5 - \Delta^*_i$. Now, repeating the same calculations as before, we can conclude that with probability $1 - \delta$, we have

$$N^{\delta}_{ij}(\tau_{ij}) \leq \frac{4\alpha \ln T}{\left(\Delta^*_{ij}\right)^2}.$$

□

A pictorial depiction of the various steps in the second part of the proof can be found in Figure 7.5, where the bottom row of plots in the figure corresponds to the confidence intervals around probabilities p_{ij} (depicted using the blue dots) for $j = 1, \ldots, K$, while

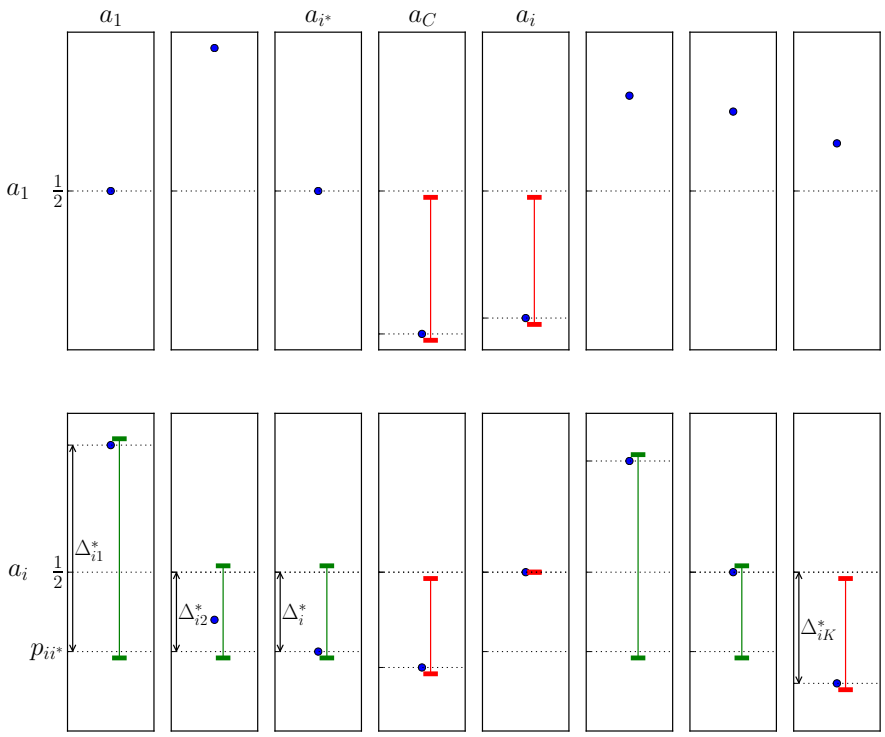

Figure 7.5: This figure illustrates the definition of the quantities Δ_i^* and Δ_{ij}^* in the case that arm a_i is not a Copeland winner, as well as the idea behind Case 2 in the proof of Proposition 7.3.

the top row corresponds to those for probabilities p_{1j}, where a_1 is by assumption one of the Copeland winners (although we could use any other Copeland winner instead).

The two boxes in the top row with red intervals represent arms to which a_1 loses (i.e. $p_{1j} < 0.5$), the number of which happens to be 2 in this example, which means that $L_C = 2$. Now, by Definition 7.2(3), i^* is the index with the index j with the $(L_C + 1)^{th}$ (in this case 3^{rd}) lowest p_{ij}, and since the three lowest p_{ij} in this example are p_{iK}, p_{iC} and p_{ii^*}, this means that the column labeled as a_{i^*} is indeed labeled correctly. Given this, Definition 7.2(4) tells us that Δ_i^* is the size of the gap shown in the block corresponding to pair (a_i, a_{i^*}).

Moreover, by Definition 7.2(5), the gap Δ_{ij}^* is defined using one of the following three cases: (1) if we have $p_{ij} < p_{ii^*}$ (as with the ones with red confidence intervals in the bottom row of plots), then we get $\Delta_{ij}^* := \Delta_{ij} = 0.5 - p_{ij}$; (2) if we have $p_{ii^*} < p_{ij} \leq 0.5$ (as in the plots in the $2^{nd}, 3^{rd}$ and 7^{th} column of the bottom row), then we get $\Delta_{ij}^* := \Delta_i^*$; (3) if we have $0.5 < p_{ij}$ (as in the 1^{st} and 6^{th} column in the bottom row), then we get $\Delta_{ij}^* := \Delta_{ij} + \Delta_i^*$.

The reasoning behind this trichotomy is as follows: in the case of arms a_j in group (1), they are not going to be chosen to be played against a_i as soon as the top of the interval

goes below 0.5, and by Lemma 7.11, we know that the bottom of the interval will be below p_{ij}. In the case of the arms in groups (2) and (3), the bottom of their interval needs to be below p_{ii^*} because otherwise that would mean that neither arm a_{i^*} nor arms in group (1) were eligible to be included in the $\arg\max$ expression in Line 13 of Algorithm 5, which can only happen if we have $u_{ij} < 0.5$ for $j = i^*$ as well as the arms in group (1), from which we can deduce that the optimistic Copeland score of a_i must have been lower than $K - 1 - L_C$, and so a_i could not have been chosen as an optimistic Copeland winner. Using the same argument, we can also see that the tops of the confidence intervals corresponding to arms in group (2) must be above 0.5, or else it would be impossible for a_i to be chosen as an optimistic Copeland winner. Moreover, by Lemma 7.11, the intervals of the arms a_j in group (3) must contain p_{ij}.

7.4.5 Proof of Lemma 7.6

Let us begin with the following direct corollary of Proposition 7.3:

Corollary 7.12. *Given any $\delta > 0$, any $T > C(\delta)$ and any sub-interval of length $\widehat{N}^\delta(T) := \sum_{i \neq j} \widehat{N}^\delta_{ij}(T) + 1$, with probability $1 - \delta$, there is at least one time-step when there exists $c \leq C$ such that*

$$\begin{aligned} \underline{\text{Cpld}}(a_c) = \text{Cpld}(a_c) &= \overline{\text{Cpld}(a_c)} \\ &\geq \overline{\text{Cpld}(a_j)} \ \ \forall\, j, \end{aligned} \tag{7.5}$$

Proof. According to Proposition 7.3, with probability $1-\delta$, there are at most $\sum_{i \neq j} \widehat{N}^\delta_{ij}(T)$ time-steps between $C(\delta)$ and T when Algorithm 5 did not compare a Copeland winner against itself: i.e. c and d in Algorithm 5 did not satisfy $c = d \leq C$.

In other words, during this time-period, in any sub-interval of length $\widehat{N}^\delta(T) := \sum_{i \neq j} \widehat{N}^\delta_{ij}(T) + 1$, there is at least one time-step when a Copeland winner was compared against itself. During this time-step, we must have had

$$\begin{aligned} \underline{\text{Cpld}}(a_c) = \text{Cpld}(a_c) &= \overline{\text{Cpld}(a_c)} \\ &\geq \overline{\text{Cpld}(a_j)} \ \ \forall\, j, \end{aligned}$$

where the first two equalities are due to the fact that in order for Algorithm 5 to set $c = d$, we must have $0.5 \notin [l_{cj}, u_{cj}]$ for each $j \neq c$, or else a_c would not be played against itself; on the other hand, the last inequality is due to the fact that a_c was chosen as an optimistic Copeland winner by Line 8 of Algorithm 5, so its optimistic Copeland score must have been greater than or equal to the optimistic Copeland score of the rest of the arms. □

Lemma 7.13. *If there exists an arm a_i with $i > C$ such that $\mathcal{B}^i_{C(\delta/2)}$ contains an arm a_j that loses to a_i (i.e. $p_{ij} > 0.5$) or such that $\mathcal{B}^i_{C(\delta/2)}$ contains fewer than $L_C + 1$ arms, then the probability that by time-step T_0 the sets $\mathcal{B}^i_t$ and $\mathcal{B}_t$ are not reset by Line 9.A of*

Algorithm 5 is less than $\delta/6$, where we define

$$T_0 := C(\delta/2) + \widehat{N}^{\delta/2}(T_\delta) + \frac{32\alpha K(L_C+1)\ln T_\delta}{\Delta_{\min}^2} + 8K^2(L_C+1)^2 \ln \frac{6K^2}{\delta}.$$

Proof. By Line 9.A of Algorithm 5, as soon as we have $l_{ij} > 0.5$, the set $\mathcal{B}_t^i$ will be emptied. In what follows, we will show that the probability that the number of time-steps before we have $l_{ij} > 0.5$ is greater than

$$\Delta T := \widehat{N}^{\delta/2}(T_\delta) + N$$

with

$$N := \frac{32\alpha K(L_C+1)\ln T_\delta}{\Delta_{\min}^2} + 8K^2(L_C+1)^2 \ln \frac{6K^2}{\delta}$$

is bounded by $\delta/6K^2$. This is done using the amount of exploration infused by Line 10 of Algorithm 5. To begin, let us note that by Corollary 7.12, there is a time-step before $T_0 := C(\delta/2) + \widehat{N}^{\delta/2}(T_\delta)$ when the condition of Line 9.C of Algorithm 5 is satisfied for some Copeland winner. At this point, if $\mathcal{B}_t^i$ contains fewer than $L_C + 1$ elements, then it will be emptied; furthermore, for all $k > C$, the sets $B_{T_0}^k$ will have at most $L_C + 1$ elements and so the set

$$\mathcal{S}_t := \{(k,\ell) | a_\ell \in \mathcal{B}_t^k \text{ and } 0.5 \in [l_{k\ell}, u_{k\ell}]\}$$

contains at most $K(L_C + 1)$ elements for all $t \geq T_0$. Moreover, if at time-step $T_1 := C(\delta/2) + \Delta T$ we have $a_j \in \mathcal{B}_{T_1}^i$, then we can conclude that $(i,j) \in \mathcal{S}_t$ for all $t \in [C(\delta/2), T_1]$, since, if at any time after $C(\delta/2)$ arm a_j were to be removed from $\mathcal{B}_t^i$, it will never be added back because that can only happen through Line 9.B of Algorithm 5 and by Lemma 7.11 and the assumption of the lemma we have $u_{ij} > p_{ij} > 0.5$.

What we can conclude from the observations in the last paragraph is that if at time-step T_1 we still have $a_j \in \mathcal{B}_{T_1}^i$, then there are ΔT time-steps during which the probability of comparing arms a_i and a_j was at least $\frac{1}{4K(L_C+1)}$ and yet no more than $\frac{4\alpha \ln T_\delta}{\Delta_{ij}^2}$ comparisons took place, since otherwise, we would have $l_{ij} > 0.5$ at some point before T_1. Now, let B_n^{ij} denote the indicator random variable that is equal to 1 if arms a_i and a_j were chosen to be played against each other by Line 10 of Algorithm 5 during time-step $T_1 + n$. Also, let $X_1, \ldots, X_N$ be iid Bernoulli random variables with mean $\frac{1}{4K(L_C+1)}$. Since B_n^{ij} and X_n are Bernoulli and we have $\mathbb{E}\left[B_n^{ij}\right] \leq \mathbb{E}[X_n]$ for each n, then we can conclude that

$$P\left(\sum_{n=1}^{N} B_n^{ij} < s\right) \leq P\left(\sum_{n=1}^{N} X_n < s\right) \text{ for all } s.$$

On the other hand, we can use the Hoeffding bound to show that the right hand side of

the above inequality is smaller than $\delta/6$ if we set $s = \frac{4\alpha \ln T_\delta}{\Delta_{ij}^2}$:

$$\begin{aligned}
P\left(\sum_{n=1}^{N} X_n < \frac{4\alpha \ln T_\delta}{\Delta_{ij}^2}\right) &\leq P\left(\sum_{n=1}^{N} X_n < \frac{4\alpha \ln T_\delta}{\Delta_{\min}^2}\right) \\
&= P\left(\sum_{n=1}^{N} X_n < \frac{N}{4K(L_C+1)} - a\right) \leq e^{-\frac{2a^2}{N}} \\
&\qquad \text{with } a := -\frac{4\alpha \ln T_\delta}{\Delta_{\min}^2} + \frac{N}{4K(L_C+1)} \\
&= e^{-\frac{32\alpha^2 \ln^2 T_\delta}{\Delta_{\min}^4 N} + \frac{4\alpha \ln T_\delta}{K(L_C+1)\Delta_{\min}^2} - \frac{N}{8K^2(L_C+1)^2}} \\
&\leq e^{\frac{4\alpha \ln T_\delta}{K(L_C+1)\Delta_{\min}^2} - \frac{N}{8K^2(L_C+1)^2}} \\
&= e^{-\ln 6K^2/\delta} = \delta/6K^2.
\end{aligned}$$

Now, if we take a union bound over all pairs of arms a_i and a_j satisfying the condition stated at the beginning of this scenario, we get that with probability $\delta/6$ by time-step $C(\delta/2) + \Delta T$ all such erroneous hypotheses are reset by Line 9.A of Algorithm 5, emptying the sets $\mathcal{B}_t^i$. □

Lemma 7.14. *Let $t_1 \in [C(\delta/2), T_\delta)$ be such that for all i, j satisfying $a_j \in \mathcal{B}_{t_1}^i$ we have $p_{ij} < 0.5$. Then, the following two statements hold with probability $1 - 5\delta/6$:*

1. *If the set $\mathcal{B}_{t_1}$ in Algorithm 5 contains at least one Copeland winner, then if we set $t_2 = t_1 + n_{\max}$, where*

$$n_{\max} := 2K \max_{i>C} \widehat{N}_i^{\delta/2}(T_\delta) + \frac{K^2 \ln(6K/\delta)}{2},$$

   then $\mathcal{B}_{t_2}$ is non-empty and contains no non-Copeland winners, i.e. for all $a_i \in \mathcal{B}_{t_2}$ we have $i \leq C$.

2. *If the set $\mathcal{B}_{t_1}$ in Algorithm 5 contains no Copeland winners, i.e. for all $a_i \in \mathcal{B}_{t_1}$, we have $i > C$, then within $n_{\max}$ time-steps the set $\mathcal{B}_t$ will be emptied by Line 9.B of Algorithm 5.*

Therefore, with probability $1 - 5\delta/6$, by time $t_1 + 2n_{\max}$ all non-Copeland winners (i.e. arms a_i with $i > C$) are eliminated from $\mathcal{B}_t$.

Proof. We will consider the two cases in the following, conditioning on the conclusions of Lemma 7.11, Proposition 7.3 and Corollary 7.12, all simultaneously holding with $1 - \delta/2$:

1. $\mathcal{B}_{t_1}$ **contains** a Copeland winner (i.e. $a_c \in \mathcal{B}_{t_1}$ for some $c \leq C$): in this case, by Lemma 7.11, we know that the Copeland winner will forever remain in the set $\mathcal{B}_t$ because

$$\overline{\text{Cpld}}(a_c) \geq \max_j \text{Cpld}(a_j) \geq \max_j \underline{\text{Cpld}}(a_j),$$

then $\mathcal{B}_{t_2}$ will indeed be empty. Moreover, in what follows, we will show that the probability that any non-Copeland winner in $\mathcal{B}_t$ is not eliminated by time t_2 is less than $\delta/6$. Let us assume by way of contradiction that there exists an arm a_b with $b > C$ such that a_b is in $\mathcal{B}_{t_2}$: we will show that the probability of this happening is less than $\delta/6K$, and so, taking a union bound over non-Copeland winning arms, the probability that any non-Copeland winner is in $\mathcal{B}_{t_2}$ is seen to be smaller than $\delta/6$.

Now, to see that the probability of a_b being in the set $\mathcal{B}_{t_2}$ is small, note that the fact that a_b being in $\mathcal{B}_{t_2}$ implies that a_b was in the set $\mathcal{B}_t$ for the entirety of the time interval $[C(\delta/2), t_2]$ as we will show in the following. If a_b is eliminated from $\mathcal{B}_t$ at some point between t_1 and t_2, it will not get added back into $\mathcal{B}_t$ because that can only take place if the set $\mathcal{B}_t$ is reset at some point and there are only two ways for that to happen:

(a) By Line 9.A of Algorithm 5 in the case that for some pair (i, j) with $a_j \in \mathcal{B}_t^i$ we have $l_{ij} > 0.5$; however, this is ruled out by our assumption that at time t_1 we have $p_{ij} < 0.5$ and by Lemma 7.11, which stipulates that we have $l_{ij} \leq p_{ij} < 0.5$.

(b) By Line 9.B of Algorithm 5 in the case that all arms are eliminated from $\mathcal{B}_t$, but this cannot happen by the fact mentioned above that a_c will not not be removed from $\mathcal{B}_t$.

So, as mentioned above, we indeed have that at each time-step between t_1 and t_2, the set $\mathcal{B}_t$ contains a_b. Next, we will show that the probability of this happening is less than $\delta/6K$. To do so, let us denote by $\mathcal{S}_b$ the time-steps when arm a_b was in the set of optimistic Copeland winners, i.e.

$$\mathcal{S}_b := \left\{ t \in (t_1, t_2] \,\middle|\, a_b \in \mathcal{C}_t \right\}.$$

We can use Corollary 7.12 above with $T = T_\delta$ to show that the size of the set $\mathcal{S}_b$ (which we denote by $|\mathcal{S}_b|$) is bounded from below by $t_2 - t_1 - \sum_{i \neq j} \widehat{N}_{ij}^{\delta/2}(T_\delta)$: this is because whenever any Copeland winner a_c is played against itself, Equation (7.5) holds, and so if we were to have $a_b \notin \mathcal{C}_t$ during that time-step a_b would have had to get eliminated from $\mathcal{B}_t$ because a_b not being an optimistic Copeland winner would imply that

$$\overline{\text{Cpld}}(a_b) < \underline{\text{Cpld}}(a_c) = \overline{\text{Cpld}}(a_c).$$

But, we know from facts (a) and (b) above that a_b remains in $\mathcal{B}_t$ for all $t \in (t_1, t_2]$. Therefore, as claimed, we have

$$|\mathcal{S}_b| \geq t_2 - t_1 - \sum_{i \neq j} N_{ij}^{\delta/2}(T_\delta) \geq 2K \widehat{N}_b^{\delta/2}(T_\delta) + \frac{K^2 \ln(6K/\delta)}{2} =: n_b, \quad (7.6)$$

where the last inequality is due to the definition of $n_{\max} := t_2 - t_1$. On the other hand, Proposition 7.3 tells us that the number of time-steps between t_1 and t_2 when

a_b could have been chosen as an optimistic Copeland winner is bounded as

$$N_b^{\delta/2}(T_\delta) \leq \widehat{N}_b^{\delta/2}(T_\delta). \tag{7.7}$$

Furthermore, given the fact that during each time-step $t \in \mathcal{S}_b$ we have $a_b \in \mathcal{B}_t \cap \mathcal{C}_t$, the probability of a_b being chosen as an optimistic Copeland winner is at least $1/K$ because of the sampling procedure in Lines 14-17 of Algorithm 5. However, this is considerably higher than the ratio obtained by dividing the right-hand sides of Inequality (7.7) by that of Inequality (7.6). We will make this more precise in the following: for each $t \in \mathcal{S}_b$, denote by μ_t^b the probability that arm a_b would be chosen as the optimistic Copeland winner by Algorithm 5, and let X_t^b be the Bernoulli random variable that returns 1 when arm a_b is chosen as the optimistic Copeland winner and 0 otherwise. As pointed out above, we have that $\mu_t^b \geq \frac{1}{K}$ for all $t \in \mathcal{S}_b$, which, together with the fact that $|\mathcal{S}_b| \geq n_b$, implies that the random variable $X^b := \sum_{t \in \mathcal{S}_b} X_t^b$ satisfies

$$P(X_b < x) \leq P(Binom(n_b, 1/K) < x). \tag{7.8}$$

This is both because the Bernoulli summands of X_b have higher means than the Bernoulli summands of $Binom(n_b, 1/K)$ and because X_b is the sum of a larger number of Bernoulli variables, so X_b has more mass away from 0 than does $Binom(n_b, 1/K)$. So, we can bound the right-hand side of Inequality (7.8) by $\delta/6K$ with $x = \widehat{N}_b^{\delta/2}(T_\delta)$ to get our desired result. But, this is a simple consequence of the Hoeffding bound, a more general form of which is quoted in Section 7.4.3. More precisely, we have

$$\begin{aligned}
P\left(Binom(n_b, 1/K) < \widehat{N}_b^{\delta/2}(T_\delta)\right) &= P\left(Binom(n_b, 1/K) < \frac{n_b}{K} - a\right) \\
&\qquad \text{with } a := \frac{n_b}{K} - \widehat{N}_b^{\delta/2}(T_\delta) \\
&< e^{-2a^2/n_b} \\
&= e^{\frac{-2\left(\frac{n_b}{K} - \widehat{N}_b^{\delta/2}(T_\delta)\right)^2}{n_b}} \\
&= e^{-2n_b/K^2 + 4\widehat{N}_b^{\delta/2}(T_\delta)/K - 2\widehat{N}_b^{\delta/2}(T_\delta)^2/n_b} \\
&\leq e^{-2n_b/K^2 + 4\widehat{N}_b^{\delta/2}(T_\delta)/K} \\
&= e^{-\ln(6K/\delta)} = \delta/6K
\end{aligned}$$

Using the union bound over the non-Copeland winning arms that were in $\mathcal{B}_{t_1}$, of whom there is at most $K-1$, we can conclude that with probability $\delta/6$ they are all eliminated from $\mathcal{B}_{t_2}$.

2. $\mathcal{B}_{t_1}$ **does not contain** any Copeland winners: in this case, we can use the exact same argument as above to conclude that the probability that the set $\mathcal{B}_t$ is non-empty for all $t \in (t_1, t_2]$ is less than $\delta/6$ because as before the probability that each arm $a_b \in \mathcal{B}_{t_1}$ is not eliminated within n_b time-steps is smaller than $\delta/6K$. □

Let us now state the following consequence of the previous lemmas:

Lemma 7.6. *Given $\delta > 0$, the following fact holds with probability $1 - \delta$: for each $i > C$, the set $\mathcal{B}^i_{T_\delta}$ contains exactly $L_C + 1$ elements with each element a_j satisfying $p_{ij} < 0.5$. Moreover, for all $t \in [T_\delta, T]$, we have $\mathcal{B}^i_t = \mathcal{B}^i_{T_\delta}$.*

Proof. In the remainder of the proof, we will condition on the high probability event that the conclusions of Lemma 7.11, Corollary 7.12, Lemma 7.13 and Lemma 7.14 all hold simultaneously with probability $1 - \delta$.

Combining Lemma 7.14, we can conclude that by time-step $T_1 := T_0 + 2n_{\max}$ all non-Copeland winners are removed from $\mathcal{B}_{T_1}$, which also means by Line 9.B of Algorithm 5 that the corresponding sets $\mathcal{B}^i_{T_1}$, with $i > C$ are non-empty, and Lemma 7.13 tells us that these sets have at least $L_C + 1$ elements a_j each of which beats a_i (i.e. $p_{ij} < 0.5$).

Now, applying Corollary 7.12, we know that within $\widehat{N}^{\delta/2}(T_\delta)$ time-steps, Line 9.C of Algorithm 5 will be executed, at which point we will have $\overline{L}_C = L_C$ and so $\mathcal{B}^i_t$ will be reduced to $L_C + 1$ elements. Moreover, by Lemma 7.11, for all $t > T_1$ and $a_j \in \mathcal{B}^i_t$ we have $l_{ij} \leq p_{ij} < 0.5$ and so $\mathcal{B}^i_t$ will not be emptied by any of the provisions in Line 9 of Algorithm 5.

Now, since by definition we have $T^\delta \geq T_1 + \widehat{N}^{\delta/2}(T_\delta)$, we have the desired result. □

7.4.6 Proof of Lemma 7.7

Lemma 7.7 *Given a Copeland bandit problem satisfying Assumption* **A** *and any $\delta > 0$, with probability $1-\delta$ the following statement holds: the number of time-steps between $T_{\delta/2}$ and T when each non-Copeland winning arm a_i can be chosen as optimistic Copeland winners (i.e. time-steps when arm a_c in Algorithm 5 satisfies $c = i > C$) is bounded by*

$$\widehat{N}^i := 2\widehat{N}^i_{\mathcal{B}} + 2\sqrt{\widehat{N}^i_{\mathcal{B}} \ln \frac{2K}{\delta}},$$

where

$$\widehat{N}^i_{\mathcal{B}} := \sum_{j \in \mathcal{B}^i_{T_{\delta/2}}} \widehat{N}^{\delta/4}_{ij}(T).$$

Proof. The idea of the argument is outlined in the following sequence of facts:

1. By Lemma 7.6, we know that with probability $1 - \delta/2$, for each $i > C$ and all times $t > T_{\delta/2}$ the sets $\mathcal{B}^i_t$ will consist of exactly $L_C + 1$ arms that beat the arm a_i, and that $\mathcal{B}^i_t = \mathcal{B}^i_{T_{\delta/2}}$.

2. Moreover, if at time $t > T_{\delta/2} > C(\delta/4)$, Algorithm 5 chooses a non-Copeland winner as an optimistic Copeland winner (i.e. $i > C$), then with probability $1 - \delta/4$ we know that

$$\overline{\text{Cpld}}(a_i) \geq \overline{\text{Cpld}}(a_1) \geq \text{Cpld}(a_1) = K - 1 - L_C.$$

3. This means that there could be at most L_C arms a_j that optimistically lose to a_i (i.e. $u_{ij} < 0.5$) and so at least one arm $a_b \in \mathcal{B}^i_t$ does satisfy $u_{ib} \geq 0.5$

4. This, in turn, means that in Line 13 of Algorithm 5 with probability 0.5 the arm a_d will be chosen from $\mathcal{B}^i_t$.

5. By Proposition 7.3, we know that with probability $1 - \delta/4$, in the time interval $[T_{\delta/2}, T]$ each arm $a_j \in \mathcal{B}^i_{T_{\delta/2}}$ can be compared against a_i at most $\widehat{N}^{\delta/4}_{ij}(T)$ many times.

Given that by Fact 3 above we need at least one arm $a_j \in \mathcal{B}^i_t$ to satisfy $u_{ij} \geq 0.5$ for Algorithm 5 to set $(c, d) = (i, j)$, and that by Fact 4 arms from $\mathcal{B}^i_t$ have a higher probability of being chosen to be compared against a_i, this means that arm a_i will be chosen as optimistic Copeland winner roughly twice as many times we had $(c, d) = (i, j)$ for some $j \in \mathcal{B}^i_{T_{\delta/2}}$. A high probability version of the claim in the last sentence together with Fact 5 would give us the bound on regret claimed by the theorem. In the remainder of this proof, we will show that indeed the number of times we have $c = i$ is unlikely to be too many times higher than twice the number of times we get $(c, d) = (i, j)$, where $j \in \mathcal{B}^i_{T_{\delta/2}}$. To do so, we will introduce the following notation:

N^i**:** the number of time-steps between $T_{\delta/2}$ and T when arm a_i was chosen as optimistic Copeland winner.

B^i_n**:** the indicator random variable that is equal to 1 if Line 13 in Algorithm 5 decided to choose arm a_d only from the set $B^i_{t_n}$ and zero otherwise, where t_n is the n^{th} time-step after $T_{\delta/2}$ when arm a_i was chosen as optimistic Copeland winner. Note that B^i is simply a Bernoulli random variable with mean 0.5.

$N^i_{\mathcal{B}}$**:** the number of time-steps between T_δ and T when arm a_i was chosen as optimistic Copeland winner and that Line 13 in Algorithm 5 chose to pick an arm from $\mathcal{B}^i_{T_{\delta/2}}$ to be played against a_i. Note that this definition implies that we have

$$N^i_{\mathcal{B}} = \sum_{n=1}^{N^i} B^i_n. \tag{7.9}$$

Moreover, by Fact 5 above, we know that with probability $1 - \delta/4$ we have

$$N^i_{\mathcal{B}} \leq \widehat{N}^i_{\mathcal{B}} := \sum_{j \in \mathcal{B}^i_{T_{\delta/2}}} \widehat{N}^{\delta/4}_{ij}(T). \tag{7.10}$$

Now, we will use the above high probability bound on $N^i_{\mathcal{B}}$ to put the following high probability bound on N^i: with probability $1 - \delta/2$ we have

$$N^i \leq \widehat{N}^i := 2\widehat{N}^i_{\mathcal{B}} + 2\sqrt{\widehat{N}^i_{\mathcal{B}}} \ln \frac{2K}{\delta}.$$

To do so, let us assume that the we have $N^i > \widehat{N}^i$ and consider the first $\widehat{N}^i$ time-steps after $T_{\delta/2}$ when arm a_i was chosen as optimistic Copeland winner and note that by Equation (7.9) we have

$$\sum_{n=1}^{\widehat{N}^i} B^i_n \leq N^i_{\mathcal{B}}$$

and so by Inequality (7.10) with probability $1-\delta/4$ the left-hand side of the last inequality is bounded by $\widehat{N}^i_{\mathcal{B}}$: let us denote this event with $\mathcal{E}$. On the other hand, if we apply the Hoeffding bound (cf. §7.4.3) to the variables $B^i_1, \ldots, B^i_{\widehat{N}^i}$, we get

$$
\begin{aligned}
P\left(\mathcal{E} \wedge N^i > \widehat{N}^i\right) &\leq P\left(\sum_{n=1}^{\widehat{N}^i} B^i_n < \widehat{N}^i_{\mathcal{B}}\right) \\
&= P\left(\sum_{n=1}^{\widehat{N}^i} B^i_n < \widehat{N}^i/2 - \sqrt{\widehat{N}^i_{\mathcal{B}} \ln \frac{2K}{\delta}}\right) \\
&\leq e^{-\frac{2\widehat{N}^i_{\mathcal{B}} \left(\ln \frac{2K}{\delta}\right)^2}{2\widehat{N}^i_{\mathcal{B}} + 2\sqrt{\widehat{N}^i_{\mathcal{B}} \ln \frac{2K}{\delta}}}} \qquad \text{(by Chernoff-Hoeffding)} \qquad (7.11)
\end{aligned}
$$

To simplify the last expression in the last chain of inequalities, let us use the notation $\alpha := \widehat{N}^i_{\mathcal{B}}$ and $\beta := \ln \frac{2K}{\delta}$. Given this notation, we claim that the following inequality holds if we have $\alpha \geq 4$ and $\beta \geq 2$ (which hold by the assumptions of the theorem):

$$
\frac{\alpha\beta^2}{\alpha + \sqrt{\alpha}\beta} \geq \beta. \qquad (7.12)
$$

To see this, let us multiply both sides by the denominator of the left-hand side of the above inequality:

$$
\alpha\beta^2 \geq \alpha\beta + \sqrt{\alpha}\beta. \qquad (7.13)
$$

To see why Inequality (7.13) holds, let us note that the restrictions imposed on α and β imply the following pair of inequalities, whose sum is equivalent to Inequality (7.13):

$$
\begin{array}{rrcl}
 & \alpha\beta^2 & \geq & 2\alpha\beta \\
+ & \alpha\beta^2 & \geq & 2\sqrt{\alpha}\beta^2 \\
\hline
= & 2\alpha\beta^2 & \geq & 2\alpha\beta + 2\sqrt{\alpha}\beta^2
\end{array}
$$

Now that we know that Inequality (7.12) holds, we can combine it with Inequality (7.11) to get

$$
P\left(\mathcal{E} \wedge N^i > \widehat{N}^i\right) \leq e^{-\ln \frac{2K}{\delta}} = \frac{\delta}{2K}.
$$

Taking a union over the non-Copeland winning arms, we get

$$
P(\mathcal{E} \wedge \forall i > C,\, N^i > \widehat{N}^i) > 1 - \delta/2.
$$

So, given the fact that we have $P(\mathcal{E}) < \delta/4$, we know that with probability $1-\delta$ each non-Copeland winner is selected as optimistic Copeland winner between $T_{\delta/2}$ and T no more than $\widehat{N}^i$ times. □

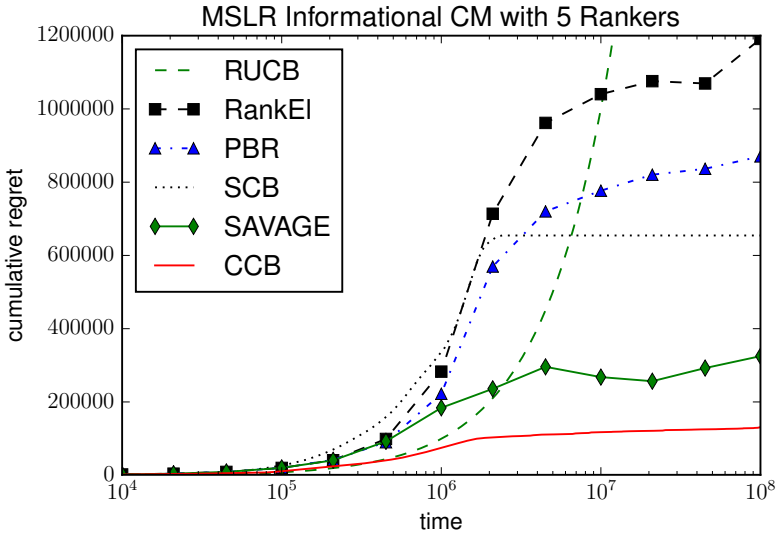

Figure 7.6: Small-scale regret results for a 5-armed Copeland dueling bandit problem arising from ranker evaluation.

7.5 Experiments

To evaluate CCB, we apply it to a Copeland dueling bandit problem arising from *ranker evaluation*.

We follow the experimental approach in [73, 78] and use a preference matrix to simulate comparisons between each pair of arms (a_i, a_j) by drawing samples from Bernoulli random variables with mean p_{ij}. We compare CCB against five K-armed dueling bandit algorithms, RUCB [78], Copeland SAVAGE [65], Preference-Based Racing (PBR) [15] and Rank Elicitation (RankEl) [16] and Scalable Copeland Bandits (SCB) [79]. We include RUCB in order to verify our claim that K-armed dueling bandit algorithms that assume the existence of a Condorcet winner have linear regret if applied to a Copeland dueling bandit problem without a Condorcet winner.

More specifically, we consider a 5-armed dueling bandit problem obtained from comparing five rankers, none of whom beat the other four, i.e. there is no Condorcet winner.[3] Figure 7.6 shows the regret accumulated by CCB, SCB, the Copeland variants of SAVAGE, PBR, RankEl and RUCB on this problem. The horizontal time axis uses a log scale, while the vertical axis, which measures cumulative regret, uses a linear scale. CCB outperforms all other algorithms in this 5-armed experiment.

Note that three of the baseline algorithms under consideration here (i.e., SAVAGE, PBR and RankEl) require the horizon of the experiment as an input, either directly or through a failure probability δ, which we set to $1/T$ (with T being the horizon), in order to obtain a finite-horizon regret algorithm, as prescribed in [65, 73]. Therefore, we ran

[3] Sample code and the preference matrices used in the experiments can be found at http://bit.ly/nips15data.

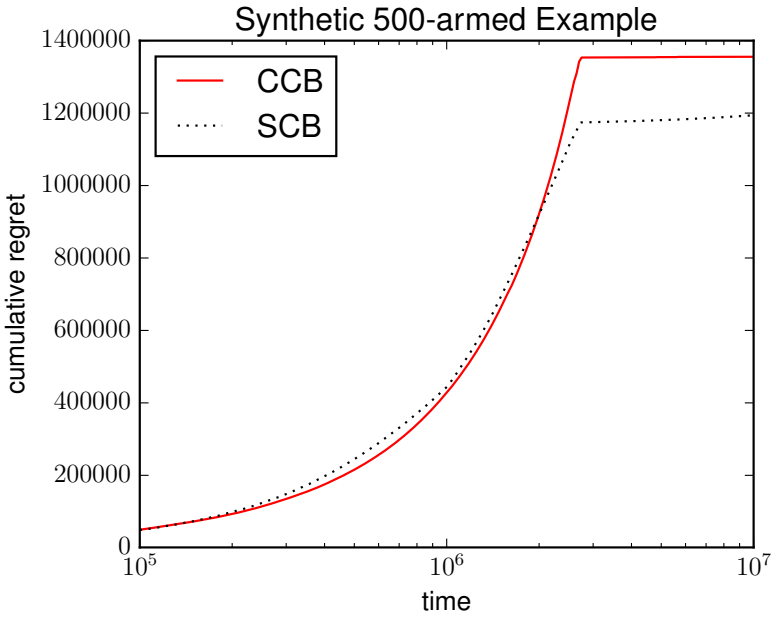

Figure 7.7: Large-scale regret results for a synthetic 500-armed Copeland dueling bandit problem.

independent experiments with varying horizons and recorded the accumulated regret: the markers on the curves corresponding to these algorithms represent these numbers. Consequently, the regret curves are not monotonically increasing. For instance, SAVAGE's cumulative regret at time 2×10^7 is lower than at time 10^7 because the runs that produced the former number were not continuations of those that resulted in the latter, but rather completely independent. Furthermore, RUCB's cumulative regret grows linearly, which is why the plot does not contain the entire curve.

Additionally, we carry out a more detailed investigation of our proposed algorithm. In particular, we conduct both a scalability experiment to understand the behaviours of CCB as the number of arms grows as well as an experiment on a dueling bandit problem that satisfies the Condorcet assumption.

Our scalability experiment uses a 500-armed synthetic example created to test the scalability of CCB in comparison to SCB. In particular, we fix a preference matrix in which the three Copeland winners are in a cycle, each with a Copeland score of 498, and the other arms have Copeland scores ranging from 0 to 496.

Figure 7.7, which depicts the results of this experiment, shows that when there are many arms, SCB can outperform CCB. We omit SAVAGE, PBR and RankEl from this experiment because they scale poorly in the number of arms [15, 16, 65].

The reason for the sharp transition in the regret curves of CCB and SCB in the synthetic experiment is as follows. Because there are many arms, as long as one of the two arms

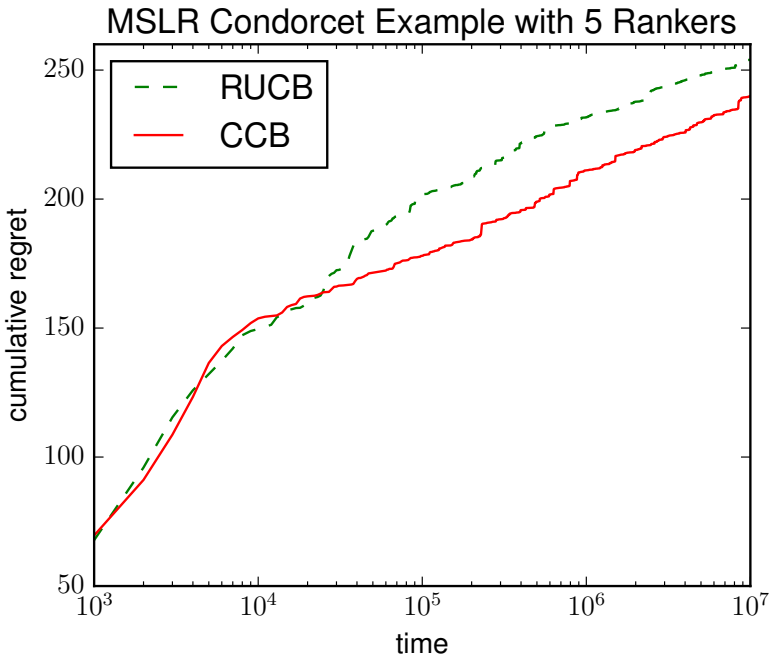

Figure 7.8: Regret results for a Condorcet example.

being compared is not a Copeland winner, the comparison can result in substantial regret; since both algorithms choose the second arm in each round based on some criterion other than the Copeland score, even if the first chosen arm in a given time-step is a Copeland winner, the incurred regret may be as high as 0.5. The sudden transition in Figure 7.7 occurs when the algorithm becomes confident enough of its choice for the first arm to begin comparing it against itself, at which point it stops accumulating regret.

As advertised previously, our next experiment is on an example with a Condorcet winner in order to show how CCB compares against RUCB when the condition required by RUCB is satisfied. The regret plots for the remaining algorithms were excluded here since they both perform substantially worse than either RUCB or CCB, as expected. This example was extracted in the same fashion as the example used in the above ranker evaluation experiment, with the sole difference that this time we ensured that one of the rankers is a Condorcet winner. The results, depicted in Figure 7.8, show that CCB enjoys a slight advantage over RUCB in this case. We attribute this to the careful process of identifying and utilizing the weaknesses of non-Copeland winners, as carried out by lines 12 and 18 of Algorithm 5.

7.6 Summary

In the dueling bandit problem, the goal is to use pairwise feedback to find the most desirable choice from a set of options. Most existing work in this area assumes the

existence of a Condorcet winner, i.e., an arm that beats all other arms with probability greater than 0.5. Even though these results have the advantage that the bounds they provide scale linearly in the number of arms, their main drawback is that in practice the Condorcet assumption is too restrictive. By contrast, other results that do not impose the Condorcet assumption achieve bounds that scale quadratically in the number of arms.

In this chapter, we set out to solve a natural generalization of the problem, where instead of assuming the existence of a Condorcet winner, we seek to find a Copeland winner, which is guaranteed to exist. We proposed an algorithm, called CCB, as a solution for this problem. We provided theoretical results bounding the regret accumulated by our algorithm: these results improve substantially over existing results in the literature, by filling the gap that exists in the current results, namely the discrepancy between results that make the Condorcet assumption and are of the form $\mathcal{O}(K \log T)$ and the more general results that are of the form $\mathcal{O}(K^2 \log T)$.

Moreover, we have evaluated the effectiveness of CCB using examples from online ranker evaluation, and as our results indicate, CCB performs very well for small numbers of arms. An interesting question raised by our scalability experiments is whether or not it is possible to devise an algorithm that has the benefits of both CCB and SCB, i.e., the scalability of the latter together with the former's better dependence on the gaps: as discussed in §7.4.2, the gap that governs the performance of CCB is substantially larger than $\Delta_{\min}$, on which the regret bound for SCB depends.

8 Conclusions

In this thesis, we set out to address the K-armed dueling bandit problem, which is designed to address situations in which we are presented with a number of possibilities (called "arms" for historical reasons) that can be adopted and we are interested in the "best" option, but we can only obtain information about the relative qualities of the various choices through noisy comparisons between pairs of arms. Such a situation arises, for instance, in the context of online ranker evaluation, where each arm corresponds to a ranker that we would like to deploy for instance as part of a search engine and where the comparisons can be carried out through interleaved comparisons.

In the following sections, we provide a summary of the results presented in this thesis and outline some directions for further research.

8.1 Summary of Results

Let us revisit the research questions that were promised to be addressed in Chapter 1:

RQ1 Can the lower bound for dueling bandits be met without the total ordering assumption?

This question was addressed in Chapter 4 with the introduction of the RUCB algorithm, which greatly advanced the state of the art by being the first algorithm with regret bound of the form $\mathcal{O}(K^2 + K \log T)$ under very general assumptions. Furthermore, the experimental results included in Chapter 4 attest to the practicality of RUCB. In particular, in the case of the examples used in Chapter 4, the regret accumulated by RUCB is less than half of that of Condorcet SAVAGE, which was the state of the art for small-scale problems.

RQ2 Can we devise an algorithm that is more exploitative than RUCB?

The answer to this question was provided through the introduction of the RCS algorithm in Chapter 5 and demonstrating experimentally that it does outperform RUCB when the number of arms is small. In particular, the experiments comparing RUCB and RCS were carried out on examples whose numbers of arms ranged from 10 to 40 in increments of 10. In the case of the 10-armed example, RCS dominated the RUCB, but in the case of the examples with larger numbers of arms, we saw that even though RCS outperforms RUCB asymptotically, this does not hold for all time-steps.

RQ3 Can the quadratic dependence on the number of arms in the regret bound for RUCB be eliminated?

We dealt with this research question in Chapter 6 with the introduction and theoretical analysis of the mergeRUCB algorithm. We established a regret bound for mergeRUCB that takes the form $\mathcal{O}(K \log T)$ with no quadratic dependence on K, unlike the bound for RUCB, whose additive constant is $\mathcal{O}(K^2)$. It was also shown experimentally that for large-scale problems mergeRUCB outperforms RUCB significantly. In particular, we carried out comparisons between mergeRUCB, RUCB and Beat the Mean (BTM), the art algorithm for large-scale problems. These comparisons were carried out using the feature rankers of three learning to rank datasets with the numbers of feature rankers being 136, 245 and 700. The experiments demonstrated that as the number of rankers grows, the advantage obtained from using mergeRUCB, rather than RUCB and BTM, becomes more and more pronounced.

RQ4 Can the lower bound for the dueling bandit problem be met under practically general assumptions?

This research question was addressed in Chapter 7, where we introduced the CCB algorithm, whose theoretical guarantee matches the lower bound proven in [74], which takes the form $\mathcal{O}(K \log T)$ asymptotically in T, and importantly, the assumptions under which this result holds do not hinder its applicability in practice. More specifically, the cumulative regret of CCB was shown to take the form $\mathcal{O}(K^2 + K \log T)$.

From a practical point of view, the most significant accomplishment of the above results is to provide the first broadly applicable dueling bandit algorithm in the literature, i.e. CCB. Furthermore, under the more restrictive assumption that the dueling bandit problem possesses a Condorcet winner, mergeRUCB provides a practical solution for large-scale problems.

8.2 Future Work

As with any body of work in research, the results presented in this thesis raise more questions than they answer. Indeed, there are some limitations to the work on which we have reported in the thesis. For instance, an algorithm that possesses the advantages of both CCB and mergeRUCB continues to remain elusive: this is discussed below.

In the following, we discuss some directions for further inquiry:

Thompson sampling. It is a meta-theorem (or perhaps a form of superstition) in the field of bandits that whenever there is a bandit algorithm that uses confidence intervals to obtain theoretical guarantees, there should be a similar algorithm which instead of confidence bounds uses samples from a posterior distribution, which we call the "Thompson" version of the algorithm. Moreover, this Thompson version should work better than the confidence bound version, although its theoretical analysis is considerably more difficult [2, 44]. In the case of RUCB, it is very easy to envisage what this Thomson version should be; indeed, RCS is a partial step in that direction, where half of RUCB has been "Thompsonized," for

lack of a better word. However, even this partial "Thompsonization" proved too difficult to deal with theoretically using the techniques that exist in the literature at this time.

The main difference between the dueling setting and the K-armed bandit problem is that, in the case of a latter, if a suboptimal arm has a posterior that is not very well concentrated, then we can rest easy because that means that the arm has not been pulled many times. However, in the case of dueling bandits, a suboptimal arm might have been compared against some arms many times and not so much against others. In other words, just because the posterior for p_{ij} for some suboptimal arm a_i is not concentrated, it does not imply that the regret due to a_i is small. Naturally, this is a very interesting problem that merits further investigation.

Scalability. A major flaw of the mergeRUCB algorithm is its crucial reliance on the existence of the Condorcet winner. So, a natural question is if this issue can be addressed. A partial answer to this question is provided by the Scalable Copeland Bandit (SCB) algorithm proposed in [79], where it was shown that the regret bound for SCB has no quadratic dependence on K. However, as experimental results show, this algorithm does not provide a very practical solution to this problem, and the question remains whether the scalability of SCB can be combined with the practicality of CCB.

Extensions. As with the multi-armed bandit problem, one can formulate and investigate various extensions of the dueling bandit problem, e.g., adversarial bandits, where an adversay chooses the preference matrix, or contextual bandits, where the environment provides some side information that could be used by the algorithm to decide which arms to choose for comparison, etc. Even though there exist partial solutions to these problems in the literature [24, 28], the study of such modifications of the dueling bandit problem remains largely unexplored. Indeed, a very interesting and challenging open problem is to design a computationally efficient contextual dueling bandit problem with $\mathcal{O}(\sqrt{T})$ regret bound: the best known result so far is $\mathcal{O}(T^{2/3})$ [24].

Bibliography

[1] Y. Abbasi-yadkori, D. Pal, and C. Szepesvari. Improved algorithms for linear stochastic bandits. In *NIPS*, 2011. (Cited on page 27.)

[2] S. Agrawal and N. Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *Conference on Learning Theory*, pages 1–26, 2012. (Cited on pages 41, 43, and 98.)

[3] N. Ailon, Z. Karnin, and T. Joachims. Reducing dueling bandits to cardinal bandits. In *ICML*, 2014. (Cited on pages 4, 17, and 18.)

[4] K. Amin, M. Kearns, and U. Syed. Bandits, query learning, and the haystack dimension. In *COLT*, 2011. (Cited on page 57.)

[5] A. Antos, G. Bartók, D. Pál, and C. Szepesvári. Toward a classification of finite partial-monitoring games. *Theoretical Computer Science*, 2012. (Cited on page 20.)

[6] J.-Y. Audibert, R. Munos, and C. Szepesvári. Exploration-exploitation tradeoff using variance estimates in multi-armed bandits. *Theor. Comput. Sci.*, 410(19):1876–1902, 2009. (Cited on page 27.)

[7] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Symposium on Foundations of Computer Science*, pages 322–331, 1995. (Cited on page 18.)

[8] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002. (Cited on page 23.)

[9] A. Balsubramani, Z. Karnin, R. Schapire, and M. Zoghi. Instance-dependent regret bounds for dueling bandits. In *Conference on Learning Theory*, 2016. (Cited on page 8.)

[10] G. Bartók, N. Zolghadr, and C. Szepesvári. An adaptive algorithm for finite stochastic partial monitoring. In *ICML*, 2012. (Cited on page 20.)

[11] E. Brochu, T. Brochu, and N. de Freitas. A Bayesian interactive optimization approach to procedural animation design. In *ACM SIGGRAPH*, 2010. (Cited on pages 2 and 52.)

[12] S. Bubeck and N. Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 5:1–122, 2012. (Cited on page 18.)

[13] S. Bubeck, R. Munos, G. Stoltz, and C. Szepesvari. X-armed bandits. *Journal of Machine Learning Research*, 12:1655–1695, 2011. (Cited on pages 41 and 52.)

[14] R. Busa-Fekete and E. Hüllermeier. A survey of preference-based online learning with bandit algorithms. In *Algorithmic Learning Theory*, pages 18–39. Springer, 2014. (Cited on pages 13 and 68.)

[15] R. Busa-Fekete, B. Szörényi, P. Weng, W. Cheng, and E. Hüllermeier. Top-k selection based on adaptive sampling of noisy preferences. In *ICML*, 2013. (Cited on pages 20, 93, and 94.)

[16] R. Busa-Fekete, B. Szörényi, and E. Hüllermeier. PAC rank elicitation through adaptive sampling of stochastic pairwise preferences. In *National Conference on Artificial Intelligence (AAAI)*, 2014. (Cited on pages 20, 93, and 94.)

[17] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006. (Cited on page 79.)

[18] O. Chapelle and Y. Chang. Yahoo! learning to rank challenge overview. *Journal of Machine Learning Research-Proceedings Track*, 14:1–24, 2011. (Cited on page 21.)

[19] O. Chapelle and L. Li. An empirical evaluation of thompson sampling. In *NIPS*, 2011. (Cited on page 43.)

[20] O. Chapelle, T. Joachims, F. Radlinski, and Y. Yue. Large-scale validation and analysis of interleaved search evaluation. *ACM Trans. Inf. Syst.*, 30(1):6:1–6:41, 2012. (Cited on page 2.)

[21] A. Chuklin, I. Markov, and M. de Rijke. *Click Models for Web Search*. Morgan & Claypool, 2015. (Cited on page 22.)

[22] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *WSDM '08*, pages 87–94, 2008. (Cited on pages 22 and 40.)

[23] N. de Freitas, A. Smola, and M. Zoghi. Exponential regret bounds for Gaussian process bandits with deterministic observations. In *ICML*, 2012. (Cited on pages 41 and 52.)

[24] M. Dudík, K. Hofmann, R. E. Schapire, A. Slivkins, and M. Zoghi. Contextual dueling bandits. In *Conference on Learning Theory*, 2015. (Cited on pages 3, 8, 18, 20, 68, 69, and 99.)

[25] J. Elster and A. Hylland. *Foundations of Social Choice Theory*. Cambridge University Press, 1989. (Cited on page 3.)

[26] W. Feller. *An Introduction to Probability Theory and Its Applications*, volume 1. Wiley, 1968. (Cited on page 36.)

[27] S. Fox, K. Karnawat, M. Mydland, S. Dumais, and T. White. Evaluating implicit measures to improve web search. *TOIS*, 23(2):147–168, 2005. (Cited on page 2.)

[28] P. Gajane, T. Urvoy, and F. Clérot. A relative exponential weighing algorithm for adversarial utility-based dueling bandits. In *ICML*, 2015. (Cited on page 99.)

[29] M. Gardner. Mathematical games: The paradox of the nontransitive dice and the elusive principle of indifference. *Scientific American*, 223:110–114, 1970. (Cited on pages 3 and 68.)

[30] F. Guo, L. Li, and C. Faloutsos. Tailoring click models to user goals. In *WSCD '09*, pages 88–92, 2009. (Cited on page 22.)

[31] F. Guo, C. Liu, and Y. Wang. Efficient multiple-click models in web search. In *WSDM '09*, pages 124–131, New York, NY, USA, 2009. ACM. (Cited on pages 22 and 40.)

[32] D. K. Harman. Overview of the second text retrieval conference (trec-2). In *NIST Special Publication. Presented at the Second Text Retrieval Conference (TREC 2)*. Department of Commerce, National Institute of Standards and Technology, 1993. (Cited on page 1.)

[33] J. He, C. Zhai, and X. Li. Evaluation of methods for relative comparison of retrieval systems based on clickthroughs. In *CIKM '09*, pages 2029–2032, 2009. (Cited on page 22.)

[34] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963. (Cited on page 11.)

[35] K. Hofmann. *Fast and reliable online learning to rank for information retrieval*. PhD thesis, University of Amsterdam, Netherlands, 2013. (Cited on page 21.)

[36] K. Hofmann, S. Whiteson, and M. de Rijke. A probabilistic method for inferring preferences from clicks. In *CIKM '11*, pages 249–258, USA, 2011. ACM. (Cited on pages 21, 22, and 40.)

[37] K. Hofmann, S. Whiteson, and M. de Rijke. Fidelity, soundness, and efficiency of interleaved comparison methods. *ACM Transactions on Information Systems*, 31(4), 2013. (Cited on pages 22, 27, 61, and 64.)

[38] K. Hofmann, S. Whiteson, and M. de Rijke. Balancing exploration and exploitation in listwise and pairwise online learning to rank for information retrieval. *Information Retrieval*, 16(1):63–90, 2013. (Cited on pages 22 and 40.)

[39] K. Hofmann, L. Li, and F. Radlinski. Online evaluation. *Manuscript*, 2016. (Cited on page 1.)

[40] K. Jamieson, R. Nowak, and B. Recht. Query complexity of derivative-free optimization. In *NIPS*, 2012. (Cited on page 41.)

[41] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, October 2002. (Cited on page 1.)

[42] T. Joachims. Optimizing search engines using clickthrough data. In *KDD*, 2002. (Cited on page 22.)

[43] T. Joachims. Evaluating retrieval performance using clickthrough data. In J. Franke, G. Nakhaeizadeh, and I. Renz, editors, *Text Mining*, pages 79–96. Springer, Berlin, Germany, 2003. (Cited on pages 2 and 22.)

[44] E. Kauffmann, N. Korda, and R. Munos. Thompson sampling: an asymptotically optimal finite time analysis. In *International Conference on Algorithmic Learning Theory*, 2012. (Cited on pages 41, 43, and 98.)

[45] D. Kelly. Methods for evaluating interactive information retrieval systems with users. *Foundations and Trends in Information Retrieval*, 3:1–224, 2009. (Cited on page 1.)

[46] R. Kohavi, R. Longbotham, D. Sommerfield, and R. M. Henne. Controlled experiments on the web: survey and practical guide. *Data Mining and Knowledge Discovery*, 18(1), 2008. URL `http://www.springerlink.com/index/10.1007/s10618-008-0114-1`. (Cited on page 2.)

[47] R. Kohavi, A. Deng, B. Frasca, T. Walker, Y. Xu, and N. Pohlmann. Online controlled experiments at large scale. In *KDD*, 2013. (Cited on page 2.)

[48] R. Kohavi, A. Deng, B. Frasca, T. Walker, Y. Xu, and N. Pohlmann. Online controlled experiments at large scale. In *KDD '13*, pages 1168–1176. ACM, 2013. (Cited on page 53.)

[49] J. Komiyama, J. Honda, H. Kashima, and H. Nakagawa. Regret lower bound and optimal algorithm in dueling bandit problem. In *Conference on Learning Theory*, 2015. (Cited on page 19.)

[50] T.-Y. Liu, J. Xu, T. Qin, W. Xiong, and H. Li. Letor: Benchmark dataset for research on learning to rank for information retrieval. In *LR4IR '07, in conjunction with SIGIR '07*, 2007. (Cited on page 40.)

[51] C. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. (Cited on pages 1 and 38.)

[52] Microsoft Learning to Rank Datasets, 2012. `http://research.microsoft.com/en-us/projects/mslr/default.aspx`. (Cited on page 21.)

[53] R. Munos. Optimistic optimization of a deterministic function without the knowledge of its smoothness. In *NIPS*, 2011. (Cited on pages 41 and 52.)

[54] S. Negahban, S. Oh, and D. Shah. Iterative ranking from pair-wise comparisons. In *NIPS*, 2012. (Cited on page 20.)

[55] G. Owen. *Game Theory*. Emerald Group Publishing Limited, 3rd edition, 1995. (Cited on page 18.)

[56] A. Piccolboni and C. Schindelhauer. Discrete prediction games with arbitrary feedback and loss. In *Computational Learning Theory*, pages 208–223, 2001. (Cited on page 20.)

[57] F. Radlinski, M. Kurup, and T. Joachims. How does clickthrough data reflect retrieval quality? In *CIKM*, 2008. (Cited on pages 21, 22, and 38.)

[58] R. L. Rivest and E. Shen. An optimal single-winner preferential voting system based on game theory. In V. Conitzer and J. Rothe, editors, *Proceedings Third International Workshop on Computational Social Choice*. Düsseldorf University Press, 2010. (Cited on pages 67 and 68.)

[59] H. Robbins. Some Aspects of the Sequential Design of Experiments. *Bulletin of the American Mathematical Society*, 58:527–535, 1952. (Cited on page 2.)

[60] M. Sanderson. Test collection based evaluation of information retrieval systems. *Foundations and Trends in Information Retrieval*, 4:247–375, 2010. (Cited on page 1.)

[61] M. Schulze. A new monotonic, clone-independent, reversal symmetric, and Condorcet-consistent single-winner election method. *Social Choice and Welfare*, 36(2):267–303, 2011. (Cited on pages 11, 13, and 67.)

[62] A. Schuth, K. Hofmann, and F. Radlinski. Predicting search satisfaction metrics with interleaved comparisons. In *SIGIR 2015*. ACM, 2015. (Cited on page 2.)

[63] N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *ICML*, 2010. (Cited on pages 27, 41, and 52.)

[64] W. R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, pages 285–294, 1933. (Cited on pages 9, 41, and 43.)

[65] T. Urvoy, F. Clerot, R. Féraud, and S. Naamane. Generic exploration and k-armed voting bandits. In *ICML*, 2013. (Cited on pages 4, 12, 13, 16, 20, 27, 46, 63, 93, and 94.)

[66] M. Valko, A. Carpentier, and R. Munos. Stochastic simultaneous optimistic optimization. In *ICML*, 2013. (Cited on pages 41 and 52.)

[67] Z. Wang, M. Zoghi, F. Hutter, D. Matheson, and N. De Freitas. Bayesian optimization in high dimensions via random embeddings. In *IJCAI*, 2013. (Cited on page 8.)

[68] L. Wasserman. *All of Statistics: A Concise Course in Statistical Inference*. Springer Publishing Company, Incorporated, 2010. (Cited on page 10.)

[69] H. Wu, X. Liu, and R. Srikant. Double thompson sampling for dueling bandits. In *NIPS*, 2016. (Cited on page 41.)

[70] Yandex Internet Mathematics 2009 Dataset, 2009. `http://imat2009.yandex.ru/en/datasets`. (Cited on page 21.)

[71] Y. Yue and T. Joachims. Interactively optimizing information retrieval systems as a dueling bandits problem. In *ICML*, 2009. (Cited on pages 14 and 41.)

[72] Y. Yue and T. Joachims. Interactively optimizing information retrieval systems as a dueling bandits problem. In *ICML '09*, pages 1201–1208, New York, NY, USA, 2009. ACM. (Cited on page 52.)

[73] Y. Yue and T. Joachims. Beat the mean bandit. In *ICML*, 2011. (Cited on pages 4, 14, 15, 27, 40, 61, 63, and 93.)

[74] Y. Yue, J. Broder, R. Kleinberg, and T. Joachims. The k-armed dueling bandits problem. In *Conference on Learning Theory (COLT)*, 2009. (Cited on pages 3 and 98.)

[75] Y. Yue, J. Broder, R. Kleinberg, and T. Joachims. The K-armed dueling bandits problem. *Journal of Computer and System Sciences*, 78(5):1538–1556, Sept. 2012. (Cited on pages 9, 11, 14, 15, 27, 61, and 66.)

[76] J.-Y. Zhu, A. Agarwala, A. A. Efros, E. Shechtman, and J. Wang. Mirror mirror: Crowdsourcing better portraits. *ACM Transactions on Graphics (SIGGRAPH Asia 2014)*, 33(6), 2014. (Cited on page 2.)

[77] M. Zoghi, S. Whiteson, M. de Rijke, and R. Munos. Relative confidence sampling for efficient on-line ranker evaluation. In *WSDM*, 2014. (Cited on pages 7, 41, and 61.)

[78] M. Zoghi, S. Whiteson, R. Munos, and M. de Rijke. Relative upper confidence bound for the k-armed dueling bandits problem. In *ICML*, 2014. (Cited on pages 3, 7, 19, 23, 48, 49, 61, 66, and 93.)

[79] M. Zoghi, Z. Karnin, S. Whiteson, and M. de Rijke. Copeland dueling bandits. In *NIPS*, 2015. (Cited on pages 5, 8, 19, 93, and 99.)

[80] M. Zoghi, S. Whiteson, and M. de Rijke. MergeRUCB: A method for large-scale online ranker evaluation. In *WSDM*, 2015. (Cited on page 8.)

[81] M. Zoghi, T. Tunys, L. Li, D. Jose, J. Chen, C. M. Chin, and M. de Rijke. Click-based hot fixes for underperforming torso queries. In *SIGIR*, 2016. (Cited on page 8.)

Summary:

In every domain where a service or a product is provided, an important question is that of evaluation: given a set of possible choices for deployment, what is the best one? An important example, which is considered in this work, is that of ranker evaluation from the field of information retrieval (IR). The goal of IR is to satisfy the information need of a user in response to a query issued by them, where this information need is typically satisfied by a document (or a small set of documents) contained in what is often a much larger collection. This goal is often attained by ranking the documents according to their usefulness to the issued query using an algorithm, called a ranker, a procedure that takes as input a query and a set of documents and specifies how the documents need to be ordered.

This thesis is concerned with ranker evaluation. The goal of ranker evaluation is to determine the quality of the rankers under consideration to allow us to choose the best option: given a finite set of possible rankers, which one of them leads to the highest level of user satisfaction? There are two main methods for carrying this out: absolute metrics and relative comparisons. This thesis is concerned with the second, relative form of ranker evaluation because it is more efficient at distinguishing between rankers of different quality: for instance interleaved comparisons take a fraction of the time required by A/B testing, but they produce the same outcome. More precisely, the problem of online ranker evaluation from relative feedback can be described as follows: given a finite set of rankers, choose the best using only pairwise comparisons between the rankers under consideration, while minimizing the number of comparisons involving sub-optimal rankers. This problem is an instance of what is referred to as the dueling bandit problem in the literature.

The main contribution of this thesis is devising a dueling bandit algorithm, called Copeland Confidence Bounds (CCB), that solves this problem under practically general assumptions and providing theoretical guarantees for its proper functioning. In addition to that, the thesis contains a number of other algorithms that are better suited for dueling bandit problems with particular properties.

Samenvatting:

In elk domein waar een dienst of een product wordt geleverd is een belangrijke vraag die van de evaluatie: gegeven een verzameling van mogelijke keuzes, wat is de beste? Een belangrijk voorbeeld, dat wordt beschouwd in dit werk, is die van rankerevaluatie binnen het vakgebied van de information retrieval (IR). Het doel van IR is te voldoen aan de informatiebehoefte van een gebruiker in reactie op een zoekvraag, waarbij de informatie gewoonlijk wordt voldaan door middel van een document (of een kleine verzameling documenten) te selecteren uit een veel grotere verzameling van documenten. Dit doel wordt vaak bereikt door het rangschikken van de documenten op basis van hun nut voor de gegeven zoekvraag met behulp van een algoritme, een zogenaamde ranker, een procedure die als input een query neemt en als output een geordende reeks documenten oplevert.

Dit proefschrift houdt zich bezig met rankerevaluatie. Het doel van rankerevaluatie is om de kwaliteit van de rankers in kwestie vast te stellen om zo de beste ranker te kiezen: gegeven een eindige verzameling van mogelijke rankers, welke leidt tot de grootste tevredenheid van de gebruikers? Er zijn twee belangrijke methoden voor de uitvoering: absolute evaluatiematen en relatieve vergelijkingen. Dit proefschrift betreft de tweede, relatieve, vorm van rankerevaluatie omdat het efficinter onderscheid maakt tussen rankers van verschillende kwaliteit. Bijvoorbeeld het om-en-om presenteren van resultaten van rankers kost een fractie van de tijd van een A/B test, maar leidt tot dezelfde uitkomst. Meer in het bijzonder, kan het probleem van de online ranker evaluatie met behulp van relatieve feedback als volgt worden omschreven: gegeven een eindige verzameling van rankers, kies de beste met behulp van slechts paarsgewijze vergelijkingen tussen de rankers in kwestie, waarbij een minimaal aantal vergelijkingen van suboptimale rankers gebruikt wordt. Dit probleem is een voorbeeld van wat wordt aangeduid als het duellerend bandietprobleem (dualing bandit problem) in de literatuur.

De belangrijkste bijdrage van dit proefschrift is het opstellen van een duellerend bandiet- algoritme, genaamd Copeland Confidence Bounds (CCB), die dit probleem oplost onder nagenoeg algemene aannames en het verstrekken van theoretische garanties voor de goede werking ervan. Het proefschrift bevat bovendien een aantal andere algoritmen die beter geschikt zijn voor duellerende bandietproblemen met bijzondere eigenschappen.