Adaptive Sampling for Coarse Ranking

Sumeet Katariya
katariya@wisc.edu

Lalit Jain
lalitj@umich.edu

Nandana Sengupta
nandana@uchicago.edu

James Evans
jevans@uchicago.edu

Robert Nowak
rdnowak@wisc.edu

Abstract

We consider the problem of active *coarse ranking*, where the goal is to sort items according to their means into clusters of pre-specified sizes, by adaptively sampling from their reward distributions. This setting is useful in many social science applications involving human raters and the *approximate* rank of *every* item is desired. Approximate or coarse ranking can significantly reduce the number of ratings required in comparison to the number needed to find an exact ranking. We propose a computationally efficient PAC algorithm `LUCBRank` for coarse ranking, and derive an upper bound on its sample complexity. We also derive a nearly matching distribution-dependent lower bound. Experiments on synthetic as well as real-world data show that `LUCBRank` performs better than state-of-the-art baseline methods, even when these methods have the advantage of knowing the underlying parametric model.

1 Introduction

We consider the problem of efficiently sorting items according to their means into clusters of pre-specified sizes, which we refer to as *coarse ranking*. In many big-data applications, finding the *total* ranking can be infeasible and/or unnecessary, and we may only be interested in the top items, bottom items, or quantiles. Consider for instance the problem of assessing the safety of neighborhoods from pairwise comparisons of Google street view images, as is done in the Place Pulse project (Naik et al., 2014), which can be applied to develop social policy (Dubey et al., 2016). Finding a complete ordering of the images in this case is impractical because many images are difficult to compare i.e., their safety scores are very close (see Section 7.2). Furthermore, a total ordering may be unnecessary from a public policy point of view, since the *approximate* rank of every image on the safe-unsafe spectrum may suffice.

Motivated by these applications, we model the coarse ranking problem as follows. Given K random variables, $c \geq 2$ clusters, and cluster boundaries $1 \leq \kappa_1 < \kappa_2 < \cdots < \kappa_{c-1} < \kappa_c = K$, the goal is to reliably identify the κ_1 random variables with the highest means, the $\kappa_2 - \kappa_1$ random variables with the highest means among the remaining $K - \kappa_1$ random variables, and so on, by observing samples from their reward distributions (for a precise formulation see Section 4). The focus of this paper is on algorithms that achieve this clustering by requesting samples adaptively. The coarse ranking setting applies to the scenarios above, and also subsumes many well-studied problems. The problem of finding the best item corresponds to $\kappa_1 = 1, \kappa_2 = K$. The problem of finding the top-m items corresponds to $\kappa_1 = m, \kappa_2 = K$. The problem of sorting the items into c equal-sized clusters corresponds to $\kappa_i = \text{round}(iK/c), 1 \leq i \leq c$. Finally, the complete ranking can be obtained by setting $\kappa_i = i, 1 \leq i \leq i \leq K$.

The problem of *completely sorting* items is in general hard in real-world applications, and does not exhibit gains from adaptivity. Maystre and Grossglauser (2017) who analyze the performance of Quicksort, observe in their real-world experiments:

> "The improvement is noticeable but modest. We notice that item parameters are close to each other on average; ... This is because there is a considerable fraction of items that have their parameters (means) very close to one another ... Figuring out the exact order of these images is therefore difficult and probably of marginal value." (1)

The fact that *adaptivity doesn't help for complete ranking* is true not just for Quicksort, but other adap-

Proceedings of the 21^{st} International Conference on Artificial Intelligence and Statistics (AISTATS) 2018, Lanzarote, Spain. PMLR: Volume 84. Copyright 2018 by the author(s).

tive algorithms as well - as we observe in our experiments. Adaptivity does however help for coarse ranking, and this can be explained. Consider the case when the K items have bounded reward distributions, and their means are equally separated, with a gap Δ between consecutive means. Correctly ordering any two consecutive items requires $\Omega(1/\Delta^2)$ samples, and thus *any* algorithm would require $\Omega(K/\Delta^2)$ to find a total ordering. A non-adaptive algorithm sampling the items uniformly would gather approximately equal samples from every item, and hence will find the correct ranking after roughly these many samples (up to perhaps log factors). Thus adaptivity doesn't help in this case. However, if the goal is to find only the quartiles say, an adaptive algorithm can quickly stop sampling items that are far from the quartile boundaries and gain over non-adaptive algorithms.

In this work, we make six contributions. First, we motivate the coarse ranking setting. We do this by arguing that most real-life problems have high noise, and by explaining why adaptive methods are ineffective in producing a complete ranking in these high-noise regimes (Section 3). Second, we precisely formulate the online probably approximately correct (PAC)-coarse ranking problem with error tolerance ϵ and failure probability δ that can model real-valued as well as pairwise comparison feedback (Section 4). Third, we propose a nonparametric PAC Upper Confidence Bound (UCB)-type algorithm `LUCBRank` to solve this problem. To the best of our knowledge, this is the first UCB-type algorithm for ranking (Section 5). Fourth, we analyze the sample complexity of `LUCBRank` and prove an upper bound which is inversely proportional to the distance of the item to its closest cluster boundary, where the distance is measured in terms of Chernoff information (Section 6). Fifth, we also prove a nearly matching distribution-dependent lower bound. The contribution of an item to the lower bound is inversely proportional to the distance of the item to the closest item in an adjacent cluster, with distance in this case measured using KL-divergences (Section 6.3). Finally, we compare the performance of our algorithm to several baselines on synthetic as well as real-world data gathered using MTurk, and observe that it performs 2 - 3x better than existing algorithms even when they have the advantage of knowing the underlying parametric model (Section 7).

1.1 Ranking using Pairwise Comparisons

We use the term direct-feedback or real-rewards to indicate a setting where the learner can sample directly from the item's reward distribution. Our algorithm is stated for this setting. In contrast, in the pairwise-comparison or dueling setting, the learner compares two items and receives 1-bit feedback about who won the duel. We next explain how to translate our algorithm to this setting.

Any algorithm designed to solve the direct-feedback coarse ranking problem can also be used with pairwise comparison feedback using Borda reduction (Jamieson et al., 2015b). According to this technique, whenever the algorithm asks to draw a sample from item i, we compare item i to a randomly chosen item j, and ascribe a reward of 1 to item i if i wins the duel, and 0 otherwise. This is equivalent to the rewards being sampled from a Bernoulli distribution with means given by the Borda scores of the items. The Borda score of an item i is defined as

$$p_i := \frac{1}{K-1} \sum_{j \neq i} \mathbb{P}(i > j). \tag{2}$$

2 Related Work

There is extensive work on ranking from noisy pairwise comparisons, we refer the reader to excellent surveys by Busa-Fekete and Hüllermeier (2014); Agarwal (2016). We discuss the most relevant work next.

2.1 Ranking from Pairwise Comparisons

The pairwise comparison matrix P (where $P_{ij} = \mathbb{P}(i > j)$) and assumptions on it play a major role in the design of ranking algorithms (Agarwal, 2016). A sequence of progressively relaxed assumptions on P can be shown where ranking methods that work under restrictive assumptions fail when these assumptions are relaxed (Rajkumar and Agarwal, 2014; Rajkumar et al., 2015). Spectral ranking algorithms have been proposed when comparisons are available for a fixed set of pairs (Negahban et al., 2012a,b); this corresponds to a partially observed P matrix. Braverman and Mossel (2009); Wauthier et al. (2013) propose and analyze algorithms for the noisy-permutation model; this corresponds to a P matrix which has two types of entries: $1-p$ in the upper triangle and p in the lower triangle (assuming the true ordering of the items is $1 \ldots K$). They also focus on settings where queries cannot be repeated. Our work makes no assumptions on the P matrix and ranks items using their Borda scores. This is important given the futility of parametric models to model real-life scenarios (Shah et al., 2016).

Quicksort is another highly recommended algorithm for ranking using noisy pairwise comparisons. Maystre and Grossglauser (2017) study Quicksort under the BTL noise model, and Alonso et al. (2003) analyze Quicksort under the noisy permutation model. We comment on these in Section 3.

Jamieson and Nowak (2011) propose an algorithm for active ranking from pairwise comparisons when points can be embedded in Euclidean space. Ailon (2012) consider ranking when query responses are fixed. More recently, Agarwal et al. (2017) consider top-m item identification and ranking under limited rounds of adaptivity, Falahatgar et al. (2017) consider the problem of finding the maximum and ranking assuming strong-stochastic transitivity and the stochastic-triangle inequality. We do not need these assumptions.

Our setting is closest to the setting proposed by Heckel et al. (2016), in the context of ranking using pairwise comparisons. Our setting however applies to real-valued rewards as well as pairwise comparison feedback. Furthermore, our setting incorporates the notion of ϵ-optimality which allows the user to specify an error tolerance (Even-Dar et al., 2006). This is important in practice if the item means are very close to each other. Finally, as they note, their Active Ranking (AR) algorithm is an elimination-style algorithm, our `LUCBRank` is UCB-style; it is known that the latter perform better in practice (Jiang et al., 2017). We also verify this empirically in Section 7.2, and observe that `LUCBRank` requires 2-3x fewer samples than AR in our synthetic as well as real-world experiments (see Fig. 2 and Fig. 5).

2.2 Relation to Bandits

The idea of sampling items based on lower and upper confidence bounds is well-known in the bandits literature (Auer, 2002). However, these algorithms either focus on finding the best or top-m items (Audibert and Bubeck, 2010; Kalyanakrishnan et al., 2012; Kaufmann et al., 2015; Chen et al., 2017), or on minimizing regret (Bubeck et al., 2012). This is the first work to our knowledge that employs this tool for ranking.

3 Motivation

We argue that existing adaptive methods offer no significant gains over their non-adaptive counterparts when the goal is to find a complete ranking, and coarse ranking is more appropriate for many real-world applications. We provide brief theoretical justification for this claim in the discussion after quote (1), and empirically verify this behavior in Fig. 4. In this section, we focus on Quicksort, because it has been well-studied under multiple noise models. Quicksort has optimal sample complexity when comparisons are noiseless (Sedgewick and Wayne, 2011) and is naturally appealing when comparisons are noisy (Maystre and Grossglauser, 2017). Intuitively it feels like the right thing to do - by comparing an item with the pivot and putting it left or right appropriately, Quicksort performs a binary search for the true position of an item. However it is far from optimal under two noise models as we argue next.

First, consider the noisy-permutation (NP) noise model (Feige et al., 1994) where the outcomes of pairwise comparisons are independently flipped with an error probability p. In the first stage of Quicksort, every item that is compared with the pivot and put in the wrong bucket contributes on average $\frac{K}{2}$ to the Kendall tau error (total number of inverted pairs). Now, Kp items are put in the wrong bucket on average in the first stage of Quicksort, and hence the total number of inverted pairs is at least $\Omega(K^2p)$. Alonso et al. (2003) show that $\Theta(K^2p)$ is indeed the expected number of inversions. This is far from optimal because Braverman and Mossel (2009) propose an algorithm which has a Kendall tau error of $O(K)$ *with high probability*, using $K \log K$ comparisons (same as quicksort). Alonso et al. (2003) conjecture that for quicksort to have $O(K)$ expected inversions, p needs to go down faster than $1/K$, like $\frac{1}{K \log K}$. As the above calculation shows, they conjecture that this is because Quicksort is extremely brittle: "the main contribution (to the total inversions) comes from the 'first' error, in some sense." One may be able to get rid of this lack of robustness by repeating queries, but this requires knowledge of the error probability p or adapting to its unknown value. This is possible, but as we argue shortly, a good model for real-world problems where comparisons are made by humans is one where p increases to $1/2$ as K grows, since it becomes more difficult to compare adjacent items in the true ranking as K increases. Quicksort certainly fails in this regime.

The other class of well-studied noise models are the Bradley-Terry-Luce (BTL) (Bradley and Terry, 1952) or Thurstone (Thurstone, 1927) models, which assume a K-dimensional weight vector that measures the quality of each item, and the pairwise comparison probabilities are determined via some fixed function of the qualities of pair of objects. These models are more realistic than the NP model since under these models, comparisons between items that are far apart in the true ranking are less noisy than those between nearby items. Maystre and Grossglauser (2017) analyze the expected number of inversions of Quicksort under the BTL model, and show that when the average gap between adjacent items is Δ, the expected number of inversions is $O(\Delta^{-3})$. They note however that real-world datasets have extremely small Δ ($\hat{\Delta}^{-1} = 376$ in their experiments) and Quicksort performs no better than random (see quote (1)). We make similar observations about the inefficacy of Quicksort (and other adaptive algorithms) in our real-world experiments (see Fig. 4).

The problem in finding an exact/total ranking is that if the means of the items lie in a bounded range, e.g., $[0,1]$, then the minimum gap must decrease at least linearly with K and many items become essentially indistiguishable. To see this, suppose there is a constant gap Δ between consecutive means and let $m = \lceil \frac{3}{\Delta} \rceil$. Then, assuming the logistic model, the m-th item beats the 1^{st} item with probability ≥ 0.95, the $2m$-th item beats the m-th item with probability ≥ 0.95, and so on. Thus, items that are m-apart can be considered distinguishable. Assuming the range of possible means is bounded implies that $\Delta \propto 1/K$. Thus, the number of items that are essentially indistiguishable increases linearly with K, suggesting that seeking a total ranking is a futile effort. This situation arises in applications such as Place Pulse where humans rate street view images according to their perceived safety (Naik et al., 2014), or the task in Wood et al. (2017) where humans rate face images according to the strength of their emotions.

Coarse ranking allows the experimenter to set the number of clusters in accordance with the number of distinguishable levels, and thus frees the algorithm from the task of distinguishing incomparable items. In this sense, it converts a high-noise problem to a low-noise one. Even though the gap between adjacent items is small, most items are far from their nearest cluster boundary, and an adaptive algorithm can stop sampling these items early.

4 Setting

In this section, we precisely formulate the coarse ranking setting. For ease of reference, we use terminology from the bandits literature and refer to an item as arm. Also, pulling or drawing an arm is equivalent to sampling from the item's reward distribution.

Consider a multi-armed bandit with K arms. Each arm a corresponds to a Bernoulli distribution with an unknown mean p_a, denoted $\mathcal{B}(p_a)$. A draw / pull of arm a yields a reward from distribution $\mathcal{B}(p_a)$. Without loss of generality, assume the arms are numbered so that $p_1 \geq p_2 \cdots \geq p_K$.

Given an integer $c \geq 2$ representing the number of clusters, let $1 \leq \kappa_1 < \kappa_2 < \cdots < \kappa_c = K$ be a collection of positive integers. Any such collection of positive integers defines a partition of $[K]$ into c disjoint sets of the form

$$M_1^* := \{1, \ldots, \kappa_1\},\ M_2^* := \{\kappa_1 + 1, \ldots, \kappa_2\}, \ldots, \\ \ldots,\ M_c^* := \{\kappa_{c-1} + 1, \ldots, K\}. \tag{3}$$

To solve the *coarse ranking* problem given a set of cluster boundaries $(\kappa_i)_{i=1}^c$, an algorithm may sample arms of the K-armed bandit and record the results; the algorithm is required to terminate and cluster the arms into an ordered set of disjoint sets of the form (3). We refer to this output as a *coarse ranking.*

We next define the notion of ϵ-tolerance. For some fixed tolerance $\epsilon \in [0,1]$ and $1 \leq i \leq c$, let $M_{i,\epsilon}^*$ be the set of all arms that should be in cluster i upto a tolerance ϵ, i.e.

$$M_{i,\epsilon}^* := \{a : p_{\kappa_{i-1}+1} + \epsilon \geq p_a \geq p_{\kappa_i} - \epsilon\},$$

(with the convention that $p_{\kappa_0} = 1$). Note that the true set of arms in cluster i: $M_i^* := \{\kappa_{i-1} + 1, \ldots, \kappa_i\}$, is a subset of $M_{i,\epsilon}^*$; the latter set contains in addition arms that are ϵ close to the boundary.

For a given mistake probability $\delta \in [0,1]$ and a given error tolerance $\epsilon \in [0,1]$, we call an algorithm (ϵ, δ)-PAC if, with a probability greater that $1-\delta$, after using a finite number of samples, it returns a rank for each arm such that the i^{th} ranked cluster according to the returned ranking is a subset of $M_{i,\epsilon}^*$ for all $1 \leq i \leq c$. Formally, if $\sigma(a)$ is the rank of arm a returned by the algorithm after using a finite number of samples, we can define the empirical cluster i as

$$\hat{M}_i := \{a : \kappa_{i-1} + 1 \leq \sigma(a) \leq \kappa_i\},$$

and we say the algorithm is (ϵ, δ)-PAC if

$$\mathbb{P}\left(\exists\, i \text{ such that } \hat{M}_i \not\subseteq M_{i,\epsilon}^*\right) \leq \delta. \tag{4}$$

5 Algorithm

Let $(\kappa_1, \ldots, \kappa_c = K)$ be the cluster boundaries. We describe here the `LUCBRank` algorithm using generic confidence intervals $\mathcal{I}_a = [L_a(t), U_a(t)]$, where t indexes rounds of the algorithm. Let $N_a(t)$ be the number of times arm a has been sampled up to round t, and $S_a(t)$ be the sum of rewards of arm a up to round t. Let $\hat{p}_a(t) = \frac{S_a(t)}{N_a(t)}$ be the corresponding empirical mean reward. Sort the arms in the decreasing order of their empirical mean rewards, and for $1 \leq i \leq c-1$, let $J_i(t)$ denote the κ_i arms with the highest empirical mean rewards. Define

$$l_t^i := \operatorname*{arg\,min}_{a \in J_i(t)} L_a(t), \qquad u_t^i := \operatorname*{arg\,max}_{a \notin J_i(t)} U_a(t) \tag{5}$$

to be the two *critical* arms from $J_i(t)$ and $J_i^c(t)$ that are likely to be misclassified (see Fig. 1).

Algorithm 1 contains the pseudocode of `LUCBRank`, which is also depicted in Fig. 1. The algorithm maintains active cluster boundaries in the set C, where a cluster boundary i is active if the overlap of confidence intervals in J_i and J_i^c is not less than ϵ. In every round, it samples both the critical arms at every active cluster boundary (lines 11-15). At the end of every round, it checks if the critical arms at any boundary are separated according to the tolerance criterion, and removes such boundaries from the active

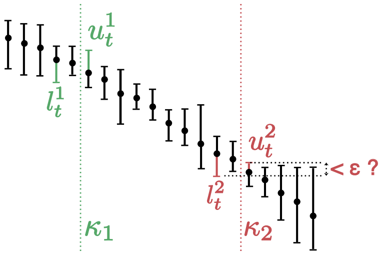


Figure 1: A visualization of LUCBRank on a bandit instance with $K = 20$ arms, $c = 3$ clusters, with boundaries at $\kappa_1 = 5, \kappa_2 = 15$. Also shown are the critical arms l_t^i, u_t^i pulled at each boundary. The algorithm stops sampling a boundary when the confidence interval overlap is less than ϵ.

set (lines 21-25). For our experiments, we use KL-UCB (Garivier and Cappé, 2011) confidence intervals. For an exploration rate $\beta(t, \delta)$, the KL-UCB upper and lower confidence bounds for arm a are calculated as

$$\begin{aligned} U_a(t) &:= \max\{q \in [\hat{p}_a(t), 1] : N_a(t) d(\hat{p}_a(t), q) \leq \beta(t, \delta)\}, \\ L_a(t) &:= \min\{q \in [0, \hat{p}_a(t)] : N_a(t) d(\hat{p}_a(t), q) \leq \beta(t, \delta)\}. \end{aligned} \tag{6}$$

where $d(x, y)$ is the Kullback-Leibler divergence between two Bernoulli distributions, given by $d(x, y) = x \log \frac{x}{y} + (1 - x) \log \frac{1-x}{1-y}$.

LUCBRank can also be easily modified for pairwise-comparison queries: whenever the algorithm calls for drawing an arm i, duel arm i with another arm chosen uniformly at random.

6 Analysis

We prove the accuracy of LUCBRank in Theorem 1, and give an upper bound on the sample complexity in Theorem 2. Our distribution-dependent lower bound for the sample complexity of any δ-PAC algorithm is stated in Theorem 3. All proofs can be found in the Appendix. Recall that $1 \leq \kappa_1 < \kappa_2 < \cdots < \kappa_{c-1} < \kappa_c = K$ are the cluster boundaries.

6.1 PAC Guarantee

Theorem 1 gives choices of $\beta(t, \delta)$ such that LUCBRank is correct with probability at least δ, in the sense defined by (4).

Theorem 1. LUCBRank *using* $\beta(t, \delta) = \log(\frac{k_1 K t^\alpha}{\delta}) + \log\log(\frac{k_1 K t^\alpha}{\delta})$ *with* $\alpha > 1$ *and* $k_1 > \left(\frac{c-1}{2}\right)^\alpha + \frac{2e}{\alpha - 1} + \frac{4e}{(\alpha-1)^2}$, *is correct with probability* $1 - \delta$.

6.2 Sample Complexity

Our sample complexity results are stated in terms of Chernoff information (Cover and Thomas, 2012).

Chernoff Information: Consider two Bernoulli distributions $\mathcal{B}(x)$ and $\mathcal{B}(y)$, and let $d(x, y)$ denote the KL-divergence between these distributions. The Chernoff information $d^*(x, y)$ between these two Bernoulli distributions is defined by

Algorithm 1 LUCBRank

```
1:  Input: $\epsilon > 0$, cluster boundaries $1 \leq \kappa_1, \ldots, \kappa_c = K$
2:  $t \leftarrow 1$
3:  $C \leftarrow \{1, \ldots, c-1\}$    //active cluster boundaries
4:
5:  for $a = 1, \ldots, K$ do
6:      Sample item $a$, compute $U_a(1)$ and $L_a(1)$
7:  end for
8:
9:  while $C \neq \varnothing$ do
10:     // Sample active cluster boundaries
11:     for $i \in C$ do
12:         Sample item $l_t^i$
13:         Sample item $u_t^i$
14:         (If pairwise comparing, compare item $l_t^i$ to a random other item, and compare item $u_t^i$ to a random other item. See Section 1.1)
15:     end for
16:     $t = t + 1$
17:     $\forall a \in [K]$ : Update reward-estimate $\hat{p}_a(t)$, number of samples $N_a(t)$, and confidence bounds $U_a(t), L_a(t)$ (see (6))
18:     $\forall i \in C$: Compute $l_t^i, u_t^i$ (see (5))
19:
20:     // Eliminate unambiguous cluster boundaries
21:     for $i \in C$ do
22:         if $U_{u_t^i}(t) - L_{l_t^i}(t) < \epsilon$ then
23:             $C = C \setminus i$
24:         end if
25:     end for
26: end while
27:
28: Return items sorted by their empirical mean rewards.
```

$$d^*(x, y) := d(z^*, x) = d(z^*, y)$$

where z^* is the unique z such that $d(z, x) = d(z, y)$.

Next we introduce some notation. For an arm a, let $g(a)$ (read group of arm a) denote the index of the cluster that arm a belongs to. Formally,

$$g(a) := \min\{1 \leq i \leq c : p_a \leq p_{\kappa_i}\}. \tag{7}$$

Let $b_i \in [p_{\kappa_i}, p_{\kappa_i + 1}], 1 \leq i \leq c - 1$ be any points in the cluster boundary gaps, and $b := (b_1, b_2, \ldots, b_{c-1})$. Define

$$\Delta_b^*(a) := \begin{cases} d^*(p_a, b_1) & a \in \{1, \ldots, \kappa_1\} \\ \min(d^*(p_a, b_{g(a)-1}), d^*(p_a, b_{g(a)}) & \\ & a \in \{\kappa_1 + 1, \ldots, \kappa_{c-1}\} \\ d^*(p_a, b_{c-1}) & a \in \{\kappa_{c-1} + 1, \ldots, K\} \end{cases} \tag{8}$$

to be the "distance" of each arm from the closest cluster boundary. Our upper bound on the sample complexity of `LUCBRank` is stated in Theorem 2, and contains the quantity $H^*_{\epsilon,b}$ where

$$H^*_{\epsilon,b} := \sum_{a \in \{1,\dots,K\}} \frac{1}{\max(\Delta^*_b(a), \epsilon^2/2)}. \tag{9}$$

Theorem 2. *Let $b = (b_1, b_2, \dots, b_{c-1})$, where $b_i \in [p_{\kappa_i}, p_{\kappa_i+1}]$. Let $\epsilon > 0$. Let $\beta(t,\delta) = \log(\frac{k_1 K t^\alpha}{\delta}) + \log\log(\frac{k_1 K t^\alpha}{\delta})$ with $k_1 > \left(\frac{c-1}{2}\right)^\alpha + \frac{2e}{\alpha-1} + \frac{4e}{(\alpha-1)^2}$. Let τ be the random number of samples taken by* `LUCBRank` *before termination. If $\alpha > 1$,*

$$\mathbb{P}\left(\tau \le 2C_0(\alpha) H^*_{\epsilon,b} \log\left(\frac{k_1 K (2H^*_{\epsilon,b})^\alpha}{\delta}\right)\right) \ge 1 - \delta$$

where $C_0(\alpha)$ is such that $C_0(\alpha) \ge \left(1 + \frac{1}{e}\right)\left(\alpha \log(C_0(\alpha)) + 1 + \frac{\alpha}{e}\right)$.

6.3 Distribution-Dependent Lower Bound

In this section, we state our non-asymptotic lower bound on the expected number of samples needed by any δ-PAC algorithm to cluster and rank the arms into groups of sizes $(\kappa_1, \kappa_2 - \kappa_1, \dots, K - \kappa_{c-1})$. For simplicity, we focus on the case $\epsilon = 0$. The proof of the lower bound uses standard change of measure arguments (Kaufmann et al., 2015), which requires some continuity and well-separation assumptions. We state these next.

We consider the following class of bandit models where the clusters are unambiguously separated, i.e.

$$\mathcal{M}_\kappa = \{p = (p_1, \dots, p_K) : p_i \in \mathcal{P}, p_{\kappa_i} > p_{\kappa_i+1}, 1 \le i < c\}, \tag{10}$$

where $\mathcal{P}$ is a set that satisfies

$$\forall\, p, q \in \mathcal{P}^2, p \ne q \Rightarrow 0 < KL(p,q) < +\infty.$$

We also assume the following:

Assumption 1. *For all $p, q \in \mathcal{P}^2$ such that $p \ne q$, for all $\alpha > 0$,*

there exists $q_1 \in \mathcal{P}$: $KL(p,q) < KL(p,q_1) < KL(p,q) + \alpha$ and $\mathbb{E}_{X\sim q_1}[X] > \mathbb{E}_{X\sim q}[X]$,

there exists $q_2 \in \mathcal{P}$: $KL(p,q) < KL(p,q_2) < KL(p,q) + \alpha$ and $\mathbb{E}_{X\sim q_2}[X] < \mathbb{E}_{X\sim q}[X]$.

To state our lower bound, we need to define for each arm a, another "distance" from the boundary, similar to (8). Define

$$\Delta^{\mathrm{KL}}_\kappa(a) := \begin{cases} \mathrm{KL}(p_a, p_{\kappa_1+1}) & a \in \{1,\dots,\kappa_1\} \\ \min(\mathrm{KL}(p_a, p_{\kappa_{g(a)-1}}), \mathrm{KL}(p_a, p_{\kappa_{g(a)}+1}) & \\ & a \in \{\kappa_1+1, \dots, \kappa_{c-1}\} \\ \mathrm{KL}(p_a, p_{\kappa_{c-1}}) & a \in \{\kappa_{c-1}+1, \dots, K\}, \end{cases} \tag{11}$$

where $g(a)$ defined in (7) is the cluster that arm a belongs to. We highlight the differences from (8). First, the Chernoff information in (8) is replaced with KL-divergence in (11), and second, the distance is measured with the closest arm in either adjacent cluster here, as opposed to a point in the gap between the clusters in (8).

Our lower bound involves the quantity

$$\sum_{a \in 1,\dots,K} \frac{1}{\Delta^{\mathrm{KL}}_\kappa(a)} \tag{12}$$

and is as follows:

Theorem 3. *Let $p \in \mathcal{M}_\kappa$, and assume that $\mathcal{P}$ satisfies Assumption 1; any coarse ranking algorithm that is δ-PAC on $\mathcal{M}_\kappa$ satisfies, for $\delta \le 0.15$,*

$$\mathbb{E}_p[\tau] \ge \left[\sum_{a \in 1,\dots,K} \frac{1}{\Delta^{KL}_\kappa(a)}\right] \log\left(\frac{1}{2.4\delta}\right)$$

6.4 Remarks

- The tightest high-probability upper bound is obtained by setting b equal to $\underset{b: b_i \in [p_{\kappa_i}, p_{\kappa_i+1}]}{\arg\min} H^*_{\epsilon,b}$ in Theorem 2.
- Although stated for Bernoulli distributions, the results in this paper can easily be extended to rewards in the exponential family (Garivier and Cappé, 2011) by using the appropriate d function.

7 Experiments

7.1 Ranking from Direct Feedback

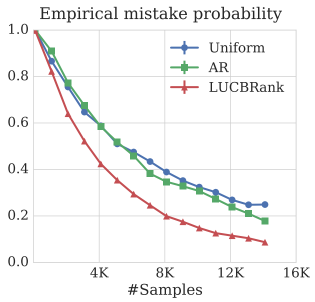


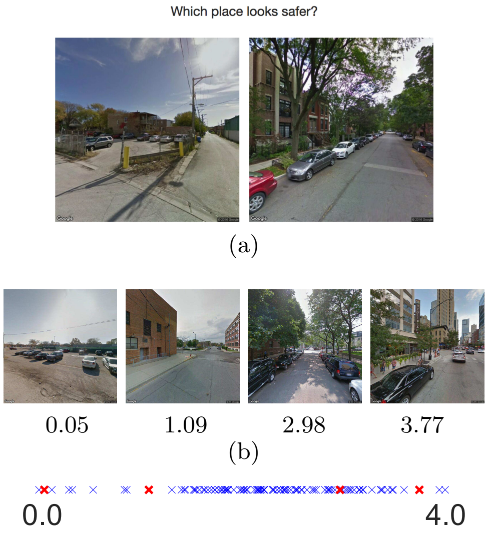
Figure 2: Exp 1 (description in text)

We first compare `LUCBRank` with uniform sampling and the Active Ranking (AR) algorithm (Heckel et al., 2016). AR is an adaptation of the successive elimination approach to solve the coarse ranking problem. It maintains a set of unranked items and samples every item in this set, removing an item from the set when it is confident of the cluster the item belongs to. Although developed for pairwise comparison feedback, AR can easily be adapted to the direct-feedback setting.

We look at the bandit instance B with $K = 15$ arms whose rewards are Bernoulli distributed with means ($p_1 = \frac{1}{2}; p_a = \frac{1}{2} - \frac{a}{40}$ for $a = 2, 3, \ldots, K$). This problem has been studied in the literature in the context of finding the best-arm (Bubeck et al., 2013). We consider the problem of finding the top-3 and the bottom-3 arms, which corresponds to $\kappa_1 = 3, \kappa_2 = 12$.

In Fig. 2, we record the probability (averaged over 1000 simulations) that the empirical clusters returned by the algorithm do not match the true clusters. We set $\delta = 0.1$ for both `LUCBRank` and AR, and $\epsilon = 0$ in `LUCBRank` to have a fair comparison with AR. We see that the mistake probability drops faster for `LUCBRank` than for AR.

7.2 Ranking from Pairwise Comparisons



(a)

(b)

(c)

Figure 3: (a) A sample query on NEXT. (b) Four sample images and their estimated BTL scores beneath. (c) Scatter plot of all the BTL scores, with the sample image markers highlighted.

To measure the performance of our algorithm on real-world data, we selected $K = 100$ Google street view images in Chicago, and collected 6000 pairwise responses on MTurk using NEXT (Jamieson et al., 2015a), where we asked users to choose the safer-looking image out of two images. This experiment is similar to the Place Pulse project (Naik et al., 2014), where the objective is to assess how the appearance of a neighborhood affects its perception of safety. Fig. 3(a) shows a sample query from our experiment. We estimated the safety scores of these street view images from the user-responses by fitting a Bradley-Terry-Luce (BTL) model (Bradley and Terry, 1952) using maximum likelihood estimation, and used this as the ground truth to generate noisy comparisons. Given two items i and j with scores θ_i and θ_j, the BTL model estimates the probability that item i is preferred to item j as $\mathbb{P}(i > j) = \frac{e^{\theta_i}}{e^{\theta_i} + e^{\theta_j}}$. Fig. 3(b) shows 4 images overlayed with their estimated BTL scores (where the lowest score was set to 0), and Fig. 3(c) shows a scatter plot of the scores of all 100 images.

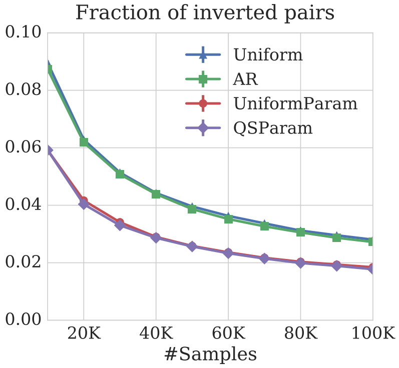


Figure 4: The futility of adaptive methods if the goal is to obtain a complete ranking. We compare uniform sampling with Active Ranking (both use non-parametric rank aggregation), and uniform sampling with quicksort (where both use parametric rank aggregation). We see that the Kendall tau distance of adaptive methods is no lower than those of their non-adaptive counterparts.

We first study the performance of adaptive methods with the goal of finding a *complete ranking*, and observe that adaptive methods offer no advantages when items means are close to each other as they are in this dataset. Oblivious of the generative model, a lower bound (ignoring constants and log factors) on the number of samples required to sort the items by their Borda scores is given by $\sum 1/\Delta_i^2$ (Jamieson et al., 2015b), where the Δ_is are gaps between consecutive sorted Borda scores. For the dataset considered in this experiment, $\sum 1/\Delta_i^2 = 322$ million! We verify the futility of adaptive methods in Fig. 4, where we compare the performance of parametric as well as non-parametric adaptive methods in the literature (we describe these methods shortly) to their non-adaptive counterparts, with a goal of finding a complete ranking of the images. In the parametric algorithms (UniformParam and QSParam), we find MLE estimates of the BTL scores that best fit the pairwise responses. In the non-parametric algorithms (Uniform and AR), we estimate the scores using empirical probabilities in Eq. (2). In Fig. 4, we plot the fraction of pairs that are inverted in the empirical ranking compared to the true ranking, and see no benefits for adaptive methods. We do see gains from adaptivity in the coarse formulation (Fig. 5), as we explain next.

`LUCBRank` can be used in the pairwise comparison setting using Borda reduction, as described in Section 1.1. The adaptive methods in literature we compare to are AR (as in the previous section), and Quicksort (QS) (Ailon et al., 2008; Maystre and Grossglauser, 2017). The Quicksort algorithm works exactly like its non-noisy counterpart: it compares a randomly chosen pivot to all elements, and divides the elements into two subsets - elements preferred to the pivot, and elements the pivot was preferred over. The algorithm then recurses into these two subsets. In this experiment, we stop the quicksort algorithm early as soon as all the subsets are inside the user-specified clusters. Continuing the algorithm further won't change the items in any cluster. This reduces the sample complexity of Quicksort.

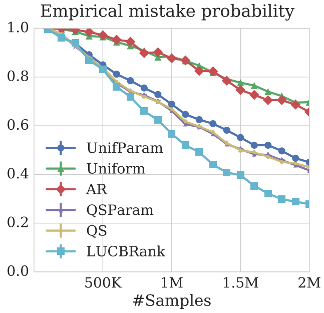


Figure 5: Probability of error in identifying the clusters: `LUCBRank` does better than parametric versions of other active algorithms.

We consider the problem of clustering the images into pentiles ($\kappa_i = 20\,i,\ 1 \le i \le 5$). We set $\delta = 0.1$ for both `LUCBRank` and AR, and $\epsilon = 0$ in `LUCBRank` to ensure a fair comparison with AR. In Fig. 5, we record the probability (averaged over 600 simulations) that the empirical pentiles returned by the algorithm do not match the true pentiles. We find that `LUCBRank` has a lower mistake probability than even the parametric version of Quicksort, which assumes knowledge of the BTL model. As an aside, note that when the items are close as in this experiment, the parametric versions of Uniform and Quicksort perform similarly, and the active nature of Quicksort offers no significant advantage.

In Fig. 6(a) and (b) we plot the ratio of inter-cluster and intra-cluster inversions respectively of `LUCBRank` and Uniform. An inter-cluster pair is a pair of items that are in different clusters in the true ranking, while an intra-cluster pair is a pair of items from the same cluster. We see the that ratio of inter-cluster inversions goes down in Fig. 6(a), because that is the metric `LUCBRank` focuses on. `LUCBRank` does not expend effort on refining its estimate of an item's rank once its cluster has been found, and hence pays a price in the form of intra-cluster inversions (Fig. 6(b)).

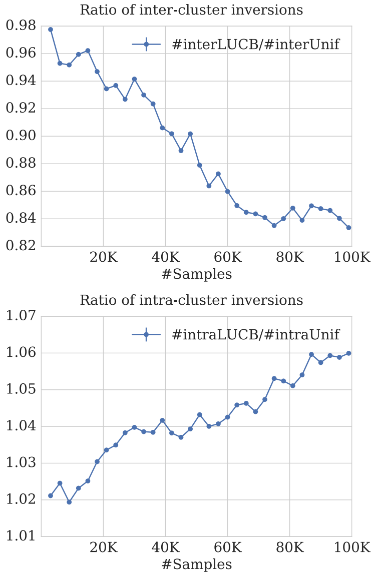


Figure 6: (a) The ratio of inter-cluster inversions of `LUCBRank` and Uniform. (b) The ratio of intra-cluster inversions of `LUCBRank` and Uniform. `LUCBRank` focuses on minimizing inter-cluster inversions at the cost of intra-cluster inversions.

8 Conclusion

The coarse ranking setting is motivated from real-world problems where humans rate items. These problems have high noise and are hard, and a complete ranking is not feasible; fortunately, it is often also not necessary. We propose a practical online algorithm for solving it, `LUCBRank`, and prove distribution-dependent upper and lower bounds on its sample complexity. We evaluate its performance on crowdsourced data gathered using MTurk, and observe that it performs better than existing algorithms in the literature.

We leave open several questions. First, our upper bound is stated in terms of Chernoff information between distributions, while our lower bound is in terms of KL-divergences, and there is a gap between the two. Second, the cluster boundaries need to be user-specified in our current setting. If the gap between the nearest items in adjacent clusters is small, this can adversely affect the sample complexity. Although this is partially addressed through the error-tolerance ϵ, an attractive algorithm would be one which auto-tunes the positions of the cluster boundaries at the widest gaps, subject to user-specified constraints.

To the best of our knowledge, this paper presents the first bandit UCB algorithm for ranking.

Acknowledgements

The authors would like to thank Scott Sievert and Xiaomin Zhang for help with experiments, and Ervin Tanczos for discussions.

References

A. Agarwal, S. Agarwal, S. Assadi, and S. Khanna. Learning with limited rounds of adaptivity: Coin tossing, multi-armed bandits, and ranking from pairwise comparisons. In *Conference on Learning Theory*, pages 39–75, 2017.

S. Agarwal. On ranking and choice models. In *IJCAI*, pages 4050–4053, 2016.

N. Ailon. An active learning algorithm for ranking from pairwise preferences with an almost optimal query complexity. *Journal of Machine Learning Research*, 13(Jan):137–164, 2012.

N. Ailon, M. Charikar, and A. Newman. Aggregating inconsistent information: ranking and clustering. *Journal of the ACM (JACM)*, 55(5):23, 2008.

L. Alonso, P. Chassaing, F. Gillet, S. Janson, E. M. Reingold, and R. Schott. Sorting with unreliable comparisons: A probabilistic analysis. 2003.

J.-Y. Audibert and S. Bubeck. Best arm identification in multi-armed bandits. In *COLT-23th Conference on Learning Theory-2010*, pages 13–p, 2010.

P. Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(Nov):397–422, 2002.

R. A. Bradley and M. E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

M. Braverman and E. Mossel. Sorting from noisy information. *arXiv preprint arXiv:0910.1191*, 2009.

S. Bubeck, N. Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

S. Bubeck, T. Wang, and N. Viswanathan. Multiple identifications in multi-armed bandits. In *International Conference on Machine Learning*, pages 258–265, 2013.

R. Busa-Fekete and E. Hüllermeier. A survey of preference-based online learning with bandit algorithms. In *International Conference on Algorithmic Learning Theory*, pages 18–39. Springer, 2014.

L. Chen, J. Li, and M. Qiao. Nearly instance optimal sample complexity bounds for top-k arm selection. In *Artificial Intelligence and Statistics*, pages 101–110, 2017.

T. M. Cover and J. A. Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

A. Dubey, N. Naik, D. Parikh, R. Raskar, and C. A. Hidalgo. Deep learning the city: Quantifying urban perception at a global scale. In *European Conference on Computer Vision*, pages 196–212. Springer, 2016.

E. Even-Dar, S. Mannor, and Y. Mansour. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *Journal of machine learning research*, 7(Jun):1079–1105, 2006.

M. Falahatgar, A. Orlitsky, V. Pichapati, and A. T. Suresh. Maximum selection and ranking under noisy comparisons. *arXiv preprint arXiv:1705.05366*, 2017.

U. Feige, P. Raghavan, D. Peleg, and E. Upfal. Computing with noisy information. *SIAM Journal on Computing*, 23(5):1001–1018, 1994.

A. Garivier and O. Cappé. The kl-ucb algorithm for bounded stochastic bandits and beyond. In *COLT*, pages 359–376, 2011.

R. Heckel, N. B. Shah, K. Ramchandran, and M. J. Wainwright. Active ranking from pairwise comparisons and the futility of parametric assumptions. *arXiv preprint arXiv:1606.08842*, 2016.

K. G. Jamieson and R. Nowak. Active ranking using pairwise comparisons. In *Advances in Neural Information Processing Systems*, pages 2240–2248, 2011.

K. G. Jamieson, L. Jain, C. Fernandez, N. J. Glattard, and R. Nowak. Next: A system for real-world development, evaluation, and application of active learning. In *Advances in Neural Information Processing Systems*, pages 2656–2664, 2015a.

K. G. Jamieson, S. Katariya, A. Deshpande, and R. D. Nowak. Sparse dueling bandits. In *AISTATS*, 2015b.

H. Jiang, J. Li, and M. Qiao. Practical algorithms for best-k identification in multi-armed bandits. *arXiv preprint arXiv:1705.06894*, 2017.

S. Kalyanakrishnan, A. Tewari, P. Auer, and P. Stone. Pac subset selection in stochastic multi-armed bandits. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 655–662, 2012.

E. Kaufmann and S. Kalyanakrishnan. Information complexity in bandit subset selection. In *COLT*, pages 228–251, 2013.

E. Kaufmann, O. Cappé, and A. Garivier. On the complexity of best arm identification in multi-armed bandit models. *The Journal of Machine Learning Research*, 2015.

L. Maystre and M. Grossglauser. Just sort it! a simple and effective approach to active preference learning. In *Proceedings of Machine Learning Research*, volume 70, 2017.

N. Naik, J. Philipoom, R. Raskar, and C. Hidalgo. Streetscore-predicting the perceived safety of one million streetscapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 779–785, 2014.

S. Negahban, S. Oh, and D. Shah. Iterative ranking from pair-wise comparisons. In *Advances in Neural Information Processing Systems*, pages 2474–2482, 2012a.

S. Negahban, S. Oh, and D. Shah. Rank centrality: Ranking from pair-wise comparisons. *arXiv preprint arXiv:1209.1688*, 2012b.

A. Rajkumar and S. Agarwal. A statistical convergence perspective of algorithms for rank aggregation from pairwise data. In *Proceedings of the 31st International Conference on Machine Learning*, pages 118–126, 2014.

A. Rajkumar, S. Ghoshal, L.-H. Lim, and S. Agarwal. Ranking from stochastic pairwise preferences: Recovering condorcet winners and tournament solution sets at the top. In *International Conference on Machine Learning*, pages 665–673, 2015.

R. Sedgewick and K. Wayne. *Algorithms.* Addison-Wesley Professional, 2011.

N. Shah, S. Balakrishnan, A. Guntuboyina, and M. Wainwright. Stochastically transitive models for pairwise comparisons: Statistical and computational issues. In *International Conference on Machine Learning*, pages 11–20, 2016.

L. L. Thurstone. A law of comparative judgment. *Psychological review*, 34(4):273, 1927.

F. Wauthier, M. Jordan, and N. Jojic. Efficient ranking from pairwise comparisons. In *International Conference on Machine Learning*, pages 109–117, 2013.

A. Wood, J. Martin, and P. Niedenthal. Towards a social functional account of laughter: Acoustic features convey reward, affiliation, and dominance. *PloS one*, 12(8):e0183811, 2017.

9 Appendix

9.1 PAC Guarantee

We'll use the following lemma (Kaufmann and Kalyanakrishnan, 2013) which bounds the probability of 'bad' events in round t.

Lemma 1. *Let $U_a(t)$ and $L_a(t)$ be the confidence bounds defined in Eq.* (6)*. For any algorithm and arm a,*

$$\mathbb{P}(U_a(t) < p_a) = \mathbb{P}(L_a(t) > p_a) \leq e(\beta(t,\delta)\log t + 1)\exp(-\beta(t,\delta))$$

We shall also need the following technical lemma, which we'll use to upper bound the probability of any bad event.

Lemma 2. *If $\beta(t,\delta) = \log(\frac{k_1 K t^\alpha}{\delta}) + \log\log(\frac{k_1 K t^\alpha}{\delta})$,*

$$\sum_{t=1}^{\infty}(\beta(t,\delta)\log t + 1)\exp(-\beta(t,\delta)) \leq \frac{\delta}{k_1 K}\left(\frac{2}{(\alpha-1)^2} + \frac{1}{(\alpha-1)}\right)$$

Proof. Let us consider

$$\begin{aligned}
\beta(t,\delta)(\log t)e^{-\beta(t,\delta)} &= \left(\log\left(\tfrac{k_1 K t^\alpha}{\delta}\right) + \log\log\left(\tfrac{k_1 K t^\alpha}{\delta}\right)\right)(\log t)\left(\frac{\delta}{k_1 K t^\alpha}\cdot\frac{1}{\log\frac{k_1 K t^\alpha}{\delta}}\right) \\
&\leq 2\log\left(\frac{k_1 K t^\alpha}{\delta}\right)\cdot\log t\cdot\left(\frac{\delta}{k_1 K t^\alpha}\cdot\frac{1}{\log\frac{k_1 K t^\alpha}{\delta}}\right) \\
&= 2\log t\cdot\frac{\delta}{k_1 K t^\alpha}
\end{aligned}$$

Hence

$$\begin{aligned}
\sum_{t=1}^{\infty}(\beta(t,\delta)\log t + 1)\exp(-\beta(t,\delta)) &\leq \sum_{t=1}^{\infty}\left(2\log t\cdot\frac{\delta}{k_1 K t^\alpha} + \frac{\delta}{k_1 K t^\alpha}\right) \\
&\leq \frac{\delta}{k_1 K}\left(\frac{2}{(\alpha-1)^2} + \frac{1}{(\alpha-1)}\right)
\end{aligned}$$

□

9.1.1 Proof of Theorem 1

Theoremm. `LUCBRank` *using $\beta(t,\delta) = \log(\frac{k_1 K t^\alpha}{\delta}) + \log\log(\frac{k_1 K t^\alpha}{\delta})$ with $\alpha > 1$ and $k_1 > 1 + \frac{2e}{\alpha-1} + \frac{4e}{(\alpha-1)^2}$, is correct with probability $1-\delta$.*

Proof. Consider the event

$$W = \bigcap_{t\in\mathbb{N}}\bigcap_{a\in\{1,\ldots,K\}}((U_a(t) > p_a)\cap(L_a(t) < p_a))$$

where all arms are well-behaved i.e. their true means are inside their confidence intervals. We show that `LUCBRank` is correct on the event W.

Assume `LUCBRank` fails, which means that when it terminates, there exists a cluster i, such that arm a belongs to cluster i in the returned ranking, and $a \in M^{*,c}_{\epsilon,i}$; that is, either 1) $p_a > p_{\kappa_{i-1}+1} + \epsilon$ or 2) $p_a < p_{\kappa_i} - \epsilon$.

Consider the first case: $p_a > p_{\kappa_{i-1}+1}+\epsilon$. Consequently, there exists arm b such that $p_b \le p_{\kappa_{i-1}+1}$, and $\tau(b) \le \kappa_{i-1}$ in the returned ranking. Since the algorithm stopped and boundary $i-1$ was removed from the set of active boundaries C, it must be the case that $U_a(t) - L_b(t) < \epsilon$ upon stopping. Hence, the following holds:

$$\begin{aligned}
&\bigcup_{t\in\mathbb{N}} (\exists\, a,b : p_a > p_{\kappa_{i-1}+1} + \epsilon, p_b \le p_{\kappa_{i-1}+1}, U_a(t) - L_b(t) < \epsilon)\\
\subseteq &\bigcup_{t\in\mathbb{N}} (\exists\, a,b : (U_a(t) < p_b + \epsilon < p_a) \cup (L_b(t) > p_b))\\
\subseteq &\bigcup_{t\in\mathbb{N}} \bigcup_{a\in\{1,\ldots,K\}} (U_a(t) < p_a) \bigcup_{b\in\{1,\ldots,K\}} (L_b(t) > p_b) \subseteq W^c
\end{aligned}$$

Consider the second case: $p_a < p_{\kappa_i} - \epsilon$. Consequently, there exists an arm b such that $p_b \ge p_{\kappa_i}$, and $\tau(b) > \kappa_i$ in the returned ranking. Since the algorithm stopped and boundary i was removed from the set of active boundaries C, it must be the case that $U_b(t) - L_a(t) < \epsilon$ upon stopping. Hence, the following holds:

$$\begin{aligned}
&\bigcup_{t\in\mathbb{N}} (\exists\, a,b : p_a < p_{\kappa_i} - \epsilon, p_b \ge p_{\kappa_i}, U_b(t) - L_a(t) < \epsilon)\\
\subseteq &\bigcup_{t\in\mathbb{N}} (\exists\, a,b : (U_b(t) < p_b) \cup (L_a(t) > p_b - \epsilon > p_a))\\
\subseteq &\bigcup_{t\in\mathbb{N}} \bigcup_{b\in\{1,\ldots,K\}} (U_b(t) < p_b) \bigcup_{a\in\{1,\ldots,K\}} (L_a(t) > p_a) \subseteq W^c
\end{aligned}$$

Hence

$$\begin{aligned}
\mathbb{P}(\texttt{LUCBRank}\text{ fails}) &\le \mathbb{P}(W^c)\\
&\le 2eK \sum_{t=1}^{\infty} (\beta(t,\delta)\log t + 1)\exp(-\beta(t,\delta)) && \text{(by Lemma 1)}\\
&\le \frac{\delta}{k_1}\left(\frac{4e}{(\alpha-1)^2} + \frac{2e}{(\alpha-1)}\right) && \text{(by Lemma 2)}\\
&\le \delta && \text{(by the constraint on } k_1\text{)}
\end{aligned}$$

□

9.2 Sample Complexity

We define the event W_t which says that all arms are well-behaved in round t i.e. their true means are contained inside their confidence intervals.

$$W_t = \bigcap_{a\in\{1,2,\ldots,K\}} ((U_a(t) > p_a) \cap (L_a(t) < p_a))$$

Note that the event W defined earlier is $W = \cup_{t\in\mathbb{N}} W_t$.

Proposition 1 gives a sufficient condition for stopping.

Proposition 1. *Let $b_i \in [p_{\kappa_i}, p_{\kappa_i+1}]$. If $U_{u_t^i} - L_{l_t^i} > \epsilon$ and W_t holds, then either $k = l_t^i$ or $k = u_t^i$ satisfies*

$$b_i \in \mathcal{I}_k(t) \text{ and } \tilde{\beta}_k(t) > \frac{\epsilon}{2},$$

where we define $\tilde{\beta}_a(t) = \sqrt{\frac{\beta(t,\delta)}{2N_a(t)}}$

Proof. Our W_t condition is stronger than that required in the Proposition 1 in Kaufmann and Kalyanakrishnan (2013), and hence their proof applies. □

Lemma 3 is another concentration result that will be used in our sample complexity guarantee.

Lemma 3. *Let $T \geq 1$ be an integer, and $1 \leq i \leq (c-1)$ be any cluster boundary. Let $\delta > 0, \gamma > 0$ and $x \in]0,1[$ be such that $p_a \neq x$. Then*

$$\sum_{t=1}^{T} \mathbb{P}\left(a = u_i^t \vee a = l_i^t, N_a(t) > \left\lceil \frac{\gamma}{d^*(p_a, x)} \right\rceil, N_a(t) d(\hat{p}_a(t), x) \leq \gamma\right) \leq \frac{\exp(-\gamma)}{d^*(p_a, x)}$$

We prove the following lemma, which states that the Chernoff information increases as the second distribution moves away from the first.

Lemma 4. *If $x < y < y'$ or $x > y > y'$, $d^*(x,y) \leq d^*(x,y')$*

Proof. We shall prove the statement for the case $x < y < y'$. The proof for $x > y > y'$ is analogous.
Let z^* be the unique z such that $d(z^*, x) = d(z^*, y)$. Since $z^* < y < y'$, $d(z^*, y') \geq d(z^*, y) = d(z^*, x)$. Hence, there exists $z^{*'} \geq z^*$ such that $d^*(x,y) = d(z^*, x) \leq d(z^{*'}, x) = d(z^{*'}, y') = d^*(x, y')$. □

Lemma 5. *Let x^* be the solution of the equation:*

$$x = \frac{1}{\gamma}\left(\log \frac{x^\alpha}{\eta} + \log\log \frac{x^\alpha}{\eta}\right)$$

Then if $\gamma < 1$ and $\eta < 1/e^e$,

$$\frac{1}{\gamma}\log\left(\frac{1}{\eta\gamma^\alpha}\right) \leq x^* \leq \frac{C_0}{\gamma}\log\left(\frac{1}{\eta\gamma^\alpha}\right)$$

where C_0 is such that $C_0 \geq \left(1 + \frac{1}{e}\right)\left(\alpha \log C_0 + 1 + \frac{\alpha}{e}\right)$.

Proof. x^* is upper bounded by any x such that $\frac{1}{\gamma}\left(\log \frac{x^\alpha}{\eta} + \log\log \frac{x^\alpha}{\eta}\right) \leq x$. We look for x^* of the form $\frac{C_0}{\gamma}\log\left(\frac{1}{\eta\gamma^\alpha}\right)$.

$$\begin{aligned}
\frac{1}{\gamma}\left(\log \frac{x^\alpha}{\eta} + \log\log \frac{x^\alpha}{\eta}\right) &\leq \frac{1}{\gamma}\left(1 + \frac{1}{e}\right)\log\left(\frac{x^\alpha}{\eta}\right) \\
&= \frac{1}{\gamma}\left(1 + \frac{1}{e}\right)\left(\alpha \log C_0 + \log \frac{1}{\eta\gamma^\alpha} + \alpha \log\log \frac{1}{\eta\gamma^\alpha}\right) \\
&\leq \frac{1}{\gamma}\left(1 + \frac{1}{e}\right)\left(\alpha \log C_0 + \left(1 + \frac{\alpha}{e}\right)\log \frac{1}{\eta\gamma^\alpha}\right) \\
&\leq \frac{1}{\gamma}\left(1 + \frac{1}{e}\right)\left(\alpha \log C_0 + 1 + \frac{\alpha}{e}\right)\log \frac{1}{\eta\gamma^\alpha}
\end{aligned}$$

where the first and second inequalities hold because $\log x \leq \frac{x}{e}$, and the last inequality holds because $\frac{1}{\eta\gamma^\alpha} > e$. Choosing C_0 such that

$$C_0 \geq \left(1 + \frac{1}{e}\right)\left(\alpha \log C_0 + 1 + \frac{\alpha}{e}\right)$$

gives us our upper bound.

To prove the lower bound, consider the series defined by

$$\begin{aligned}
x_0 &= 1 \\
x_{n+1} &= \frac{1}{\gamma}\left(\log \frac{x_n^\alpha}{\eta} + \log\log \frac{x_n^\alpha}{\eta}\right)
\end{aligned}$$

First note that since $\gamma < 1$ and $\eta < 1/e^e$, the sequence is increasing. Second, note that the sequence converges to x^*. Hence

$$\begin{aligned}
x^* \geq x_2 &= \frac{1}{\gamma}\left[\log\left(\frac{1}{\eta\gamma^\alpha}\left(\log\frac{1}{\eta}+\log\log\frac{1}{\eta}\right)^\alpha\right)+\log\log\left(\frac{1}{\eta\gamma^\alpha}\left(\log\frac{1}{\eta}+\log\log\frac{1}{\eta}\right)^\alpha\right)\right]\\
&= \frac{1}{\gamma}\left[\log\frac{1}{\eta\gamma^\alpha}+\alpha\log\left(\log\frac{1}{\eta}+\log\log\frac{1}{\eta}\right)+\log\log\frac{1}{\eta\gamma^\alpha}+\alpha\log\log\left(\log\frac{1}{\eta}+\log\log\frac{1}{\eta}\right)\right]\\
&\geq \frac{1}{\gamma}\log\frac{1}{\eta\gamma^\alpha}
\end{aligned}$$

since $\eta < 1/e^e$. □

Corollary 1. *Let $\gamma = \frac{1}{2H^*_{\epsilon,b}}, \eta = \frac{\delta}{k_1 K}$. Then applying Lemma 5 gives*

$$2H^*_{\epsilon,b}\log\left(\frac{k_1K(2H^*_{\epsilon,b})^\alpha}{\delta}\right) \leq S^*_1 \leq 2C_0(\alpha)H^*_{\epsilon,b}\log\left(\frac{k_1K(2H^*_{\epsilon,b})^\alpha}{\delta}\right)$$

9.3 Proof of Theorem 2

Theoremm. *Let $b = (b_1, b_2, \ldots, b_{c-1})$, where $b_i \in [p_{\kappa_i}, p_{\kappa_i+1}]$. Let $\epsilon > 0$. Let $\beta(t,\delta) = \log(\frac{k_1Kt^\alpha}{\delta}) + \log\log(\frac{k_1Kt^\alpha}{\delta})$ with $k_1 > 1 + \frac{2e}{\alpha-1} + \frac{4e}{(\alpha-1)^2}$. Let τ be the random number of samples taken by* `LUCBRank` *before termination. If $\alpha > 1$,*

$$\mathbb{P}\left(\tau \leq 2C_0(\alpha)H^*_{\epsilon,b}\log\left(\frac{k_1K(2H^*_{\epsilon,b})^\alpha}{\delta}\right)\right) \geq 1-\delta$$

where $C_0(\alpha)$ is such that $C_0(\alpha) \geq \left(1+\frac{1}{e}\right)\left(\alpha\log(C_0(\alpha))+1+\frac{\alpha}{e}\right)$.

Proof. The `LUCBRank` algorithm proceeds in rounds. In a round, it samples the two arms on opposite sides of an active boundary whose confidence intervals overlap the most. A boundary is active as long as this overlap is less than ϵ. Thus, the number of samples up to round T is

$$\begin{aligned}
\#\text{samples}(T) &\leq 2\sum_{t=1}^{T}\sum_{i=1}^{c-1}\mathbb{1}_{(U_{u^i_t}-L_{l^i_t}>\epsilon)}\\
&= 2\sum_{t=1}^{T}\sum_{i=1}^{c-1}\mathbb{1}_{(U_{u^i_t}-L_{l^i_t}>\epsilon)}(\mathbb{1}_{W_t}+\mathbb{1}_{W^c_t})\\
&\leq 2\sum_{t=1}^{T}\sum_{i=1}^{c-1}\mathbb{1}_{(U_{u^i_t}-L_{l^i_t}>\epsilon)}\mathbb{1}_{W_t}+2\sum_{t=1}^{T}\sum_{i=1}^{c-1}\mathbb{1}_{W^c_t}\\
&\leq 2\sum_{t=1}^{T}\sum_{i=1}^{c-1}\sum_{a\in\{1,2,\ldots,K\}}\mathbb{1}_{(a=l^i_t)\vee(a=u^i_t)}\mathbb{1}_{(b_i\in\mathcal{I}_a(t))}\mathbb{1}_{(\tilde{\beta}_a(t)>\frac{\epsilon}{2})}+2\sum_{t=1}^{T}\sum_{i=1}^{c-1}\mathbb{1}_{W^c_t}\\
&\text{(by Proposition 1)}
\end{aligned}$$

We now split the first sum into two depending on whether an arm a belongs to the set $\mathcal{A}_\epsilon = \{a \in \{1, 2, ..., K\} :$

$\Delta_b^* < \epsilon^2/2\}$.

$$\begin{aligned}
\#\text{samples}(T) &\leq 2 \sum_{a \in \mathcal{A}_\epsilon} \sum_{t=1}^{T} \sum_{i=1}^{c-1} \mathbb{1}_{(a=l_t^i)\vee(a=u_t^i)} \mathbb{1}_{\left(N_a(t) < \frac{\beta(t,\delta)}{\epsilon^2/2}\right)} + \\
&\quad 2 \sum_{a \in \mathcal{A}_\epsilon^c} \sum_{t=1}^{T} \sum_{i=1}^{c-1} \mathbb{1}_{(a=l_t^i)\vee(a=u_t^i)} \mathbb{1}_{(b_i \in \mathcal{I}_a(t))} + 2 \sum_{t=1}^{T} \sum_{i=1}^{c-1} \mathbb{1}_{W_t^c} \\
&\leq 2 \sum_{a \in \mathcal{A}_\epsilon} \frac{\beta(T,\delta)}{\epsilon^2/2} + 2 \sum_{a \in \mathcal{A}_\epsilon^c} \sum_{t=1}^{T} \sum_{i=1}^{c-1} \mathbb{1}_{(a=l_t^i)\vee(a=u_t^i)} \mathbb{1}_{\left(N_a(t) \leq \left\lceil \frac{\beta(T,\delta)}{\Delta_b^*(a)} \right\rceil\right)} + \\
&\quad \underbrace{2 \sum_{a \in \mathcal{A}_\epsilon^c} \sum_{t=1}^{T} \sum_{i=1}^{c-1} \mathbb{1}_{(a=l_t^i)\vee(a=u_t^i)} \mathbb{1}_{\left(N_a(t) > \left\lceil \frac{\beta(T,\delta)}{\Delta_b^*(a)} \right\rceil\right)} + 2 \sum_{t=1}^{T} \sum_{i=1}^{c-1} \mathbb{1}_{W_t^c}}_{R_T} \\
&= 2H_{\epsilon,b}^* \, \beta(T,\delta) + R_T
\end{aligned}$$

where

$$R_T = 2 \sum_{a \in \mathcal{A}_\epsilon^c} \sum_{t=1}^{T} \sum_{i=1}^{c-1} \mathbb{1}_{(a=l_t^i)\vee(a=u_t^i)} \mathbb{1}_{\left(N_a(t) > \left\lceil \frac{\beta(T,\delta)}{\Delta_b^*(a)} \right\rceil\right)} \mathbb{1}_{(b_i \in \mathcal{I}_a(t))} + 2 \sum_{t=1}^{T} \sum_{i=1}^{c-1} \mathbb{1}_{W_t^c}$$

If we define $S_1^* = \min\{x : 2H_{\epsilon,b}^* \, \beta(x,\delta) < x\}$, then we get that for $S > S_1^*$, the algorithm must have stopped before S samples on the event $(R_T = 0)$. Denoting the total number of samples used by the algorithm by τ, we have that, for any $S > S_1^*$, $\mathbb{P}(\tau > S) \leq \mathbb{P}(R_T \neq 0)$.

$$\begin{aligned}
&\mathbb{P}(\tau > S) \leq \mathbb{P}(R_T \neq 0) \\
&\leq \mathbb{P}\left(\exists\, a \in \mathcal{A}_\epsilon^c, t \leq T, 1 \leq i \leq (c-1) : a = l_t^i \vee a = r_t^i, N_a(t) > \left\lceil \frac{\beta(T,\delta)}{\Delta_b^*(a)} \right\rceil, b_i \in \mathcal{I}_a(t)\right) + \mathbb{P}(W^c) \\
&\leq \mathbb{P}\left(\exists\, a \in \mathcal{A}_\epsilon^c, t \leq T, 1 \leq i \leq (c-1) : a = l_t^i \vee a = r_t^i, N_a(t) > \left\lceil \frac{\beta(T,\delta)}{d^*(p_a,b_i)} \right\rceil, b_i \in \mathcal{I}_a(t)\right) + \mathbb{P}(W^c) \qquad (13)
\end{aligned}$$

where the final inequality follows because $\Delta_b^*(a) \leq d^*(p_a,b_i) \, \forall\, 1 \leq i \leq c-1$ (by Lemma 4).

Let us look at the first term:

$$\begin{aligned}
&\mathbb{P}\left(\exists\, a \in \mathcal{A}_\epsilon^c, t \leq T, 1 \leq i \leq (c-1) : a = l_t^i \vee a = r_t^i, N_a(t) > \left\lceil \frac{\beta(T,\delta)}{d^*(p_a,b_i)} \right\rceil, b_i \in \mathcal{I}_a(t)\right) \\
&\leq \sum_{a \in \mathcal{A}_\epsilon^c} \sum_{i=1}^{c-1} \sum_{t=1}^{T} \mathbb{P}\left(a = l_t^i \vee a = r_t^i, N_a(t) > \left\lceil \frac{\beta(T,\delta)}{d^*(p_a,b_i)} \right\rceil, b_i \in \mathcal{I}_a(t)\right) \\
&\leq \sum_{a \in \mathcal{A}_\epsilon^c} \sum_{i=1}^{c-1} \frac{\exp(-\beta(T,\delta))}{d^*(p_a,b_i)} \quad \text{(by } \textit{Lemma } 3\text{)} \\
&\leq (c-1)\exp(-\beta(T,\delta)) \sum_{a \in \mathcal{A}_\epsilon^c} \frac{1}{d^*(p_a,b_i)} \\
&\leq (c-1)H_{\epsilon,b}^* \exp(-\beta(T,\delta)) \\
&\leq (c-1)H_{\epsilon,b}^* \exp\left(-\beta\left(\tfrac{S_1^*}{c-1},\delta\right)\right) \qquad \left(\text{if } \tau > S_1^*, T > \tfrac{S_1^*}{c-1}\right)) \\
&= (c-1)H_{\epsilon,b}^* \cdot \frac{\delta(c-1)^\alpha}{k_1 K S_1^{*,\alpha}} \frac{1}{\log \frac{k_1 K S_1^{*,\alpha}}{\delta(c-1)^\alpha}} \\
&\leq (c-1)^{\alpha+1} H_{\epsilon,b}^* \cdot \frac{\delta}{k_1 K \left(2H_{\epsilon,b}^* \log\left(\frac{k_1 K (2H_{\epsilon,b}^*)^\alpha}{\delta}\right)\right)^\alpha} \frac{1}{\log \frac{k_1 K S_1^{*,\alpha}}{\delta(c-1)^\alpha}} \qquad \text{(by the lower bound in } \textit{Corollary } 1\text{)} \\
&\leq \frac{\delta}{k_1} \cdot \left(\frac{c-1}{2}\right)^\alpha \qquad \text{(since } (c-1) \leq K \text{ and } \alpha > 1\text{)}
\end{aligned}$$

For the second term, note that

$$\begin{aligned}\mathbb{P}(W^c) &\leq 2eK\sum_{t=1}^{\infty}(\beta(t,\delta)\log t + 1)\exp(-\beta(t,\delta)) && \text{(by Lemma 1)}\\ &\leq \frac{\delta}{k_1}\left(\frac{4e}{(\alpha-1)^2}+\frac{2e}{(\alpha-1)}\right) && \text{(by Lemma 2)}\end{aligned}$$

Substituting in Eq. (13), we get that for $S > S_1^*$,

$$\mathbb{P}(\tau > S) \leq \frac{\delta}{k_1}\left(\left(\frac{c-1}{2}\right)^{\alpha}+\frac{4e}{(\alpha-1)^2}+\frac{2e}{(\alpha-1)}\right) \leq \delta$$

by the choice of k_1. □

9.4 Lower Bound

The proof uses standard change of measure arguments used to prove lower bounds. For bandit problems, this is succinctly expressed through Lemma 1 in Kaufmann et al. (2015) that we restate here for completeness.

Lemma 6. *Let p and p' be two bandit models with K arms such that for all a, the distributions p_a and p'_a are mutually absolutely continuous. Let σ be a stopping time with respect to $(\mathcal{F}_t)$ and let $A \in \mathcal{F}_\sigma$. Then*

$$\sum_{a=1}^{K}\mathbb{E}_p[N_a(\sigma)]KL(p_a, p'_a) \geq d(\mathbb{P}_p(A), \mathbb{P}_{p'}(A))$$

where $d(x,y) = x\log(x/y) + (1-x)\log((1-x)/(1-y))$.

9.4.1 Proof of Theorem 3

Consider any arm a. By Assumption 1, there exists alternative model p' such that:

$$\mathrm{KL}(p_a, p_{\kappa_{g(a)}+1}) < \mathrm{KL}(p_a, p'_a) < \mathrm{KL}(p_a, p_{\kappa_{g(a)}+1}) + \alpha \text{ and } p'_a < p_{\kappa_{g(a)}+1}$$

Note that in the model p', arm a no longer belongs to the cluster $g(a)$. Let $\hat{M}_{g(a)}$ be the set of arms returned by an algorithm in the $g(a)^{\mathrm{th}}$ cluster. If we define the event $A = \{a \in \hat{M}_{g(a)}\} \in \mathcal{F}_\tau$, then by definition, for any δ-PAC algorithm, $\mathbb{P}_p(A) \geq 1-\delta$ and $\mathbb{P}_{p'}(A) \leq \delta$. Letting $N_a(\tau)$ denote the number of pulls of arm a by time τ, we have by Lemma 6 and the monotonicity of $d(x,y)$ that

$$\mathrm{KL}(p_a, p'_a)\mathbb{E}_p[N_a(\tau)] \geq d(1-\delta, \delta) \geq \log(\tfrac{1}{2.4\delta})$$

where we use the property that for $x \in [0,1]$, $d(x, 1-x) \geq \log\frac{1}{2.4x}$. This gives us that

$$\mathbb{E}_p[N_a(\tau)] \geq \frac{1}{\mathrm{KL}(p_a, p_{\kappa_{g(a)}+1}) + \alpha}\log\left(\frac{1}{2.4\delta}\right)$$

Letting $\alpha \to 0$, we get

$$\mathbb{E}_p[N_a(\tau)] \geq \frac{1}{\mathrm{KL}(p_a, p_{\kappa_{g(a)}+1})}\log\left(\frac{1}{2.4\delta}\right) \tag{14}$$

Similarly, by considering an alternative model p'' such that

$$\mathrm{KL}(p_a, p_{\kappa_{g(a)-1}}) < \mathrm{KL}(p_a, p''_a) < \mathrm{KL}(p_a, p_{\kappa_{g(a)-1}}) + \alpha \text{ and } p''_a > p_{\kappa_{g(a)-1}}$$

we get

$$\mathbb{E}_p[N_a(\tau)] \geq \frac{1}{\mathrm{KL}(p_a, p_{\kappa_{g(a)-1}})}\log\left(\frac{1}{2.4\delta}\right) \tag{15}$$

From Eq. (14), Eq. (15), and the definition of $\Delta_\kappa^{\mathrm{KL}}(a)$ in Eq. (11), we get that

$$\mathbb{E}_p[N_a(\tau)] \geq \frac{1}{\Delta_\kappa^{\mathrm{KL}}(a)}\log\left(\frac{1}{2.4\delta}\right)$$

Summing over all the arms yields the required bound for $\mathbb{E}_p[\tau] = \sum_{a=1}^{K}\mathbb{E}_p[N_a(\tau)]$.