Group-Envy Fairness in the Stochastic Bandit Setting

by

Stephen Scinocca
B.Sc., University of Victoria, 2019
M.Sc., University of Victoria, 2022

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

Group-Envy Fairness in the Stochastic Bandit Setting

by

Stephen Scinocca
B.Sc., University of Victoria, 2019
M.Sc., University of Victoria, 2022

Supervisory Committee

Dr. N. Mehta, Supervisor
(Department of Computer Science, University of Victoria)

Dr. V. Srinivasan, Member of Supervisory Committee
(Department of Computer Science, University of Victoria)

Supervisory Committee

Dr. N. Mehta, Supervisor
(Department of Computer Science, University of Victoria)

Dr. V. Srinivasan, Member of Supervisory Committee
(Department of Computer Science, University of Victoria)

ABSTRACT

We introduce a new, group fairness-inspired stochastic multi-armed bandit problem in the pure exploration setting. We look at the discrepancy between an arm's mean reward from a group and the highest mean reward for any arm from that group, and call this the disappointment that group suffers from that arm. We define the optimal arm to be the one that minimizes the maximum disappointment over all groups. This optimal arm addresses one problem with maximin fairness, where the group used to choose the maximin best arm suffers little disappointment regardless of what arm is picked, but another group suffers significantly more disappointment by picking that arm as the best one. The challenge of this problem is that the highest mean reward for a group and the arm that gives that reward are unknown. This means we need to pull arms for multiple goals: to find the optimal arm, and to estimate the highest mean reward of certain groups. This leads to the new adaptive sampling algorithm for best arm identification in the fixed confidence setting called MD-LUCB, or Minimax Disappointment LUCB. We prove bounds on MD-LUCB's sample complexity and then study its performance with empirical simulations.

Contents

List of Tables

List of Figures

ACKNOWLEDGEMENTS

I would like to thank all of the graduate students I have worked with along the way. First, a thank you to Bingshan, who helped me understand core concepts and helped me feel like I belonged as a masters student. I would like to thank Ali, Andrea, Quan, and Mica, for keeping me motivated and entertained during our time working together in the lab, and I'd like to thank Joe for being there with me both taking and teaching classes. I will never forget your hatred of twitter.

I would like to thank my friends and family for supporting me throughout my masters, and always believing in me that I could do it, even when I was not sure I could myself. A special thank you to Arianne for helping proofread some of this thesis and being an amazing partner who helps brighten even my darkest moods.

Finally, I would like to thank my supervisor, Nishant Mehta, for being the best supervisor I could have asked for. His understanding and guidance helped me become a better writer, speaker, and overall person, and I owe a lot of this to his amazing work ethic and patience. I am very glad that you took me on as a master student.

Chapter 1

Introduction

As a society, we are putting more trust into machine learning algorithms to help make decisions that impact people in many ways. It is important these decisions are done in a fair way for all groups of people affected by those choices. When this is not considered, we get algorithms such as COMPAS, which stands for Correctional Offender Management Profiling for Alternative Sanctions. This algorithm was shown to be racially biased against African Americans defendants [Khademi and Honavar, 2020]. An example of bias in healthcare was an algorithm determining if patients in the US had complex health problems and needed extra help. This was found to be racially biased, as the metric it was using to identify people with these issues was their cost of health care, for which different ethnicities can only afford certain amounts on average compared to others [Obermeyer et al., 2019]. There are many other examples of how these algorithms can cause harm to our society [Mehrabi et al., 2021], and so we need to consider different definitions of fairness for different settings that we can judge our algorithms by. The type of fairness we will be looking at is group fairness.

In the group fairness setting, we split our population into distinct sub-populations by some variable, where our goal is to ensure we are treating each of these separate groups fairly in relation to the other groups. An example of a way we might split our groups is by sex, as we would want to ensure we are treating men and women fairly

in comparison to each other. The problem setting we look to consider group fairness to is stochastic multi-armed bandit problems. For these types of problems we get a choice of actions, where each action corresponds to distribution. At a given time step, we pick one action, and we get some reward, where the reward is random variable drawn from the distribution associated with that arm at that time. When we have groups as well, each arm-group pair corresponds to a distribution, so we get to pick an action and a group at a time step and see the reward of that action for that group at that time step. Our goal is to recommend the single best action for all of the groups, where the way we define the best action is dependent on our definition of fairness.

For example, consider the following clinical trials problem: we are in early testing for the effectiveness of a subset of drugs that are trying to alleviate anxiety. We sequentially pick one drug (action) to give to one person from a distinct group and observe the result before making our next choice of drug and group. We can only afford to bring one drug to market, as it is very expensive to mass produce a new drug. We also want to use an adaptive sampling strategy, as to not waste money testing drugs we do not end up using. One way of grouping people could be by their age or ethnicity. We want to find some rule for deciding which drug to bring to mass market. One idea is to make sure that the drug we pick has the higher reward on its worst performing group than any of the other drugs on their worst performing groups. This idea of fairness is called maximin or minimax fairness [Martinez et al., 2020, Williamson and Menon, 2019, Shekhar et al., 2021].

Maximin fairness measures each action according to the minimum mean reward over a set of groups, and the action that maximizes this measure is deemed optimal. Minimax fairness is similar, but switches rewards for losses. The maximin measure of fairness does not take into account the discrepancy in mean rewards for each group, and therefore the optimal maximin action is oblivious to this difference. This means that maximin fairness cannot capture how envious groups may be of each other. If the

gap between the mean reward from the best performing action for a group and the mean reward from the chosen action for a group is larger for one group than another, then the larger gap group may be envious of the smaller gap group if we picked that action for every group. One negative implication of such envy is that a group may decide not to participate in a collective decision-making system due to their feeling that the mechanism is not fair [Smith and McDonough, 2001].

We define an envy-based notion of fairness, called *disappointment fairness*, and consider it in the stochastic multi-armed bandit setting. We define disappointment to be the gap between the highest mean reward action in a group and the mean reward for a specific action in the same group. The goal of disappointment fairness is to minimize disappointment that groups feel when we are charged with picking one action for every group. Therefore, our notion of unfairness is the maximum disappointment over all groups, and we consider one group to envy another group if its disappointment is much higher than that of the other group. In the clinical trials example, this would look at if the public knew how effective each drug was on every group. Clearly, each group would wish to have the best performing drug for their group on the market, so someone from a group would feel a level of disappointment equal to the difference in effectiveness of the drug chosen to go to mass production in comparison to the best they could have had. Our new form of fairness, disappointment fairness, focuses on selecting the action that minimizes the maximum disappointment any group can feel; we call that action our best action or best arm. An example that shows the differences between maximin fairness and disappointment fairness can be seen in Figure 1.1.

If we think of this problem in the multi-armed bandit setting, then it becomes a fixed confidence best arm identification problem. One major difficulty with this problem is we do not know the highest mean reward of each group or which arm has that reward. To deal with this, we want to pull arms for two different purposes. We want to pull arm-group pairs to help us find the best arm, and we want to pull

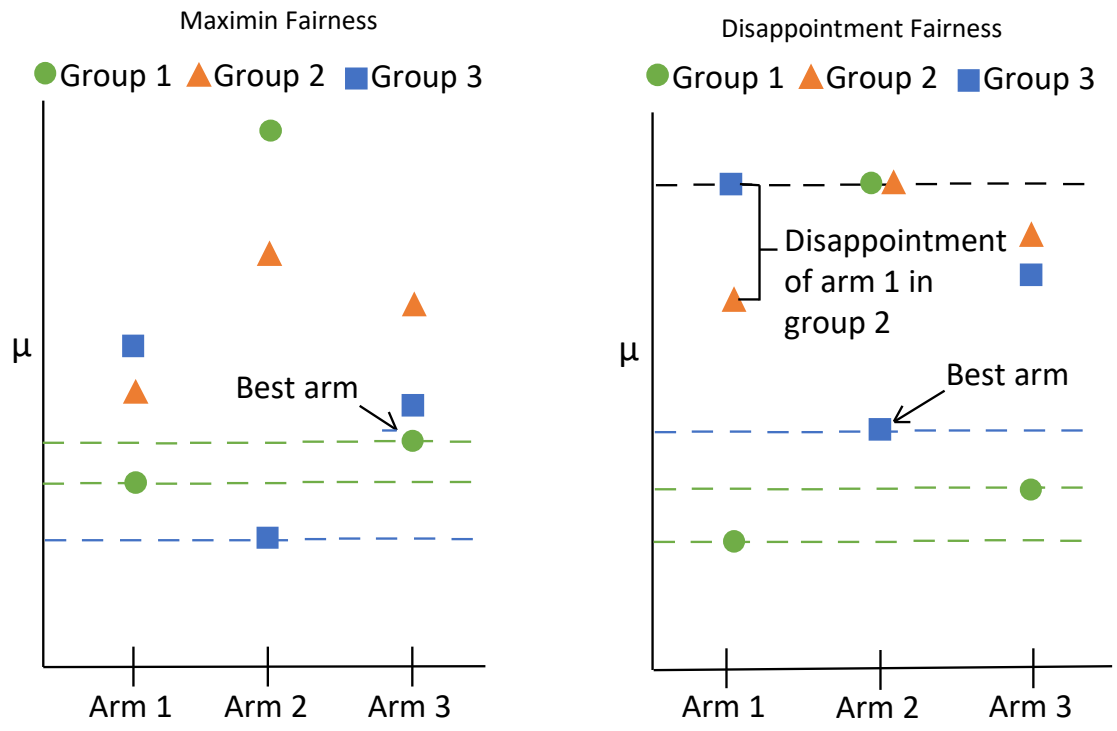

Figure 1.1: The left side displays maximin fairness with the mean rewards of 3 arms and 3 groups. The right side displays the same arms and groups with the same mean rewards, but now we have normalized the values of the highest mean rewards in each group to compare disappointment levels. The lines on both sides represent the worst performing group for each arm under the different fairness settings. The highest of these lines is our best arm, making arm 3 the best for maximin fairness, and arm 2 the best arm for minimizing maximum disappointment.

arm-group pairs to estimate the highest mean rewards from the groups.

Another difficulty is there are cases where we do not need to know with high precision the highest mean reward arm for all groups. For example, if for one of the groups, all the arms have similar mean rewards, then none of the arms could have their maximum disappointment come from this group, and it would be very costly to find what the highest mean reward arm in that group is. Thus, we would want to reduce the number of pulls from that group. So our dynamic pulling approach must also deal with the fact that we do not want to over-sample from a group trying to find the high-

est mean reward if it is unnecessary to our main objective, which is finding the best arm.

The two main contributions presented are the new idea of fairness, which we call disappointment fairness or minimizing maximum disappointment, and a new algorithm for the novel best-arm identification problem in the fixed confidence setting. This algorithm is called MD-LUCB, which stands for Minimax Disappointment Lower Upper Confidence Bound. MD-LUCB pulls arms in two different situations: pulling arms with the goal of estimating the highest mean reward arm in given groups, and pulling arms with the goal of identifying which arm minimizes the maximum disappointment. To estimate the highest mean rewards from groups, MD-LUCB uses a subroutine called the Incremental Exponential Gap (INC-EG) algorithm, which is a variation on the Exponential Gap (EG) algorithm [Karnin et al., 2013]. We give an analysis on the sample complexity of both the MD-LUCB and INC-EG algorithms, and show that MD-LUCB outputs the best arm with high probability.

In Chapter 2, we provide background information. We also introduce a new algorithm INC-EG, similar to the Exponential Gap algorithm. INC-EG outputs the mean reward of the highest reward arm instead of the arm that contains that reward. Chapter 3 introduces the notion of disappointment fairness, and we discuss some of the benefits and important facts about defining fairness this way. Chapter 4 introduces the algorithm MD-LUCB, which is a fixed confidence best arm identification algorithm for stochastic bandit problems with disappointment fairness determining the best arm. This chapter also discusses INC-EG, which is called as a subroutine of MD-LUCB for estimating the value of the highest reward arm in groups. Chapter 5 analyzes the sample complexity of these algorithms, and proves the correctness of the algorithm. Chapter 6 contains experiments using different settings of MD-LUCB, and Chapter 7 talks about future work and concludes this thesis. A glossary of terms, extra proofs and extra tables not included in the main body of the thesis are provided in Appendix A.

Chapter 2

Background

2.1 Multi Armed Bandit Problems

Multi-armed bandit problems were first proposed in the 1950's by Robbins [Robbins, 1952], but were not explored deeply until the 1980's [Lai et al., 1985]. Multi-armed bandits are useful for many real world applications, such as online ad selection [Chakrabarti et al., 2008], healthcare [Durand et al., 2018], user content matching [Pandey et al., 2007], finance [Shen et al., 2015], and more [Bouneffouf et al., 2020]. Our focus will be on stochastic multi-armed bandit problems.

In a stochastic multi-armed bandit problem, there is a collection of K unknown distributions. There is a set A indexed by $\{1, 2, \dots K\}$ of K arms, which is equal to the number of distributions there are. If we want to draw a sample from one of these distributions, we say we are pulling arm a, where there is one arm for each distribution. At each time step t, the learner picks one arm and pulls it, getting one sample from the distribution associated with that arm. The learner is then shown the reward of that sample at time t. These rewards are bounded between 0 and 1. This repeats for T times. The expected reward of arm a is denoted by μ_a. The protocol is summarized in Figure 2.1.

For time step $t = 1, 2 \ldots, T$

- Learner picks an arm/environment to give to nature
- Nature samples from that environment and receives a reward
- Nature reveals to the learner the reward from the chosen environment

Figure 2.1

We define regret R after T rounds as the difference between the expected sum of reward associated with an optimal strategy and the sum of the collected rewards from the learner. In the case of multi-armed bandits, the optimal strategy is always picking the arm with the highest mean reward, so we would say the regret our learner faces after T rounds is the expected difference in reward between our expected reward if we pulled the best arm T times and the reward our learner receives after T rounds.

Multi-armed bandit problems fall under two categories: regret minimization and best arm identification. The goal of regret minimization is as the name implies, to minimize the regret of the learner. Regret minimization requires a balance of exploration and exploitation of arms, as you want to exploit the best current arm, but still explore other arms occasionally in case they are actually be the best arm. A best arm identification problem deals with pure exploration, which means determining the best arm is our only goal. For a standard multi-armed bandit problem, the best arm is defined as the arm with the highest expected reward. Our focus will be on best arm identification problems. When working with best arm identification, we define problems as (ϵ, δ)-PAC, which means the mean reward of our outputted best arm is going to be within ϵ of the the mean reward of the actual best arm with probability at least $1-\delta$.

Best arm identification has been heavily explored (See, e.g., [Audibert et al., 2010, Even-Dar et al., 2006, Bubeck et al., 2009]), and have two different settings: fixed confidence and fixed budget. In the fixed budget setting, we are trying to maximize

the probability that we have selected the best arm when allocated with at most a fixed amount of samples. In the fixed confidence setting, we are trying to minimize the number of samples needed to guarantee with at least a fixed probability that the outputted arm is the best. The number of samples needed to be able to accurately output the correct response with high probability is known as the sample complexity of the problem. The sample complexity of an algorithm is the number of samples needed for that algorithm to output the correct response with high probability. The sample complexity of best arm identification problems are normally problem dependent, meaning that the number of samples is dependent on $\Delta_a = \mu^* - \mu_a$, where μ^* is the mean reward of the highest mean reward arm, or the best arm in the standard multi armed bandit setting, and Δ_a is the gap between the mean rewards of the best arm and some arm a. Now we look at some different algorithms for finding the best arm in the fixed confidence setting.

2.2 Best Arm Identification Algorithms

When working with standard best arm identification problems, we assume that there is a distinct best arm, meaning that only one arm can have the value of μ^*. When first exploring this problem setting, it was unknown what the lower bound on the number of samples needed is for this setting. However, in lil'UCB [Jamieson et al., 2014], it was proven that for standard best arm identification you will need at least $(1/\Delta_a^2)\log(1/\delta)\log\log(1/\Delta_a^2)$ samples. In other words, we need this many samples to be able to tell say with high probability which arm has the highest mean reward.

We are going to introduce 4 different algorithms. Initially, we introduce the Median Elimination algorithm, which is one of the first best arm identification algorithms. Next we introduce the Exponential Gap algorithm, which uses Median Elimination as a sub-routine, and has optimal sample complexity. After this we introduce the Incremental Exponential Gap algorithm, which is an original algorithm from our research. It is

included in the background to show its connection to the Exponential Gap algorithm, and to explain the differences between them. Finally, we discuss the LUCB algorithm or Lower Upper Confidence Bounds algorithm. This algorithm was initially used to solve top w arms in a bandits problem, but the concept is introduced here as the main algorithm in this work (MD-LUCB) uses the same main technique that LUCB employs.

2.2.1 Median Elimination Algorithm

Algorithm 1: Median Elimination

Input: $\epsilon > 0, \delta > 0$
Set $S_1 = [K]$, $\epsilon_1 = \epsilon/4$, $\delta_1 = \delta/2$, $r = 1$
while $S_r > 1$ **do**
    Sample every arm $a \in S_r$ for $\frac{r}{(\epsilon_r/2)^2}\log(\frac{3}{\delta_r})$ times, and let $\hat{\mu}_a^r$ denote each arm's empirical mean
    Find the median arm's empirical mean and call it $\hat{m}_r$
    Set $S_{r+1} = S_r \setminus \{a : \hat{\mu}_a^r < \hat{m}_r\}$
    Update $r = r + 1$
Output remaining arm in S_r

One of the first algorithms used for best arm identification being Median Elimination [Even-Dar et al., 2006]. Median Elimination (Algorithm 1) works in rounds, where each round it pulls all of the remaining arms the same number of times, then removes half of the remaining arms based on their empirical reward. The algorithm halts once there is one arm remaining. Median Elimination is (ϵ, δ)-PAC, with a sample complexity of

$$O\left(\frac{K}{\epsilon^2}\log(\frac{1}{\delta})\right).$$

2.2.2 Exponential Gap Algorithm

Algorithm 2: Exponential Gap Algorithm

Input: $\delta > 0$
Initialize $S_1 = [K]$, $r = 1$
while $|S_r| > 1$ **do**
    Let $\epsilon_r = 2^{-r}$, and $\delta_r = \delta/(50r^2)$
    Sample each arm $t_r = (\frac{2}{\epsilon_r}) \ln(\frac{2}{\delta_r})$ times, and let $\hat{\mu}_a^r$ be the average reward
    Invoke $a_r = \text{MedianElimination}(S_r, \epsilon_r/2, \delta_r)$
    Set $S_{r+1} = S_r \setminus \{a \in S_r : \hat{\mu}_a^r < \hat{\mu}_{a_r}^r - \epsilon_r\}$
    update $r = r + 1$
Output remaining arm in S_r

This was further improved on with the Exponential Gap (EG) Algorithm (Algorithm 2) [Karnin et al., 2013], which has the optimal sample complexity for best arm identification, as seen in Section 2 of [Jamieson et al., 2014], and uses Median Elimination as a subroutine. This algorithm works in rounds, where in each round it pulls every remaining arm the same amount in a given round to get an estimate on the mean reward of each arm a as $\hat{\mu}_a$ that has a confidence radius of $\epsilon_r/2$. Then, EG runs an iteration of Median Elimination that determines which arm is best arm for round a_r, with a guarantee that the expected mean reward μ_{a_r} is within $\epsilon_r/2$ of the expected mean reward of μ^*. Finally, EG eliminates any arms that are not within ϵ_r of the empirical reward of the arm a_r, as if the gap in empirical mean reward between a_r and some arm a is bigger than ϵ_r, we can say that with high probability the eliminated arm was not the best arm. It stops when there is one arm remaining and outputs that arm as the best arm.

Algorithm 2 is $(0, \delta)$-PAC, and we can use Theorem 1 from [Karnin et al., 2013] to give a sample complexity bound on this algorithm.

Theorem 1. *With probability at least* $1 - \delta$*, Algorithm 2 identifies the optimal arm*

using

$$O\left(\sum_{a=2}^{K}\frac{1}{(\Delta_a)^2}\log\left(\frac{1}{\delta}\log\left(\frac{1}{(\Delta_a^2)}\right)\right)\right)$$

arm pulls.

In [Karnin et al., 2013], they also introduce a variation to the Exponential Gap algorithm that is (ϵ, δ)-PAC, where the only change to the algorithm changing the stopping condition so that the algorithm either stops when there is one arm remaining or when it has completed $O(\log(1/\epsilon)$ rounds. Theorem 2 gives a sample complexity claim to this variation. We will use this theorem and the exponential gap algorithm proofs to give us a new algorithm that solves a slightly different problem than standard best arm identification in the next part, called the Incremental Exponential Gap (INC-EG) algorithm.

Theorem 2. *There exists an algorithm that with probability at least* $1-\delta$*, finds an* ϵ*-optimal arm using*

$$O\left(\sum_{a=2}^{K}\frac{1}{(\Delta_a^\epsilon)^2}\log\left(\frac{1}{\delta}\log\left(\frac{1}{(\Delta_a^\epsilon)^2}\right)\right)\right)$$

arm pulls, where $\Delta_a^\epsilon := \max(\Delta_a, \epsilon)$.

2.2.3 Incremental Exponential Gap Algorithm

Algorithm 3: Incremental Exponential Gap

Input: $\delta > 0, r > 0, S \subseteq A$
Let $\epsilon = 2^{-r}/4, \delta = \delta/(50r^2)$
Sample each arm $a \in S$ a total of $t = (\frac{2}{\epsilon^2})\ln(\frac{2}{\delta})$ times to get estimated means $\hat{\nu}_a$
Invoke $a^* = \text{MedianElimination}(S, \epsilon/2, \delta)$.
Set $S_2 = S \setminus \{a \in S : \hat{\mu}_{a^*} < \hat{\mu}_a - \epsilon_r\}$
Return $\hat{\nu}_{a^*}$, S_2

We introduce a new algorithm based on the Exponential Gap algorithm called the Incremental Exponential Gap (INC-EG) Algorithm. The main distinction between

these algorithms is that EG outputs a guess of the best arm, whereas INC-EG outputs a value for the estimated mean reward of the best arm. The goal of INC-EG is to output an estimate of the highest mean reward from a set of arms, and have guarantees on how precise that estimate is. INC-EG runs for a single round, where it still uses Median Elimination to determine which arm has is going to be our best arm, and draws samples to get an estimate of the reward of that arm.

We keep track of the set S_2 as our new set of remaining arms, as we may want to call INC-EG again to get an even more precise estimate of the mean reward of the best arm, and we would then pass in the set S_2 to INC-EG. When INC-EG eliminates arms, we can say with high certainty the eliminated arms are not the best arm, thanks to Lemma 3.3 from [Karnin et al., 2013], which holds for INC-EG as well.

If we want to use this algorithm to estimate μ^*, then we can use Theorem 3. This is how we will be using INC-EG as a subroutine in the algorithm MD-LUCB in Chapter 4.

Theorem 3. *The empirical mean reward $\hat{\mu}_{a_r}$ outputted from INC-EG for round r will be within ϵ_r of μ^* with high probability $1 - 2\delta_r$.*

Proof. Initially, from Lemma 3.3 from [Karnin et al., 2013], we can say with high probability that the best arm is still in the remaining arm set after r rounds. This holds true for INC-EG as well, as the way we pull and eliminate arms for any round r of INC-EG is the same as in round r of the while loop of EG.

Now that we know the best arm still remains in the arm set, we want to show that $\hat{\mu}_{a_r} > \mu^* - \epsilon_r$. We know from the call to Median Elimination inside of INC-EG that the mean reward from the outputted arm a_r (μ_{a_r}) will be within $\epsilon_r/2$ of μ^*. From Hoeffding's inequality, we know that the empirical mean reward $\hat{\mu}_a$ of every arm a will be within $\epsilon_r/2$ of its true mean reward μ_a with probability $1 - \delta_r$. From this we

get that $\hat{\mu}_{a_r} > \mu_{a_r} - \epsilon_r/2 > \mu^* - \epsilon_r$ with at least probability $1 - 2\delta_r$. For the other side, it is obvious that $\hat{\mu}_{a_r} < \mu^* + \epsilon_r$, as $\hat{\mu}_{a_r} < \mu_{a_r} + \epsilon_r/2 < \mu^* + \epsilon_r/2$.

Therefore we know that $\hat{\mu}_{a_r}$ can only at most be ϵ_r away from μ^*, so this algorithm will always output an empirical mean reward that is within ϵ_r of μ^*, and we can use the outputted empirical mean to form a confidence interval around μ^*. □

2.2.4 LUCB: An Algorithm for Finding the Top w Arms

Algorithm 4: LUCB

```
Input: δ > 0, Arm set A
Set h*_t = a ∈ A, l*_t = a ∈ A, t = 0
while L_{h*_t} < U_{l*_t} do
    Update which arms are in sets X and Y
    Update arms h*_t and l*_t and pull each once
    Update all confidence intervals
    t = t + 1
Output set of arms A
```

Now we look at a variation of the standard problem setting. Our goal is now to find the top w arms, instead of the single best arm, where the higher the expected reward an arm has, the better it is. The LUCB, or lower upper confidence bound technique [Kalyanakrishnan et al., 2012], was first used to solve this problem. LUCB (Algorithm 4) is an epoch based algorithm, where every epoch we split the arms into two sets: the upper set and the lower set, which we call sets X and Y. The upper set contains the w arms with the highest empirical averages, and the lower set contains the remaining $K - w$ arms. We then find arm $h_t^* = \min_{a \in X} L_a(t)$, which is the arm with the smallest lower confidence bound in set X at time t, and arm $l_t^* = \max_{a \in Y} U_a(t)$, which is the arm with the highest upper confidence bound in set Y at time t. We think of arm h_t^* as the w^{th} biggest arm at time t and arm l_t^* as the $(w+1)^{th}$ biggest arm at time t. Every epoch we check if $L_{h,t} < U_{l,t}$. If this true, we pull both arms l_t^* and h_t^* and go to the next epoch. If this is false, we so we stop our algorithm, as this implies that

the upper confidence bound of all arms in Y is lower than the lower confidence bound of all arms in X. If we can say with high probability the mean reward of every arm is contained within its confidence interval, then we can say we have found the best w arms. This stopping condition is useful in other best arm identification problems as well, such as [Garivier et al., 2016] and in MDLUCB, defined in Chapter 4.

2.3 Fairness in Bandit Problems

The most common definition of a "best" arm is the arm with the highest mean reward. However, this is not the only way of defining the best arm. Another way could be finding the most "fair" arm, where our problem setting is shifted from the original bandit setting to one where we split our population into distinct groups. Let G be a set of groups, going from 1 to m. Each time we pick an arm, we also pick a corresponding group, and then we see the reward that that group receives from that arm at that time. Each arm-group pair we assume is independent of other arm-group pairs, so there is no dependence between the same arm in different groups or the same group with different arms. An example of this can be clinical trials, where it has been shown that drugs can have different levels of effectiveness between men and women [Whitley and Lindsey, 2009]. If we define the best arm as the one which generates the highest average reward from all the groups, some groups may have very little reward compared to others with very high rewards, which can be seen as unfair. This unfairness leads to different ways of determining which is the best arm in this setting.

There have been different forms of fairness proposed in the multi-armed bandit setting. Some papers have focused on fairness for regret minimization, where they make sure each arm is still being pulled a constant fraction of the time, or a minimum number of times [Patil et al., 2021, Chen et al., 2020]. We focus on fairness with best arm identification.

[Garivier et al., 2016] consider minimax fairness in the best arm identification setting, but it is expressed game-theoretically as doing maximin action identification. This paper looks at playing a zero-sum two player game where player one makes a choice (picks an arm), and then player two makes their best choice (picks a group for that arm). Under this constraint, the optimal arm will be the arm maximizes the minimum reward player one can receive assuming player two does the optimal response action. We are going to frame this response action as player two choosing a group for player one's choice of arm. The algorithm this paper presents, called Maximin Lower Upper Confidence Bound (M-LUCB), uses a similar stopping rule to LUCB. For each arm a at time t we define the group

$$g_a(t) = \arg\min_{g \in G} L_{a,g}(t),$$

where $L_{a,g}(t)$ is the lower confidence bound of arm a in group g, so $g_a(t)$ is the group with the smallest lower confidence bound for arm a at time step t. From here, we define our two arms we pull as

$$\hat{a}_t = \arg\max_{a \in A} \hat{\mu}_{a,g_a(t)}(t),$$

and

$$\hat{b}_t = \arg\max_{a \in A, a \neq \hat{a}_t} U_{a,g}(t),$$

where $U_{a,g}(t)$ is the upper confidence bound for arm a at time t. These two arms with their groups make pairs $H_t = (\hat{a}_t, g_{\hat{a}_t})$ and $S_t = (\hat{b}_t, g_{\hat{b}_t})$. M-LUCB pulls these two pairs, and its stopping condition checks if the lower confidence bound of H_t is still higher than the upper confidence bound of S_t. If this is true, M-LUCB finishes and outputs a_t from pair H_t as the best arm; if it is not true, they recalculate H_t and S_t and run another epoch.

When we frame this same problem in terms of maximin fairness instead of two

players in a zero-sum game, we can have player one select an arm, and player 2 after seeing what arm player one has picked selects a group, and the optimal choice for player one would be the maximin fair choice for all of the groups under this alternative framing of the problem.

The origin of this definition of group envy fairness comes from a paper on principle component analysis (PCA) [Samadi et al., 2018], although they do not call it disappointment. For their problem, they are looking at how for standard PCA, there can be a higher reconstruction error for a population A than another population B when running PCA on both populations, and coming up with a fair way to reduce the difference in reconstruction error between the two populations. We can have a comparison the least reconstruction error we can have for population A by running PCA on just population A, and then do the same for population B, which will have reconstruction errors E_A^* and E_B^*. We can think of these values as the optimal choices for our two populations, or the best arms for the groups in the multi-armed bandit setting. Then when running their version of PCA with reconstruction error $E_A(S)$ for population A and $E_B(S)$ for population B, for each sample they take into account E_A^* and E_B^* depending on which population the sample comes from. Their goal is to minimize the difference between E_A^* and $E_A(S)$ for population A samples and E_B^* and $E_B(S)$ for population B samples. In other words, the goal is to pick the best low-rank subspace S, where

$$S = \arg\min_{S'} \{E_A^* - E_A(S'), E_B^* - E_B(S')\},$$

which we can think of in the multi armed bandit setting of a 2 group problem where we define our best arm as the arm which minimizes the maximum difference between the best arm in a group and that arm in a group.

Chapter 3

Disappointment Minimax Fairness

3.1 Problem Setup

We consider a stochastic game where we have a set of arms A indexed by $a \in \{1, 2, \ldots, K\}$, and a set of groups G indexed by $g \in \{1, 2, \ldots, m\}$. Each arm has a representative in each group; we use P to denote an arm-group pair (a, g) and $\mathcal{P} := A \times G$. Each arm-group pair has its own distribution, where the mean reward of each pair $P = (a, g)$ is denoted by μ_P or $\mu_{a,g}$, for all $P \in \mathcal{P}$. We define a_g^* to be any highest mean reward arm in a group g, with corresponding mean reward as μ_g^* $:= \max_{a \in A} \mu_{a,g}$. These rewards are bounded between 0 and 1. We parameterize this bandit model by $\boldsymbol{\mu} = (\mu_P)_{P \in \mathcal{P}}$. At time t from $1, 2, \ldots, T$, we pick one arm from one group (pair $P_t = (a_t, g_t)$) to get a random reward X_t, where X_t is a a Bernoulli random variable with mean μ_{P_t}.

We want to have a way of picking arms from groups dynamically so that we can determine which arm minimizes the maximum (with respect to the groups) disappointment with high probability, while sampling from the arms as little as possible. We develop an algorithm with the following information in a similar setup to [Garivier et al., 2016], which is restated here. We adopt a sequential learning strategy (P_t, τ, a^*). We define $\mathcal{F}_t = \sigma(X_1, X_2, ...X_t)$ as a sigma field created from all previous

information of the observations up to time t, and we adopt a strategy with:

- A sampling rule $P_t \in \mathcal{P}$ where P_t is $\mathcal{F}_{t-1}$-measurable,
- A stopping condition τ that decides when the algorithm will finish, measurable with respect to $\mathcal{F}_t$,
- A final guess $\hat{a}$ for a^*.

We define the disappointment of arm a in group g as

$$D_{a,g} := \mu_g^* - \mu_{a,g}.$$

If arm a_g^* was picked as our best arm, then people in group g would suffer 0 disappointment because they got their best choice. In addition, we define the cost of disappointment for an arm (C_a) to be the maximum disappointment of any group under arm a, or

$$C_a := \max_{g \in G} D_{a,g},$$

where group g_a^* is a group that witnesses the maximum disappointment under arm a. Our goal is to find the arm that minimizes the cost of disappointment. We therefore define the best arm as

$$a^* := \arg\min_{a \in A} \max_{g \in G} D_{a,g} = \arg\min_{a \in A} C_a.$$

To simplify notation, we order the arms such that $C_1 < C_2 \leq C_3 \leq \cdots \leq C_k$, making $a^* =$ arm 1. The goal is to be able to pick a^* with high confidence $1-\delta$ while sampling from as few arm-group pairs as possible. In other words, the strategy chosen should be δ-PAC.

One common stopping condition for best arm identification problems uses confidence intervals, where when working with costs, we want to stop when the upper

confidence bound of the lowest cost arm is smaller than all other arms. This technique was first used in [Kalyanakrishnan et al., 2012], and will be used for our stopping condition here. To make a confidence interval surrounding the expected disappointment level, we have to use the difference between the mean reward of two arms, which means that we need to approximate both the arm-group pairs that minimize the disappointment any group suffers and the arms which have the highest expected reward in those groups. If we form a confidence interval around each of the cost values, we can form a stopping condition. Our stopping condition will be when the confidence intervals around the cost of disappointment of our best arm C_1 no longer overlap with the confidence interval of our second best arms cost of disappoint, C_2. In other words, when upper confidence bound on C_1 is lower than the lower confidence interval on C_2, we have reached our stopping condition.

The first attempt may be to try a naive algorithm where we would pull every arm in every group equally. This will be optimal in the two arm two group setting, as we should be pulling the best arms in each group and the groups that maximize disappointment for a given arm. In this case, either one arm has the highest reward for both groups, in which case the best arm is obvious, or one arm is the best for one group and the worst for the other. In this case, we would need to pull all 4 arms to get an accurate estimate of C_1 and C_2, meaning we will pull all 4 arms an equal number of times in the best case scenario.

However, in a case with either more than two arms or more than two groups, we easily show that we can do better than the naive strategy. For an example, in Figure 1.1 we see that arm 1 in group 2 does not maximize disappointment for arm 1, and is not the maximum reward arm in group 2, so an ideal algorithm should be able to quickly realize arm 1 in group 2 does not need to be pulled often to distinguish the difference between the C_a values. In the next chapter an algorithm is presented that will be able to take advantage of these benefits.

Chapter 4

Algorithms

To find the best arm, we need to be able to estimate certain $D_{a,g}$ gaps effectively, while minimizing the number of pulls. We also want to make a point that here our goal is to minimize disappointment, instead of the standard best arm framework of maximizing reward. Our approach is to make pulls in two different situations: pulling to find the best arm, and pulling to estimate the highest reward arms in certain groups.

To deal with estimating the mean reward of the highest reward arm in groups, we can use the Incremental Exponential Gap (INC-EG) algorithm introduced in Chapter 2. We can call INC-EG for a new round r on one group g whenever we need a more precise measurement of the highest mean reward arm from group g. We define $r_g(t)$ be the number of rounds we have called INC-EG for group g by the end of time t. This subroutine outputs $\hat{\nu}_g(r)$, which is our estimate of the reward of the best arm in group g after r rounds. We can use this subroutine to get a confidence interval $[L_g(r),\ U_g(r)]$ surrounding μ_g^*, where

$$U_g(r) := \hat{\nu}_g(r) + \epsilon_g(r) \text{ and } L_g(r) := \hat{\nu}_g(r) - \epsilon_g(r), \tag{4.1}$$

and where $\epsilon_g(r)$ is the precision of the highest reward arm in group g after r rounds of INC-EG. We call this subroutine for each group, which gives an estimate of the

highest reward arm in each group. We get a new precision $\epsilon_g(r+1) = \epsilon_g(r)/2$ if we call the subroutine for round $r+1$ in group g, as the subroutine will be sampling an arm an order of $\log(1/\epsilon_g(r))$ times.

INC-EG allows us to have an algorithm that only focuses on finding the best arm, and when we need a more accurate estimate of the highest mean reward arm in a group, we can just call INC-EG for another round. We use the lower upper confidence bound technique (LUCB [Kalyanakrishnan et al., 2012]) for finding the best arm. When the upper confidence bound for C_1 is less than the lower confidence bound for C_2, we can conclude with high probability arm 1 is the best. Combining the LUCB technique with strategic calls to the subroutine is the general idea behind the MD-LUCB Algorithm.

4.1 Minimax Disappointment Lower Upper Confidence Bounds (MD-LUCB)

The MD-LUCB algorithm builds on the M-LUCB algorithm in [Garivier et al., 2016]. MD-LUCB needs to also estimate μ_g^*, which it does by periodically calling INC-EG as a subroutine. MD-LUCB also needs to control how often it calls INC-EG so that we are not over sampling to find the highest mean reward arms when not necessary. We use s to enumerate epochs, where $s \in \mathbb{N}$ and $\mathbb{N} := \mathbb{N}_0$. Epoch zero is the initial state where we pull each arm-group pair once, and all subsequent epochs s will be referring to each iteration of a while loop.

First, we will sketch the outline of how MD-LUCB works. MD-LUCB, like LUCB and M-LUCB, is going to find two arms to pull each epoch, and pull them. Then it will check a difference of upper and lower confidence bounds between them, and if

there is no overlap between them it will halt. The extra step added to MD-LUCB is that after finding and pulling the two arms, MD-LUCB checks the confidence radius around those arms, and the confidence radius around the mean reward of the best arm for each arm's respective group. If the confidence radius is ever smaller for one of the arms we are pulling compared to the confidence radius around μ_g^* from the same group as we are pulling from, then we call INC-EG. This can be seen in Algorithm 5.

Algorithm 5: MD-LUCB

Input: $\delta > 0$, A, G
Set $t = 1$
Set $\delta_E = \delta/2$, $\delta_H = \delta/2$, $\delta_g = \delta_H/|G|$
Set $r_g(t) = 1$, $\epsilon_g(r_g(t)) = 1/8$ for all $g \in G$
Set $\hat{\nu}_g(r_g(t)), S_g(r_g(t)) = \text{INC-EG}(\delta_g, r_g(t), g, A)$ for all $g \in G$
Pull every pair $P \in \mathcal{P}$ one time
Set $s = 1$, $t_0 = t - 1$
while $U_{D_{P_1(t_{s-1})}}(t_{s-1}) \geq L_{D_{P_2(t_{s-1})}}(t_{s-1})$ **do**
    Update $P_1(t_{s-1})$, $P_2(t_{s-1})$
    Pull pair $P_1(t_{s-1})$
    Pull pair $P_2(t_{s-1})$
    Set $g1 = g_{a_1(t)}(t)$, $g2 = g_{a_2(t)}(t)$
    if $\epsilon_{P_1(t)}(t) < \epsilon_{g1}(r_{g1}(t))$ **then**
        Set $\hat{\nu}_{g1}(r_{g1}(t)), S_{g1}(r_{g1}(t)) = \text{INC-EG}(\delta_g, r_{g1}(t) + 1, g1, S_{g1}(r_{g1}(t)))$
        Set $r_{g1}(t) = r_{g1}(t) + 1$, $\epsilon_{g1}(r_{g1}(t)) = \epsilon_{g1}(r_{g1}(t))/2$
    if $\epsilon_{P_2(t)}(t) < \epsilon_{g2}(r_{g2}(t))$ **then**
        Set $\hat{\nu}_{g2}(r_{g2}(t)), S_{g2}(r_{g2}(t)) = \text{INC-EG}(\delta_g, r_{g2}(t) + 1, g2, S_{g2}(r_{g2}(t)))$
        Set $r_{g2}(t) = r_{g2}(t) + 1$, $\epsilon_{g2}(r_{g2}(t)) = \epsilon_{g2}(r_{g2}(t))/2$
    Set $t_s = t - 1$
    Set $s = s + 1$
Return arm $a \in P_1(t_{s-1}$

To figure out which arms we are going to pull each epoch, we need to introduce a few things. We define a new event $E_{MD}(t)$, which happens when we are pulling with the objective of determining the best arm, instead of determining the highest reward value from groups. In other words, these are the pulls happening directly in MD-LUCB, not in INC-EG. We define $N_{a,g}(t) := \sum_{k=1}^{t} \mathbf{1}_{(P_k=(a,g))}\mathbf{1}_{(E_{MD}(t))}$ as the number of samples of pair (a, g) from MD-LUCB by the end of time t, $N(t) := \sum_{a,g \in |A| \times |G|} N_{a,g}(t)$ as

the total number of samples from MD-LUCB for all pairs by the end of time t, and $\hat{\mu}_{a,g}(t) = \sum_{k=1}^{t} X_k \mathbf{1}_{(P_k=(a,g))} \mathbf{1}_{(E_{MD}(t))} / N_{a,g}(t)$ as the empirical mean reward of pulls from MD-LUCB for pair (a, g). We use a Hoeffding style bound to get a confidence radius around $\hat{\mu}_{a,g}(t)$, defined as

$$\epsilon_{a,g}(t) := \sqrt{\frac{\beta(N(t), \delta_E)}{2N_{a,g}(t)}}, \tag{4.2}$$

where $\beta(N(t), \delta_E)$ is some exploration rate based on the number of time steps where we have pulled from MD-LUCB $N(t)$ and the failure probability δ_E. This leads to the confidence interval $[U_{a,g}(t), L_{a,g}(t)]$ surrounding $\mu_{a,g}$, where

$$U_{a,g}(t) := \hat{\mu}_{a,g}(t) + \epsilon_{a,g}(t) \text{ and } L_{a,g}(t) := \hat{\mu}_{a,g}(t) - \epsilon_{a,g}(t). \tag{4.3}$$

We define empirical estimates of $D_{a,g}$ as

$$\hat{D}_{a,g}(t) := \hat{\nu}_g(r_g(t)) - \hat{\mu}_{a,g}(t),$$

and the confidence interval surrounding $D_{a,g}$ to be $[L_{D_{a,g}}(t), U_{D_{a,g}}(t)]$, where

$$U_{D_{a,g}}(t) := U_g(r_g(t)) - L_{a,g}(t) \text{ and } L_{D_{a,g}}(t) := L_g(r_g(t)) - U_{a,g}(t).$$

Now, denoting group

$$g_a(t) := \arg\max_{g \in G} U_{D_{a,g}}(t),$$

we define empirical estimates of C_a as

$$\hat{C}_a(t) := \hat{D}_{a,g_a(t)},$$

and confidence interval surrounding C_a to be $[L_{C_a}(t), U_{C_a}(t)]$, where

$$U_{C_a}(t) := U_{D_{a,g_a(t)}}(t) \text{ and } L_{C_a}(t) := L_{D_{a,g_a(t)}}(t). \tag{4.4}$$

We define time step t_s to be the last time step of epoch s, with $t_{s=0}$ being the time right before we enter the loop in Algorithm 5 for the first time. MD-LUCB finds two arms to pull each epoch $s > 0$:

$$a_1(t_{s-1}) := \arg\min_{a \in A} \hat{C}_a(t_{s-1}) \text{ and } \; a_2(t_{s-1}) := \arg \min_{a \in A, a \neq a_1(t_{s-1})} L_{C_a}(t_{s-1}).$$

With these two arms and their estimated best response groups $g_{a_1(t_{s-1})}(t_{s-1})$ and $g_{a_2(t_{s-1})}(t_{s-1})$, we have $P_1(t_{s-1}) = (a_1(t_{s-1}), g_{a_1(t_{s-1})}(t_{s-1}))$, as the pair we think of with our best arm in epoch s, and $P_2(t_{s-1}) = (a_2(t_{s-1}), g_{a_2(t_{s-1})}(t_{s-1}))$ as the pair with the second best arm in epoch s. These two pairs of arms are the pairs that MD-LUCB pulls in epoch s. We use time t_{s-1} to find our pairs $P_1(t_{s-1})$ and $P_2(t_{s-1})$ to pull for epoch s. This is also the time step where we check our stopping condition.

We think of the pair $P_1(t_{s-1})$ as the current best arm in epoch $s > 0$, and $P_2(t_{s-1})$ as our optimistic view on the best arm in epoch $s > 0$. We pull $P_1(t_{s-1})$ to shrink the confidence interval of the current best arm, and we pull $P_2(t_{s-1})$ to guarantee that the confidence interval of all of the C_a terms are shrinking. Pulling arms in this way for any epoch $s > 0$, we know we are lowering the pessimistic cost of the best arm, and raising the optimistic cost of the second best arm.

Now that we have defined $P_1(t_s)$ and $P_2(t_s)$, we can can describe the full process of MD-LUCB. In lines 1-7, we run the first round of the INC-EG algorithm on each group, which gives an initial approximation of the highest mean reward in each group, and pulls each arm-group pair one time to form confidence intervals around $\mu_{a,g}$. We call this epoch 0. In line 9-11 we find the new pairs P_1 and P_2 for this epoch and pull them. Line 12 is used to simplify notation inside the algorithm, where $g1$ is the group in P_1 at time t_{s-1} and $g2$ is the group in P_2 at time t_{s-1}. Lines 13-15 check to see if the confidence interval surround μ^*_g is smaller than the confidence interval around $\mu_{P_1(t_{s-1})}$, and if not, we call INC-EG on the next round. Lines 16-18 check the same condition as

13-15, but now for the second pair $P_2(t_{s-1})$ and group $g2$. Line 19-20 update our epoch and the final time step of that epoch. If the upper confidence bound of $P_1(t_{s-1})$ is less than the lower confidence bound of $P_2(t_{s-1})$, then we stop the algorithm and return the arm that is a part of pair $P_1(t_{s-1})$; if is it larger, we go to the next epoch of MD-LUCB.

If $\epsilon_{P_1(t)}(t)$ or $\epsilon_{P_2(t)}(t)$ is ever smaller than the corresponding $\epsilon_g(r_g(t))$ value used to get the disappointment, we call the INC-EG algorithm on group g for round $r_g(t)+1$. This way we always have that for every arm a for a given group g, we have a tighter confidence interval around μ_g^* than $\mu_{a,g}$, which is important in the analysis of MD-LUCB.

4.2 Incremental Exponential Gap (INC-EG)

Algorithm 6: Incremental Exponential Gap

1 **Input:** $\delta_g > 0, r > 0, g \in G, S_g(r) \subseteq A$
2 Set $\epsilon_g(r) = 2^{-r}/4, \delta_{r,g} = \delta_g/50r^2$
3 Sample each arm $a \in S_g(r)$ a total of $t_r = (1/\epsilon_g(r)^2)\ln(2/\delta_{r,g})$ times to get estimated means $\hat{\nu}_a$
4 Invoke $a_g(r) = \text{MedianElimination}(S_g(r), \epsilon_g(r)/4, \delta_{r,g})$.
5 Set $S_g(r+1) = S_g(r) \setminus \{a \in S_g(r) : \hat{\mu}_{a_{r,g}} < \hat{\mu}_{a,g} - \epsilon_g(r)/2\ \}$
6 **Return** $\hat{\nu}_{a_g(r)}$, $S_g(r+1)$

The Incremental Exponential Gap (INC-EG) algorithm was first referenced in Chapter 2, and it is based off of the Exponential Gap (EG) algorithm [Karnin et al., 2013]. A version of INC-EG with groups that MD-LUCB calls is given as Algorithm 6, and from here onward, when we refer to INC-EG, we will be referring to Algorithm 6 instead of the Algorithm 3. The EG algorithm takes in a set of arms and outputs the arm with highest reward, whereas INC-EG takes in a set of active arms and outputs an estimate of the highest mean reward. We need this change as we use INC-EG to help estimate disappointment levels, which are based off of the estimate of the highest mean reward, not which arm has the highest mean reward.

We split the failure probability of δ_H equally among each group, so $\delta_g = \delta_H/|G|$. Failure probability δ_g is the probability of any call of INC-EG from group g outputting an empirical reward not within $\epsilon_g(r)$ of the true highest reward in group g. We define $S_g(r)$ as the set of remaining arms that we are still considering to be the highest reward arm after r calls to INC-EG for group g.

INC-EG is a round based algorithm that is called for a single group for a single round. Each round r of INC-EG takes the remaining arms $S_g(r)$ in g that with high probability can still be the best arm in that group and pulls them until we can say that with high probability the highest mean empirical reward from INC-EG $\hat{\nu}_g(r_g(t))$, is within $\epsilon_g(r_g(t))$ of the actual highest reward μ_g^*. At the end of each round it eliminates arms that are not within $\epsilon_g(r_g(t))$ of best arm at that epoch. Arm $a_g(r)$ is found by running median elimination on the remaining arms, and the true reward of that arm is $\mu_{a_g(r),g}$.

Chapter 5

Sample Complexity Analysis

Before commencing the analysis, it will be useful to introduce an event $E_{EG}(t)$, which, together with its disjoint event $E_{MD}(t)$, specifies which algorithm (INC-EG or MD-LUCB) made pulls at each time t. Event $E_{MD}(t)$ happens when we are pulling to find the best arm at time t, and event $E_{EG}(t)$ happens when we are pulling to find the highest reward arms from the groups at time t.

This sample complexities of INC-EG and MD-LUCB is split to make analysis easier, as counting the number of epochs in MD-LUCB to find out how many pulls we have does not work when we have a variable number of pulls happening in each epoch. For example, in some epoch s, we can pull only the pairs $P_1(t_{s-1})$ and $P_2(t_{s-1})$, and in another, we can pull those two pairs plus however many pulls happens inside of a call to INC-EG at that time, which is dependant on how many rounds INC-EG has been called already on a given group. We can also potentially have two different calls to INC-EG happen in the same epoch, one for each pair. Therefore splitting the pulls between the two different situations simplifies the problem greatly. Also, by decoupling the analysis of the two algorithms, we can treat the INC-EG algorithm as a black-box inside of MD-LUCB, and it can be replaced by a more efficient best arm identification algorithm in the future.

To analyze the algorithms, we are going to first show that the confidence intervals surrounding the disappointment levels and their costs contain their true values with high probability. For all of the following analysis, we define the event

$$\mathcal{H}_{r,g} = \mu_{a^*_g,g} \in [\ L_g(r), U_g(r)],$$

along with events $\mathcal{H}_g = \cap_{r\in\mathbb{N}}\ \mathcal{H}_{r,g}$, and $\mathcal{H} = \cap_{g\in G}\ \mathcal{H}_g$. We also define event

$$\mathcal{E}_s = \bigcap_{P\in\mathcal{P}} (\mu_P \in [L_{a,g}(t_s), U_{a,g}(t_s)]),$$

and event $\mathcal{E} = \cap_{s\in\mathbb{N}}\ \mathcal{E}_s$. Event $\mathcal{H}$ holds when the highest mean reward for a group is contained within the confidence interval in (4.1) for all groups and rounds, and event $\mathcal{E}$ holds when all arm-group pairs are within their confidence intervals in (4.3) for all time steps t_s. We define the failure probability of event $\mathcal{H}$ with δ_H, and the failure probability of event $\mathcal{E}$ is δ_E. If events $\mathcal{E}$ and $\mathcal{H}$ hold true, then we show that our confidence intervals around $D_{a,g}$ and C_a terms hold for the duration of the algorithm.

5.1 $\delta-$PAC Guarantee

To guarantee we pick the best arm with probability at least $1-\delta$, we split the failure probability δ equally between δ_E and δ_H, where δ_E controls the confidence intervals of $\mu_{a,g}$ and δ_H controls the confidence intervals around μ^*_g. If both events $\mathcal{E}$ and $\mathcal{H}$ happen, then we know that the algorithm will pick the best arm. We introduce Lemma 4 to show that we can create a confidence interval surrounding μ^*_g using the output of INC-EG for some round r and group g.

Lemma 4. *For any round r and any group g, with probability at least $1-2\delta_{r,g}$, INC-EG outputs $\hat{\nu}_g(r)$ such that $\mu_{a^*_g,g} \in [L_g(r), U_g(r)]$.*

Proof. First, we note that the empirical reward outputted from INC-EG in round r

from group g is made up of pulls from a single arm, so we define that the expected mean reward from $\hat{\nu}_g(r)$ as ν_g for round r in group g.

We want to show that $\mu_{a_g^*} \leq U_g(r)$ in round r. INC-EG calls the Median Elimination algorithm with arm set $S_g(r)$, failure probability $\delta_{r,g}$, and precision $\epsilon_g(r)/2$. From Theorem 10 in [Even-Dar et al., 2006], we know that the Median Elimination algorithm will output an arm with a reward that is within $\epsilon_g(r)/2$ of $\mu_{a_g^*,g}$ with probability at least $1 - \delta_{r,g}$. We upper bound $\mu_{a_g^*,g}$ with

$$\begin{aligned}
\mu_{a_g^*} &\leq \nu_g + \epsilon_g(r)/2 \\
&\leq \hat{\nu}_g(r)) + \epsilon_g(r) \text{ from Hoeffding's Inequality} \\
&= U_g(r).
\end{aligned}$$

We now show that $\mu_{a_g^*} \geq L_g(r)$, which is true because

$$\mu_{a_g^*} \geq \nu_r \geq L_g(r).$$

Now, it is clear that $\mu_{a_g^*,g} \in [L_g(r), U_g(r)]$ with probability at least $1 - 2\delta_{r,g}$. □

We can use Lemma 4 to show that event $\mathcal{H}$ holds with probability at least $1 - \delta_H$. To do so, we set our failure probability for a single group to be $\delta_{r,g} = \delta_g/(50r^2)$ like in [Karnin et al., 2013]. Then using Lemma 4, we know that event $\mathcal{H}_{r,g}$ fails with probability at most $2\delta_{r,g}$ for some round r. To bound the failure probability of event $\mathcal{H}_g$ we can use a union bound over the rounds of event $H_{r,g}$ failing as event $\mathcal{H}_g$ occurs if $\mathcal{H}_{r,g}$ happens for every round. We note that $\sum_{r=1}^{\infty} 2\delta_{r,g} \leq \delta_g$, and using this we get that event $\mathcal{H}_g$ happens with probability at least $1 - \delta_g$. Now we want to get an upper bound on $\mathcal{H}$ failing, which we know does not happen if event $\mathcal{H}_g$ occurs for all groups. We take a union bound over groups of event $\mathcal{H}_g$ failing, which has a failure probability of δ_g. This gives us an upper bound on the failure probability of event $\mathcal{H}$

of $\sum_{g=1}^{|G|} \delta_g = \delta_H$. Therefore event $\mathcal{H}$ holds with probability at least $1 - \delta_H$.

To show event $\mathcal{E}$ holds, we set the exploration rate such that the confidence intervals surrounding $\mu_{a,g}$ hold for all arms a and groups g, and for all times t_s, where $s \in \mathbb{N}$, with a probability of at least $1 - \delta_E$. The exploration rates used are in equations (5.3) and (5.4), which are included in Lemmas 10 and 11. We leave these lemmas and their proofs for later in this chapter, as they also include bounds on the number of samples we need from picking the different exploration rates.

If $\mathcal{E}$ and $\mathcal{H}$ hold, then Lemmas 5 and 6 below imply that the confidence intervals surrounding $D_{a,g}$ and C_a hold for all time steps t_s, where $s \in \mathbb{N}$.

Lemma 5. *Assuming events $\mathcal{H}$ and $\mathcal{E}_s$ hold, then $D_{a,g} \in [L_{D_{a,g}}(t_s), U_{D_{a,g}}(t_s)]$.*

Proof. Disappointment $\hat{D}_{a,g}$ is made up of $\hat{\mu}_{a,g}(t_s)$ and $\hat{\nu}_g(r_g(t_s))$. Our first step will be showing that these values are good estimators for their true value counterparts in $D_{a,g}$, which are $\mu_{a,g}$ and $\mu_{a^*_g,g}$.

We know $\mu_{a,g} \in [L_{a,g}(t_s), U_{a,g}(t_s)]$ from $\mathcal{E}_s$. We also know from Lemma 4 and event $\mathcal{H}$ that $\mu_{a^*_g,g} \in [L_{r_g(t)}(t_s), U_{r_g(t_s)}(t_s)]$. To upper bound $D_{a,g}$, we use these facts to get

$$\begin{aligned} D_{a,g} &= \mu_{a^*,g} - \mu_{a,g} \\ &\leq U_{r_g(t_s)}(t_s) - L_{a,g}(t_s) \\ &= U_{D_{a,g}}(t_s). \end{aligned}$$

Now that we have an upper bound on $D_{a,g}$, we can use those same facts for our lower bound to get

$$\begin{aligned} D_{a,g} &= \mu_{a^*,g} - \mu_{a,g} \\ &\geq L_{r_g(t_s)}(t_s) - U_{a,g}(t_s) \\ &= L_{D_{a,g}}(t_s). \end{aligned}$$

Therefore $D_{a,g} \in [L_{D_{a,g}}(t_s), U_{D_{a,g}}(t_s)]$. □

Lemma 6. *Assuming $\mathcal{H}$ and $\mathcal{E}_s$ hold, then $C_a \in [L_{C_a}(t_s), U_{C_a}(t_s)]$.*

Proof. For this proof we want to show that the confidence interval around the disappointment level for arm a from group $g_a(t_s)$ will contain the actual disappointment level for that same arm but from group g_a^*. To prove this, C_a will be displayed as D_{a,g_a^*}, and $\hat{C}_a$ as $\hat{D}_{a,g_a(t_s)}$ so that we can show that even if g_a^* and $g_a(t_s)$ are different groups, the confidence interval of D_{a,g_a^*} will always fall completely inside of the interval of $D_{a,g_a(t_s)}$.

First, we get an upper bound on D_{a,g_a^*} with

$$\begin{aligned} D_{a,g_a^*} &= \max_{j \in G} D_{a,j} \\ &\leq \max_{j \in G} U_{D_{a,j}}(t_s) \\ &= U_{D_{a,g_a(t_s)}}(t_s). \end{aligned}$$

Now we assume that $D_{a,g_a^*} < L_{D_{a,g_a(t_s)}}(t_s)$ and $g_a(t_s) \neq g_a^*$. This would then mean that $D_{a,g_a^*} < D_{a,g_a(t_s)}$, which is impossible, as by definition $g_a^* = \arg\max_{j \in G} D_{a,j}$. If $g_a(t_s) = g_a^*$ then we know $D_{a,g_a^*} \geq L_{D_{a,g_a^*}}(t_s)$ when events $\mathcal{E}_s$ and $\mathcal{H}$ hold. Therefore $D_{a,g_a^*} \geq L_{D_{a,g_a(t_s)}}(t_s)$. So it follows that $D_{a,g_a^*} \in [L_{D_{a,g_a(t_s)}}(t_s), U_{D_{a,g_a(t_s)}}(t_s)]$, which is the same as $C_a \in [L_{C_a}(t_s), U_{C_a}(t_s)]$. □

Now that we know that we have confidence intervals that hold for $D_{a,g}$ and C_a, we look at the sample complexity of the algorithm. The stopping time of MD-LUCB with failure probability δ is defined as τ_δ. For the sample complexity at each time step t, we pull one arm from one group, and exactly one of $E_{MD}(t)$ or $E_{EG}(t)$ happens. Let T be a deterministic time. We are going to The total sample complexity of the algorithm is upper bounded by

$$\min(\tau_\delta, T) = \sum_{t=1}^{T} \mathbf{1}_{(\tau_\delta < t)} = \sum_{t=1}^{T} \mathbf{1}_{(\tau_\delta < t)} \mathbf{1}_{(E_{MD}(t))} + \sum_{t=1}^{T} \mathbf{1}_{(\tau_\delta < t)} \mathbf{1}_{(E_{EG}(t))}.$$

If we show that the algorithm stops at some time $T > \tau_\delta$, we can bound the sample complexity. We split up the sample complexity between pulls to find the best arm (from MD-LUCB), and pulls to get more precision on the highest reward arms in groups (from INC-EG). We define $N(t)$ and $M(t)$ to be counters of how many times we have pulled from MD-LUCB and INC-EG respectively by time t, with the total pulls from each at the end of our algorithm being $N = N(\tau_\delta)$ and $M = M(\tau_\delta)$, and the total sample complexity being $\tau_\delta = N + M$.

5.2 MD-LUCB Sample Complexity

We decompose the number of pulls from MD-LUCB as $N = \sum_{(a,g)\in\mathcal{P}} N_{a,g}$, where $N_{a,g}$ counts the number of times pair (a, g) is pulled from MD-LUCB. We bound N in Theorem 7. This bound does not depend on $\epsilon_g(r)$, as we know $\epsilon_g(r_g(t_s)) < \epsilon_{a,g}(t_s)$ is true for every (a, g) pair and for all times t_s from the construction of the algorithm. This means we can upper bound the confidence radius of $D_{a,g}$ by 2 $\epsilon_{a,g}(t)$, and get an upper bound on N with failure probability $1 - \delta_E$.

Theorem 7. *Let*

$$H(\boldsymbol{\mu}) = \sum_{(a,g)\in\mathcal{P}} \frac{1}{\max((D_{a,g} - C_a)^2, (C_a - \frac{C_1+C_2}{2})^2)}.$$

On the events $\mathcal{H}$ and $\mathcal{E}$, MD-LUCB will return the best arm in a total number of samples under event $E_{MD}(t)$ upper bounded by

$$S(\boldsymbol{\mu}, \delta_E) = \inf\{s \in \mathbb{N} : 16H(\boldsymbol{\mu})\beta(N(t_s), \delta_E) < t_s\}.$$

We interpret the complexity term H as making sure we have enough pulls for an arm-group pair (a, g) such that we either

- Have pulled enough to discriminate between the disappointment $D_{a,g}$ of an arm from its cost of disappointment C_a, or

- Have pulled enough to discriminate between the cost of disappointment from that arm and the cost of disappointment of the average between the two best arms C_1 and C_2. This term is an artifact of our proof as there is one arm-group pair that will have a disappointment value equal to C_1, meaning if we used the best arm's cost of disappointment C_1, that pair would have infinite sample complexity when defined this way.

Intuitively, we need to pull enough times to to discriminate between $D_{a,g}$ and C_1, as if we do not, then we may end up picking a different arm-group pair for our value of $\hat{C}_a$, which has less disappointment than C_a does. This could change which arm is considered to be the best, and we would pick the incorrect arm as our best arm.

Intuitively, we need to pull enough times to discriminate between C_a and C_1 (in this case $(C_1 + C_2)/2$ so we can determine with high probability which of our arms in the best choice. If we do not pull this many times for a arm-group pair then we cannot say with high probability which arm is the best.

In the exploration rate there is a cost of at least $\log(1/\delta)$, so we can think of each arm-group pair as costing $H \cdot \log(1/\delta)$.

To prove Theorem 7, we first need to introduce Lemmas 8 and 9. The proofs of these lemmas are in Appendix A.

Lemma 8. *Let $c \in [C_1, C_2]$, and $s \in \mathbb{N}$. On $\mathcal{E}_s$ and $\mathcal{H}$, if $\rho_{\delta_E} > s$, there exists $P \in \{P_1(t_s), P_2(t_s)\}$ such that*

$$c \in [L_{D_P}(t_s), U_{D_P}(t_s)].$$

Lemma 9. *Let $c \in [C_1, C_2]$, and $s \in \mathbb{N}$.*

On $\mathcal{E}_s$ and $\mathcal{H}$, for every $(a, g) \in \{P_1(t_s), P_2(t_s)\}$, if $c \in [L_{D_{a,g}}(t_s), U_{D_{a,g}}(t_s)]$, then $N_{a,g}(t_s) \leq \min(\frac{1}{(C_a - D_{a,g})^2}, \frac{1}{(C_a - c)^2}) 8\beta(N(t_s), \delta_E)$.

Define for every pair $P \in \mathcal{P}$ the constant

$$c_P = \frac{1}{\max((D_{a,g} - C_a)^2, (C_a - \frac{C_1+C_2}{2})^2)},$$

where $\sum_{P\in\mathcal{P}} c_P = H(\boldsymbol{\mu})$. Combining the two lemmas above with $c = \frac{C_1+C_2}{2}$ (so that $c \in [C_1, C_2]$), we get the following statement:

$$\mathcal{E}_s \cap (\rho_{\delta_E} > s) \Rightarrow \exists P \in \{P_1(t_s), P_2(t_s)\} \text{ such that } N_P \leq 8c_P\beta(N(t_s), \delta_E).$$

Now we can prove Theorem 7.

Proof of Theorem 7

Proof. We want an upper bound on the number of pulls from MD-LUCB, and to do that we will upper bound the number of epochs MD-LUCB has, as there are always exactly 2 pulls happening in each epoch from MD-LUCB. To this we introduce our stopping epoch ρ_{δ_E}, which is the epoch MD-LUCB halts at with risk level δ_E. Let S be some deterministic number of epochs, so we know the number of epochs is going to be $\min(\rho_{\delta_E}, S)$. Every epoch we check our stopping condition at time t_s, which is at the end of each epoch. Now, we want a bound on the number of epochs we have before we reach our stopping condition, which leads to

$$\min(\rho_{\delta_E}, S) = \sum_{s=0}^{S} \mathbf{1}_{(\rho_{\delta_E} < s)}.$$

On the event $\mathcal{H}$ and $\mathcal{E}_s$, we know from Lemma 8 that there is always at least one pair $P_1(t_s)$ or $P_2(t_s)$ that have a confidence interval around its disappointment contains C_1 and C_2. Combining this fact with Lemma 9, we know that the number of pulls from

at least one of $P_1(t_s)$ or $P_2(t_s)$ is upper bounded by $8c_P\beta(N(t_s),\delta_E)$. This leads to

$$\min(\rho_{\delta_E}, S) = \sum_{s=0}^{S} \mathbf{1}_{(\rho_{\delta_E}<s)} = \sum_{\substack{s\leq S \\ s\in\mathbb{N}}} \mathbf{1}_{(\exists P\in\{P_1(t_s),P_2(t_s)\}:N_P(t_s)\leq 8c_P\beta(N(t_s),\delta_E))}. \tag{5.1}$$

This is upper bounded by a union bound of the pairs picked at times t_s+1 and t_s+2, where t_s is the last time step before a new epoch $s+1$. The exploration rate is always larger by the end of time t_S compared to t_s, and so we upper bound (5.1) by the larger exploration rate to get

$$2\sum_{\substack{s\leq S \\ s\in\mathbb{N}}} \sum_{P\in\mathcal{P}} \mathbf{1}_{(P_{(t_s+1)}=P)\cap(P_{(t_s+2)}=P)} \mathbf{1}_{N_P(t_s)\leq 8c_P\beta(N(t_S),\delta_E))}. \tag{5.2}$$

From here, we sum up over all of the S epochs and upper bound (5.2) as

$$8\sum_{P\in\mathcal{P}} c_P\beta(N(t_S),\delta_E) = 8H(\boldsymbol{\mu})\beta(N(t_S),\delta_E).$$

For any S such that $8H(\boldsymbol{\mu})\beta(N(t_S),\delta_E) < S$, one has $\min(\rho_{\delta_E}, S) < S$, which implies $\rho_{\delta_E} < S$. Now if we multiply by two, we can get an upper bound on the sample complexity of the problem, as $N(t_s) = 2s$ for all s. Therefore $N = 2\rho_{\delta_E} \leq S(\boldsymbol{\mu},\delta_E)$, where

$$S(\boldsymbol{\mu},\delta_E) = \inf(s\in\mathbb{N} : 16H(\boldsymbol{\mu})\beta(N(t_s),\delta_E) < N(t_s)).$$

□

Complexity term $H(\boldsymbol{\mu})$ can be thought of as each pair (a,g) is pulled enough to discriminate between $D_{a,g}$ and C_a, or pulled enough to discriminate between C_a from $\frac{C_1+C_2}{2}$. Now we want to pick an exploration rate that minimizes $S(\boldsymbol{\mu},\delta_E)$ but still makes event $\mathcal{E}$ happen. One method of doing this uses Hoeffding's inequality to bound the probability that each arm a in each group g is within its confidence interval for a time t_s. We can then take that bounded probability and make sure every arm is within its confidence interval at all times t_s, by making sure the confidence intervals

hold at the end of every epoch s. This gives Lemma 10.

Lemma 10. *If $H(\boldsymbol{\mu}) > 3/128$, then with probability at least $1 - \delta_E$, the MD-LUCB algorithm using exploration rate*

$$\beta(N(t), \delta_E) = \log\left(\frac{2|A||G|(N(t)+1)^2}{\delta_E}\right), \tag{5.3}$$

returns the best arm with the number of pulls from MD-LUCB upper bounded as

$$N \leq \max\left(32H(\boldsymbol{\mu})\log\left(\frac{2|A||G|}{\delta_E}\right),\ 384H(\boldsymbol{\mu})\log(128H(\boldsymbol{\mu}))\right).$$

However, this bound is not the tightest possible. Using Lemma 7 from [Kaufmann et al., 2016], we can use a smaller exploration rate to get a smaller sample complexity.

Lemma 11. *For $\delta < 0.049$ and $b > 6$, let the exploration rate be*

$$\beta(N(t), \delta) = \log(1/\delta) + b\log\log(1/\delta) + 3\log\log(eN(t)), \tag{5.4}$$

and let

$$f_b(\delta) = |A \times G|\sqrt{e}(\frac{\pi^2}{3})(\frac{\sqrt{\log(1/\delta) + b\log\log(1/\delta)}}{2\sqrt{2}} + 1)^3\frac{1}{\log(1/\delta)^b}.$$

Then with probability 1 - $f_b(\delta)\delta$, MD-LUCB returns the best arm with the number of pulls from MD-LUCB upper bounded as

$$N \leq 48H(\boldsymbol{\mu})\max(\log(1/\delta), b\log\log(1/\delta), 3\log\log(3e \cdot 48H(\boldsymbol{\mu})).$$

The proofs for Lemmas 10 and 11 are contained in Appendix A.

5.3 INC-EG Sample Complexity

The number of pulls from INC-EG will be based on Theorem 2, originally from [Karnin et al., 2013], which bounds the sample complexity of EG. We count the pulls from INC-EG using M, where $M = \sum_{(a,g)\in\mathcal{P}} M_{a,g}$, and $M_{a,g}$ is the number of pulls from INC-EG for pair (a, g). We get the following theorem for the number of pulls from INC-EG.

Theorem 12. *With probability at least $1 - \delta_H$, the number of pulls from INC-EG (event E_{EG}) will be*

$$M = \sum_{(a,g)\in\mathcal{P}} O\left(\frac{1}{(D_{a,g}^{\epsilon_g})^2} \log\left(\frac{1}{\delta_g}\log\left(\frac{1}{D_{a,g}^{\epsilon_g}}\right)\right)\right),$$

where $\delta_g = \delta_H/|G|$, $D_{a,g}^{\epsilon_g} = \max\{D_{a,g}, \epsilon_g\}$, and $\epsilon_g := \epsilon_g(r_g(\tau_\delta))$.

To prove Theorem 12, we first want to have a lower bound on how small the confidence radius $\epsilon_g(r)$ gets, which we define as ϵ_g. We interpret ϵ_g as how tight the confidence interval around μ_g^* will have to be at the stopping time τ_δ. We get a lower bound on the size of ϵ_g in Lemma 13.

Lemma 13. *For a given group g,*

$$\epsilon_g \geq \frac{1}{8}\min_{a\in A}\sqrt{\max\left((D_{a,g} - C_a)^2, \left(C_a - \frac{C_1 + C_2}{2}\right)^2\right)}.$$

Proof. By the construction of MD-LUCB, we only ever call another round of INC-EG for group g when $\epsilon_{P_1(t)}(t)$ or $\epsilon_{P_2(t)}(t)$ is smaller than the corresponding $\epsilon_g(r_g(t))$ value in at time t. In this case we call round $r_g(t) + 1$ of INC-EG for that group, and we get $\epsilon_g(r_g(t) + 1) = \epsilon_g(r_g(t))/2$. We define ϵ_g to be the value of $\epsilon_g(r_g(\tau_\delta))$, where τ_δ is our stopping time. We know that $\tau_\delta = t_{\rho_{\delta_E}}$, as ρ_{δ_E} is our stopping epoch. This means ϵ_g is the level of precision we needed from group g by the end of execution of MD-LUCB. We have a bound on the number of pulls of each arm-group pair from MD-LUCB

in Lemma 9, and so we know ϵ_g can be lower bounded as half the smallest value of $\epsilon_{a,g}(\tau_\delta)$ for a group g, which is

$$\epsilon_g = \epsilon_g(r_g(\tau_\delta)).$$

We know by the construction of the algorithm that ϵ_g will be at least half the size of the smallest value of $\epsilon_{a,g}(\tau_\delta)$ for a group g, which gives

$$\epsilon_g \geq \frac{1}{2} \min_{a \in A} \epsilon_{a,g}(\tau_\delta).$$

Now, we expand out $\epsilon_{a,g}(\tau_\delta)$ and get

$$\frac{1}{2} \min_{a \in A} \epsilon_{a,g}(\tau_\delta) = \frac{1}{2} \min_{a \in A} \sqrt{\frac{N(\beta(N(\tau_\delta), \delta_E)}{2N_{a,g}(\tau_\delta)}}. \tag{5.5}$$

We take the upper bound on $N_{a,g}(\tau_\delta)$ from Lemma 9 which is

$$N_{a,g}(\tau_\delta) \leq \frac{8\beta(N(\tau_\delta), \delta_E)}{\max((D_{a,g} - C_a)^2, (C_a - \frac{C_1+C_2}{2})^2)}$$

at random stopping time τ_δ, and plug it in for $N_{a,g}(\tau_\delta)$ from (5.5). This gives a lower bound on (5.5) of

$$\frac{1}{2} \min_{a \in A} \sqrt{\frac{\beta(N(\tau_\delta), \delta_E)}{2\left(\frac{8\beta(N(\tau_\delta), \delta_E)}{\max((D_{a,g} - C_a)^2, (C_a - \frac{C_1+C_2}{2})^2)}\right)}}. \tag{5.6}$$

After some cancellations, we get that (5.6) is equal to

$$\frac{1}{8} \min_{a \in A} \sqrt{\max\left((D_{a,g} - C_a)^2, \left(C_a - \frac{C_1 + C_2}{2}\right)^2\right)},$$

which is our lower bound of ϵ_g. □

Given a lower bound on ϵ_g, we can now put an upper bound on the sample complexity on INC-EG.

Proof of Theorem 12

Proof. We first look at the sample complexity of all of the sub-optimal arms in each group, and then the optimal highest reward arms for each group. For the sub-optimal arms, we use Theorem 2 to get a sample complexity of all of the sub-optimal arms for a group. This is because the (ϵ, δ)-PAC Exponential Gap algorithm will pull up to some precision of ϵ. The Incremental-Exponential Gap algorithm will be pulling arms only up to some level of precision ϵ_g for each group g, from Lemma 13. Hence, if we set ϵ from EG to be ϵ_g, then we have that the sample complexity for the sub-optimal arms from the EG algorithm will be the same sample complexity as from the INC-EG algorithm for the same arms.

We prove the sample complexity of the highest reward arm in each group through the following summation,

$$\sum_{r=1}^{\log_2(1/\epsilon_g)} t_r = 32 \sum_{r=1}^{\log_2(1/\epsilon_g)} 4^r \ln\left(\frac{100r^2}{\delta_g}\right) = O\left(\frac{1}{\epsilon_g^2}\log\left(\frac{1}{\delta_g}\log(\frac{1}{\epsilon_g^2})\right)\right),$$

inspired by the proof for Lemma 3.5 in [Karnin et al., 2013]. The total number of rounds we are going to have is $\log_2(1/\epsilon_g)$, as after that many rounds we will have a precision of ϵ_g.

This gives a sample complexity of the best arms in each group, for which $D_{a,g} = 0$. We know $\epsilon_g > 0$ from Lemma 13, so we get $\max(D_{a,g}, \epsilon_g) = \epsilon_g$. This means we combine the sample complexity of the two parts to a single summation and get

$$M = \sum_{(a,g)\in\mathcal{P}} O\left(\frac{1}{(D_{a,g}^{\epsilon_g})^2}\log\left(\frac{1}{\delta_g}\log\left(\frac{1}{(D_{a,g}^{\epsilon_g})^2}\right)\right)\right).$$

□

Combining the number of pulls from MD-LUCB and INC-EG, we get that with

high probability $1 - \delta$, the total number of samples we get can be upper bounded by $\tau_\delta \leq N + M$, for N in Theorem 7 and M in Theorem 12.

A discussion of the Lower bound exists in the Chapter 7.

Chapter 6

Experiments

In this section, we look at multiple settings, illustrating different parts of the theoretical results. All experiments were run for 1000 repetitions with $\delta = 0.001$. The best arm in Setting 1 is arm 2, in Settings 2 and 2V it is arm 1, and in Settings 3 and 3V it is arm 3. There are two different exploration rates used in the experiment section, with Beta1 being the exploration rate in (5.3), and Beta2 as the exploration rate in (5.4). When showing the sample complexity of MD-LUCB in tables, we denote experiments using Beta1 as an exploration rate with the symbol $N1$ and experiments using Beta2 as an exploration rate with the symbol $N2$. The symbol M in tables gives the sample complexity from INC-EG pulls. We do not distinguish the INC-EG pulls for the two different exploration rates as the number of pulls from INC-EG stays extremely similar regardless of which exploration rate is used. It is also important that the pulls from INC-EG include pulls from its calls to Median Elimination.

Setting 1 has two arms and five groups and show that this algorithm does not excessively sample to find the highest reward arm in a group if there are no arms for which that group maximizes disappointment. Table 6.1 shows the mean rewards of Setting 1, with the disappointment levels of each arm in parentheses beside the mean rewards.

	G 1	G 2	G 3	G 4	G 5
A 1	0.2 (0.3)	0.7 (0)	0.6 (0.05)	0.55 (0)	0.2 (0.6)
A 2	0.5 (0)	0.3 (0.4)	0.65 (0)	0.53 (0.02)	0.8 (0)

Table 6.1: The mean rewards of arm-group pairs and in parentheses their disappointment values of Setting 1.

	$\tau_{1,1}$	$\tau_{2,1}$	$\tau_{1,2}$	$\tau_{2,2}$	$\tau_{1,3}$	$\tau_{2,3}$	$\tau_{1,4}$	$\tau_{2,4}$	$\tau_{1,5}$	$\tau_{2,5}$
N1	1.82	0.89	0.32	33.1	0.48	0.89	0.40	0.99	33.4	0.62
N2	0.16	0.08	0.03	2.92	0.04	0.08	0.04	0.09	2.96	0.05
M	16.3	16.3	59.5	16.3	16.3	16.3	16.3	16.3	16.3	59.5

Table 6.2: Number of pulls from MD-LUCB with Beta1 function ($N1$), Beta2 function ($N2$), and INC-EG (M) in units of 1000's for Setting 1.

The results in Table 6.2 show that for all 1000 repetitions, the only times INC-EG was called more than once for a group were in groups two and five, as those groups maximized the disappointment for arms one and two. Groups one, three and four call INC-EG once, which are the groups where it is much harder to determine which arm has the highest mean reward, and have very little disappointment. This behaviour is expected, as when ϵ_g lowers we should require extra calls to INC-EG from Lemma 13.

We now look at Setting 2 in Table 6.3, and Setting 2V in Table 6.4. The only difference between these settings is we change the mean reward of $(2, 1)$ from 0.4 (0.5 disappointment) in Setting 2 to 0.45 (0.45 disappointment) in Setting 2V. We do this alteration to highlight how the number of pulls from MD-LUCB changes when only altering one arm-group pair.

	G 1	G 2
A 1	0.9 (0)	0.4 (0.3)
A 2	0.4 (0.5)	0.6 (0.1)
A 3	0.3 (0.6)	0.7 (0)

Table 6.3: The mean rewards of arm-group pairs and in parentheses their disappointment values of Setting 2.

	G 1	G 2
A 1	0.9 (0)	0.4 (0.3)
A 2	0.45 (0.45)	0.6 (0.1)
A 3	0.3 (0.6)	0.7 (0)

Table 6.4: The mean rewards of arm-group pairs and in parentheses their disappointment values of Setting 2V.

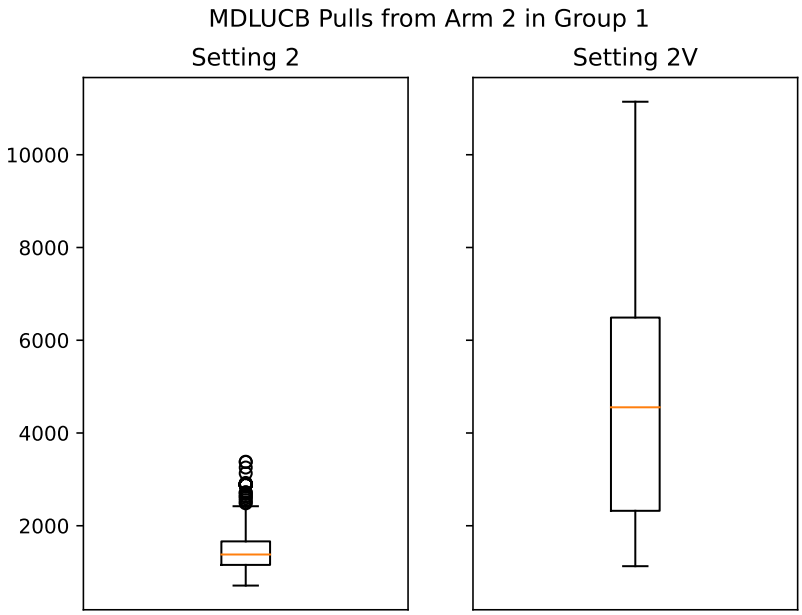

Figure 6.1: Boxplot displaying the pulls from MD-LUCB for arm 2 in group 1 from Settings 2 and 2V with exploration rate Beta2 over the 1000 simulations. The circles in the plot are outliers. There are 54 outliers in Setting 2, with 0 in Setting 2V

Figure 6.1 is a boxplot showing the different number of pulls over the 1000 repetitions for pair $(2, 1)$ in Setting 2 and 2V. We observe that the number of pulls goes up drastically, as we would expect as pair $(2, 1)$ which has cost C_2. The only other arm to go up significantly is pair $(1, 2)$, which has cost C_1, and is to be expected by our theoretical findings. Figure 6.2 shows us that MD-LUCB is only adding extra samples to the specific arm-group pairs that have a shift in disappointment.

Settings 3 in Table 6.8 and 3V in Table 6.9 are more complicated 3 arm and 3 group settings with smaller gaps between the cost of disappointment for each arm. These setting do well to show the benefits one can have with the smaller exploration rate we get in Lemma 11, so we can show how much smaller our sample complexity is for pulls from MD-LUCB. Figure 6.3 shows the number of pulls from MD-LUCB with the two different learning rates for all of the settings, where Beta1 is using the

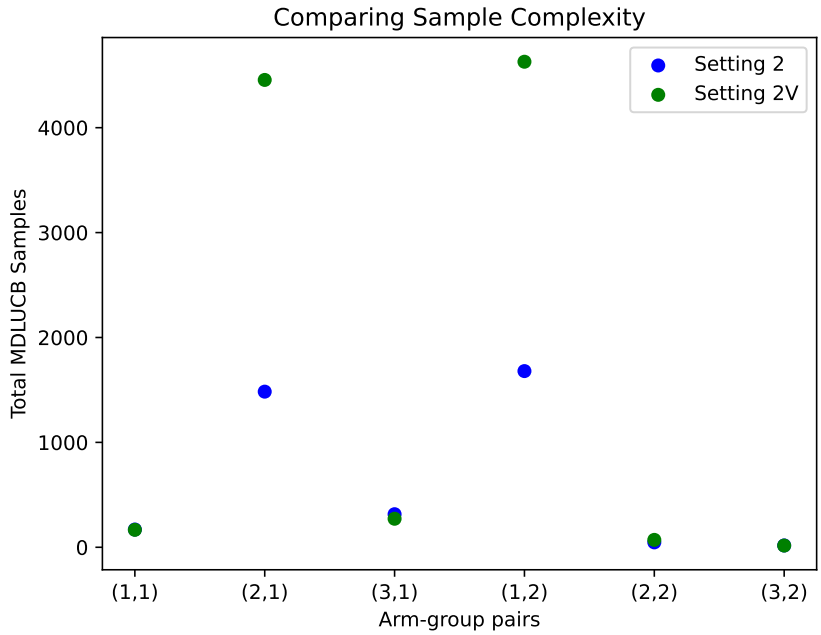

Figure 6.2: This plot shows the sample complexity from MD-LUCB pulls of the arm-group pairs in Settings 2 and 2V with Beta2 as the exploration rate.

exploration rate in Lemma 10 and Beta2 is using the exploration rate in Lemma 11.

	$\tau_{1,1}$	$\tau_{2,1}$	$\tau_{3,1}$	$\tau_{1,2}$	$\tau_{2,2}$	$\tau_{3,2}$
N1 2	1.76	16.2	3.46	18.6	0.51	0.17
N1 2V	1.78	55.0	3.23	57.3	0.81	0.17

Table 6.5: Number of pulls from MD-LUCB with Beta1 ($N1$) in units of 1000's for Settings 2 and 2V.

It is important to note however that the number of samples overall from the algorithms is still much larger, as the total pulls from INC-EG are more than the total pulls from MD-LUCB in all settings with either exploration rate. If we want to make a more efficient version of this algorithm for practical use, we would need to replace INC-EG with a different subroutine that is more efficient at estimating the value of the highest mean reward arm, or come up with a completely new strategy.

	$\tau_{1,1}$	$\tau_{2,1}$	$\tau_{3,1}$	$\tau_{1,2}$	$\tau_{2,2}$	$\tau_{3,2}$
N2 2	0.17	1.48	0.32	1.68	0.05	0.02
N2 2V	0.17	4.46	0.27	4.63	0.07	0.02

Table 6.6: Number of pulls from MD-LUCB with Beta2 ($N2$) in units of 1000's for Settings 2 and 2V.

	$\tau_{1,1}$	$\tau_{2,1}$	$\tau_{3,1}$	$\tau_{1,2}$	$\tau_{2,2}$	$\tau_{3,2}$
M 2	50.8	40.8	15.3	15.3	107	110
M 2V	69.7	40.8	15.3	15.3	107.6	129.2

Table 6.7: Number of pulls from (M) in units of 1000's for Settings 2 and 2V.

	G 1	G 2	G 3
A 1	0.5(0.1)	0.2(0.25)	0.6(0.2)
A 2	0.6(0)	0.45(0.05)	0.4(0.4)
A 3	0.4(0.2)	0.5(0)	0.8(0)

Table 6.8: The mean rewards of arm-group pairs and in parentheses their disappointment values for Setting 3.

	G 1	G 2	G 3
A 1	0.55(0.05)	0.2(0.25)	0.7(0.1)
A 2	0.6(0)	0.425(0.025)	0.5(0.3)
A 3	0.4(0.2)	0.45(0)	0.8(0)

Table 6.9: The mean rewards of arm-group pairs and in parentheses their disappointment values for Setting 3V.

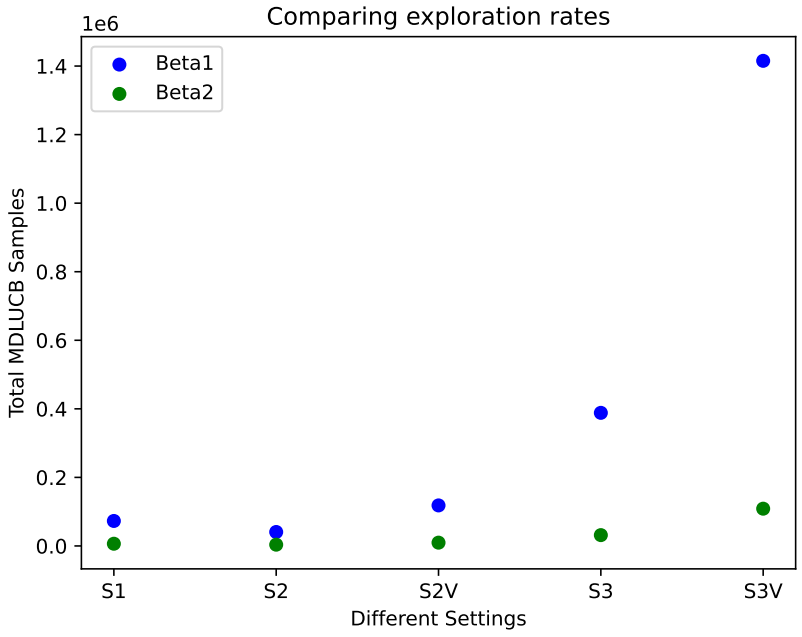

Figure 6.3: Beta1 uses the exploration rate in (5.3), and Beta2 uses the exploration rate on (5.4)

Chapter 7

Conclusions and future work

In this work, we introduced a new definition of fairness in the bandit setting called disappointment fairness. One of the main challenges of this problem setting is estimating the highest mean reward of each group. A further complication comes from the fact that the precision of the estimated highest mean reward for each group does not need to be the same, as for some groups being more precise can be unnecessary. This leads to the algorithms MD-LUCB and INC-EG, which are the first algorithms used to solve this problem. INC-EG gives us an estimate of the highest mean reward arm in a group, and depending on how many times we call it, how precise that estimate is. Then MD-LUCB will pull to find the arm that minimizes the maximum disappointment, and then strategically calls INC-EG on specific groups when it needs a more precise estimate of the highest mean reward from a group.

The proofs for MD-LUCB's correctness and sample complexity rely on the fact that we know the confidence interval around the highest mean reward from a group will always be smaller than the confidence interval surrounding a specific arm from that group every time we check our stopping condition. This allows us to double the confidence interval around an arm from that group, as seen in Lemma 9, and give us a bound on the sample complexity of pulls from MD-LUCB that does not depend on INC-EG's pulls. Whenever this is not true, we call INC-EG for that group and

get a smaller confidence interval around the highest mean reward from that group. If we have a bound on the sample complexity of MD-LUCB that does not rely upon INC-EG, then we can use that to get a bound for the number of calls of INC-EG we need from each group (and therefore its sample complexity).

7.1 Lower Bound

As of now, there does not exist an explicit lower bound in this setting, but through the use of specific problem instances, we can show that this problem has a dependence on the total number of arms and groups, and the complexity terms in MD-LUCB are at least close to optimal.

First, as of now we are not aware of any better estimator of the disappointment than forming confidence intervals around μ_g^* and $\mu_{a,g}$, and we believe that there does not exist a better way of estimating $D_{a,g}$. Another idea for estimating the disappointment would be instead of pulling only the arm-group pairs associated with $\hat{C}_1$ and $\hat{C}_2$, we look at which groups give rise to our $\hat{C}_1$ and $\hat{C}_2$ values in an given epoch and pull every arm in those groups, and then using only those pulls to form confidence intervals around the disappointment. However, this technique will cost extra pulls on all of the arm-group pairs in those groups that are not used for μ_g^* and the arm-group pair associated with $\hat{C}_1$ or $\hat{C}_2$. Therefore, to calculate the disappointment of a pair (a, g) in as few samples as possible would require pulling the arm associated with μ_g^* in the same group at least that number of times.

We know that this problem setting depends on the total number of arms, as if we have a one group, K arm problem, this simplifies to a standard best arm identification as the optimal solution, and this has a lower bound that is dependent on the total number of arms.

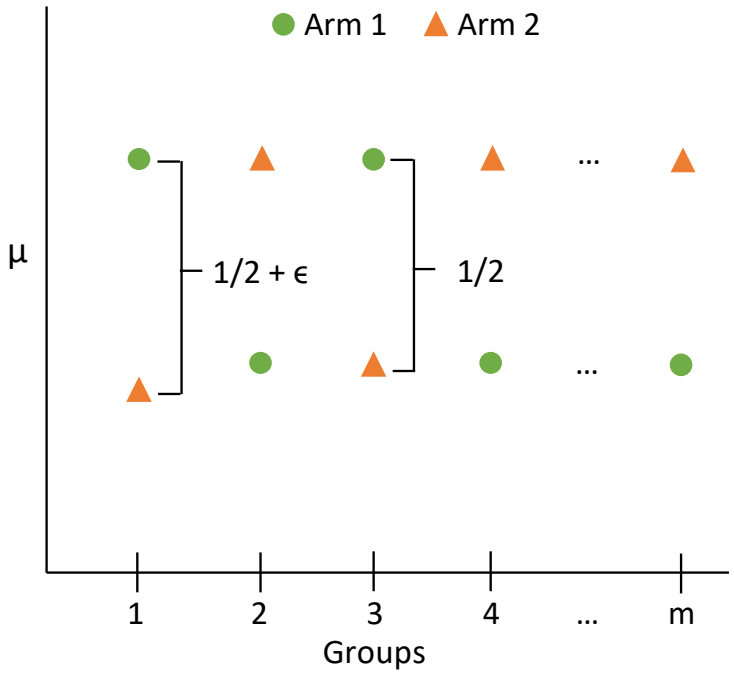

Figure 7.1: Visualization of what problem instance I_1 may look like. The y axis shows the mean reward, with the disappointment for arm 2 in groups 1 and 3 shown as $1/2 + \epsilon$ and $1/2$.

	Group 1	Group 2	Group 3	...	Group m
Arm 1 Disappointment	0	$1/2$	0	...	$1/2$
Arm 2 Disappointment	$1/2 + \epsilon$	0	$1/2$	...	0

Table 7.1: Disappointment suffered from arm-group pairs for problem instance I_1. In this case, $C_1 = 1/2$ and $C_2 = 1/2 + \epsilon$, making arm 1 the best arm.

Now we want to show a dependence on the total number of groups in the lower bound. Let us say there are m problem instances, where m is our total number of groups, labeled as $I = \{I_1, I_2, \ldots, I_m\}$. In each instance, we have 2 arms and m groups, where arm 1 has $1/2$ disappointment in even groups and 0 in odd groups, and arm 2 has $1/2$ disappointment in odd groups and 0 in even groups. The only difference is in problem instance g, where if the gth group is even, arm 1 has a disappointment of $1/2 + \epsilon$ instead of $1/2$ in group g; if the gth group is odd, arm 2 has a disappointment of $1/2 + \epsilon$ instead of $1/2$ in group g. Table 7.1 contains an example of problem instance

I_1, with Figure 7.1 visualizing it. We have all odd problem instances having arm 1 as their best arm, and all even instances having arm 2 as their best arm.

For any algorithm which knows that it has a problem instance from I, it will need $\Omega(\log(1/\delta)/\epsilon^2)$ samples from an arm in every group in order to identify the best arm with high probability $1-\delta$. If our algorithm had fewer samples than this, then it cannot tell whether it has an instance where arm 1 is the best or arm 2 is the best arm, as it cannot tell which group has an arm with disappointment of $1/2+\epsilon$, and not knowing means it cannot choose the best arm with high probability. The algorithm will not need to pull any times from arm-group pairs with 0 disappointment, as they all have the same mean reward in this setting, and so the algorithm would know this.

For any algorithm trying to solve from these subsets of problems instances, it would need to explore every group up to a high level of certainty to find which arm is the best. Therefore, we know our lower bound will be dependent on the total number of groups, and the total number of arms.

7.2 Future Work

There are a few future directions that can be taken from this work. One involves looking at returning the top or best w arms instead of just a single best arm. The best w arms are decided under the assumption that when given a group, we get to pick which arm from the w arms to respond to that group with. Therefore, the top w arms are picked such that we minimize the maximum disappointment any group may feel assuming that group will get the arm that maximizes its reward from the set of w arms. If we were to look at this through the clinical trials problem shown in Chapter 1, the only thing that would change is now we would be able to pick multiple drugs to bring to mass production, which would make sense for a pharmaceutical company.

	G1	G2
A 1	0.5(0.3)	0.5(0.3)
A 2	0.1(0.7)	0.8(0)
A 3	0.8(0)	0.1(0.7)

Table 7.2: Under this setup when $w = 1$ arm 1 is the best, but when $w = 2$ then arm's 2 and 3 are the best 2 arms.

This problem is not as simple as the standard top w arms in standard best arm identification. For best arm identification for highest reward arms, the best w arms is always a subset of the best $w + 1$ arms, but in the case of maximin fairness [Garivier et al., 2016] and disappointment minimax fairness, this is not true. Table 7.2 shows an example of this for maximin fairness and disappointment minimax fairness. This means a different approach would be needed to solve this problem, as the greedy pulling strategy would not work when trying to return the top w arms.

Finally, the algorithm for solving this problem setting is inefficient, as MC-LUCB is very impractical when it comes to estimating the highest mean reward for groups. This is because INC-EG is a variation of the Exponential Gap algorithm, which is known to be very inefficient in practice. One area of interest is if there is a more efficient way to sample to have the same guarantees about highest mean reward of an arm. There exists other algorithms such as lil'UCB [Jamieson et al., 2014] that can efficiently find the highest mean reward arm, but they do not give the estimate of the highest mean reward, which is what is needed for MD-LUCB. However, because we treat INC-EG as a black box in this algorithm, we can swap INC-EG out for another algorithm that can estimate μ_g^* and this does not change any of the proofs of derivations for the MD-LUCB sample complexity. So, this step should be quite simple and we may consider it in the future.

Appendix A

Additional Information

A.1 Proofs

Proof of Lemma 8

Proof. Let us assume that this is not true. That means one of the 4 following cases must be true:

- $(L_{D_{P_1(t_s)}}(t_s) > c) \cap (L_{D_{P_2(t_s)}}(t_s) > c)$
- $(U_{D_{P_1(t_s)}}(t_s) < c) \cap (U_{D_{P_2(t_s)}}(t_s) < c)$
- $(U_{D_{P_1(t_s)}}(t_s) < c) \cap (L_{D_{P_2(t_s)}}(t_s) > c)$
- $(L_{D_{P_1(t_s)}}(t_s) > c) \cap (U_{D_{P_2(t_s)}}(t_s) < c)$

For cases 1 and 2 we use Lemmas 5 and 6 to know that the pairs $P_1(t_s)$ and $P_2(t_s)$ are picked from a set of arms where each arm a has a confidence interval corresponding to its C_a term.

Case 1 cannot happen based on the following logic:

$$\begin{aligned} c &\geq C_1 \\ &= \min_{a \in A} C_a \\ &\geq \min_{a \in A} L_{C_a} \\ &= L_{P_1(t_s)} \text{ or } L_{P_2(t_s)}. \end{aligned}$$

Case 2 cannot happen based on one of the two following situations for arm j, where arm j and k are

$$\begin{aligned} j &= \underset{a \in \{a_1(t_s), a_2(t_s)\}}{\arg\max} U_{D_{a,g_a(t_s)}}(t_s), \\ k &= \underset{a \in \{a_1(t_s), a_2(t_s)\}}{\arg\min} U_{D_{a,g_a(t_s)}}(t_s). \end{aligned}$$

If $j > 1$, then this means that

$$\begin{aligned} U_{D_{j,g_j(t_s)}}(t_s) &\geq C_j \\ &\geq C_2 \\ &\geq c, \end{aligned}$$

and so then case 2 cannot happen in this situation. If $j = 1$, then we know that

$$\begin{aligned} U_{D_{j,g_j(t_s)}}(t_s) &> U_{D_{k,g_k(t_s)}}(t_s) \\ &\geq C_k \\ &\geq C_2 \\ &\geq c, \end{aligned}$$

which means case 2 cannot happen in this situation. Therefore, case 2 cannot happen.

Case 3 is our stopping condition, so if that were to happen then the algorithm would halt.

Case 4 cannot happen because, if it were true, that implies $L_{D_{P_1(t_s)}}(t_s) > U_{D_{P_2(t_s)}}(t_s)$. This means $\hat{C}_{a_1(t_s)} > \hat{C}_{a_2(t_s)}$ which, by the definition of $a_1(t_s)$, cannot happen as $a_1(t_s) = \arg\min_{a \in A} \hat{C}_a(t_s)$.

Therefore, because none of these 4 cases can happen, we know $c \in [L_{D_P}(t_s), U_{D_P}(t_s)]$. □

Proof of Lemma 9

Proof. Assume that $\mathcal{E}_s$ and $\mathcal{H}$ hold, and $c \in [L_{D_{a,g}}(t_s), U_{D_{a,g}}(t_s)]$. By the end of time t_s, both pairs $P_1(t_s)$ and $P_2(t_s)$ are picked from a set of arm-group pairs $(a, g_a(t_s))$, where $a \in A$. This means $c \in [L_{D_{a,g_a(t_s)}}(t_s), U_{D_{a,g_a(t_s)}}(t_s)]$. Now from (4.4) we know $c \in [L_{C_a}(t_s), U_{C_a}(t_s)]$, and then from Lemma 6, $C_a \in [L_{D_{a,g}}(t_s), U_{D_{a,g}}(t_s)]$.

Both c and C_a are contained within $[L_{D_{a,g}}(t_s), U_{D_{a,g}}(t_s)]$, which has a diameter of $\sqrt{8\beta(N(t_s), \delta_E)/N_{a,g}(t_s)}$. We get this by taking the diameter $2(\sqrt{\beta(N(t_s), \delta_E)/2N_{a,g}(t_s)} + \epsilon_g(r_g(t_s)))$, and realizing that $\sqrt{\beta(N(t_s), \delta_E)/2N_{a,g}(t_s)} \geq \epsilon_g(r_g(t_s))$ by the algorithm's construction. We upper bound the value of the diameter of $D_{a,g}$'s confidence interval by $2(2(\sqrt{\beta(N(t_s), \delta_E)/2N_{a,g}(t_s)})) = \sqrt{8\beta(N(t_s), \delta_E)/N_{a,g}(t_s)}$. Now we get

$$|C_a - c| \leq \sqrt{\frac{8\beta(N(t_s), \delta_E)}{N_{a,g}(t_s)}} \Leftrightarrow N_{a,g}(t_s) \leq \frac{8\beta(N(t_s), \delta_E)}{(C_a - c)^2}.$$

We can also use $L_{D_{a,g}}(t_s) = L_{C_a}(t_s)$ to get the following:

$$U_{D_{a,g}}(t_s) - \sqrt{\frac{8\beta(N(t_s), \delta_E)}{N_{a,g}(t_s)}} = L_{D_{a,g}}(t_s)$$
$$\Updownarrow$$
$$U_{D_{a,g}}(t_s) - \sqrt{\frac{8\beta(N(t_s), \delta_E)}{N_{a,g}(t_s)}} = L_{C_a}(t_s)$$
$$\Updownarrow$$
$$D_{a,g} - \sqrt{\frac{8\beta(N(t_s), \delta_E)}{N_{a,g}(t_s)}} \leq C_a,$$

which gives $N_{a,g}(t_s) \leq \frac{8\beta(N(t_s),\delta_E)}{(D_{a,g}-C_a)^2}$.

Therefore, we get $N_{a,g}(t_s) \leq \min(\frac{1}{(C_a-c)^2}, \frac{1}{(D_{a,g}-C_a)^2})8\beta(N(t_s), \delta_E)$ for some arm-pair (a, g). □

Proof of Lemma 10

Proof. First, we want to show that with the exploration rate in (5.3), event $\mathcal{E}$ holds with probability at least $1 - \delta_E$. First we use Hoeffding's inequality on a single arm in a single group by the end of time t and get

$$\Pr(|\mu_{a,g} - \hat{\mu}_{a,g}(t)| > \epsilon) \leq 2\exp(-N_{a,g}(t)\epsilon^2) = \delta_{a,g}(t).$$

We want the bound we get from Hoeffding's inequality to hold for all arms and groups at times t_s where $s \in \mathbb{N}$. which is what event $\mathcal{E}_s$ is defined as earlier in this paper. We say event $\mathcal{E}_s$ fails to hold with probability δ_s. We upper bound the failure probability δ_s as

$$\delta_s \leq \sum_{(a,g)\in\mathcal{P}} \delta_{a,g}(t_s) = 2|A||G|\exp(-N_{a,g}(t_s)\ \epsilon^2).$$

We want $\mathcal{E}_s$ to hold for all epochs s, which is defined as event $\mathcal{E}$ earlier in the paper. We set $\delta_s = \delta_E/(2N(t_s)+1)^2$ and then use a union bound over $\mathcal{E}_s$ for $s=0$ to $s=\infty$ to show that $\mathcal{E}$ holds with at least probability $1-\delta_E$. So, the failure probability of event $\mathcal{E}$ is upper bounded by

$$\sum_{s=0}^{\infty} \frac{\delta_E}{(N(t_s)+1)^2} \leq \delta_E.$$

Therefore, event $\mathcal{E}$ fails with probability at most δ_E. □

Now that we know that our choice of exploration rate makes event $\mathcal{E}$ hold with probability at least $1-\delta_E$, we want to find the smallest value of $N(t_s)$ for epoch s such that

$$16H(\boldsymbol{\mu}) \log\left(\frac{2|A||G|(N(t_s)+1)^2}{\delta_E}\right) < N(t_s). \tag{A.1}$$

First, define a constant $c_1 = \log(\frac{2|A||G|}{\delta_E})$. We can use c_1 and then upper bound the left side of (A.1) as

$$16H(\boldsymbol{\mu}) \log(c_1) + 64H(\boldsymbol{\mu}) \log(N(t_s)).$$

Now define constants $c_2 = 16H(\boldsymbol{\mu}) \log(c_1)$ and $c_3 = 64H(\boldsymbol{\mu})$ to simplify notation. Now we want to solve the equation

$$c_2 + c_3 \log(N(t_s)) < N(t_s). \tag{A.2}$$

It suffices to choose some $N(t_s)$ large enough such that $c_2 < N(t_s)/2$ and $c_3 \log(N(t_s)) < N(t_s)/2$. We can now define constants $c_4 = 2c_2$ and $c_5 = 2c_3$ such that we get two equations, where

$$c_4 < N(t_s) \quad \text{and} \quad c_5 \log(N(t_s)) < N(t_s).$$

We know $N(t_s) \geq c_4$, and for the second equation we use the following claim that if $x = 3C \log C$, then $C \log x \leq x$. This is true as long as $C > 3$. In this case, $C = c_5$

and $x = N(t_s)$. So, by this logic we need $N(t_s) \geq c_5 \log(c_5)$, and we will need $c_5 > 3$, which means $H(\boldsymbol{\mu}) > 3/128$.

Therefore $N(t_s) = \max(c_4, 3c_5 \log(c_5))$ makes (A.2) true, which means (A.1) true. So if $H(\boldsymbol{\mu}) > 3/128$,

$$N \leq \max(c_4, c_5 \log(c_5)) = \max\left(32H(\boldsymbol{\mu}) \log\left(\frac{2AG}{\delta}\right),\ 384H(\boldsymbol{\mu}) \log(128H(\boldsymbol{\mu}))\right).$$

Proof of Lemma 11

Proof. We want to first prove that event $\mathcal{E}$ holds with at least probability $1 - \delta_E$. To do this we denote the set of random variables from pair P that get picked by the MD-LUCB part of the algorithm by $Z_1^P, Z_2^P, \ldots Z_{N_P(t)}^P$, where $N_P(t)$ is the last random variable that has been picked from MD-LUCB for arm-group pair P at time t. $N_P(t)$ is upper bounded by $N(t)$, where pair P has been picked at every time we have picked a pair from MD-LUCB. This set of random variables can be centered around the mean reward of that arm, which is μ_P. We can then state that for each pair P,

$$\begin{aligned} &Pr\left(\exists N(t) \in \mathbb{N} : \sum_{s=1}^{N(t)} \frac{Z_s^P - \mu_P}{2\sigma^2} > \sqrt{2N(t)\beta(N(t), \delta_E)}\right) \\ &= Pr(\exists N(t) \in \mathbb{N} : S_t > \sqrt{2N(t)\beta(N(t), \delta_E)}), \end{aligned} \tag{A.3}$$

where S_t is the sum of $N(t)$ i.i.d random variables. To get a bound on the probability in (A.3) we use Lemma 7 from [Kaufmann et al., 2016], which is restated below as

Lemma 7: *Let $\zeta(u) = \sum_{k \geq 1} k^{-u}$. Let $X_1, X_2, \ldots, X_t$ be independent random variables such that for all $\lambda \in \mathbb{R} : \phi(\lambda) := \log \mathbb{E}[\lambda X_1] \leq \lambda^2 \sigma^2 / 2$. For every positive integer*

t, let $S_t = X_1 + X_2 + \cdots + X_t$. *Then, for all* $\eta > 1$, *and* $x \geq \frac{8}{(e-1)^2}$,

$$Pr(\exists t \in \mathbb{N} : S_t > |\sqrt{2\sigma^2 t(x + \eta(\log\log(et))}|) \leq 2\sqrt{e}\zeta(\eta(1 - \frac{1}{2x}))(\frac{\sqrt{x}}{2\sqrt{2}} + 1)^\eta \exp(-x).$$

This allows us to bound each arm-group pair P for all times $N(t) \in \mathbb{N}$ with the correct exploration rate, and we can use a union bound over all arm-group pairs to get a bound that with high probability $1 - \delta_E$, no arm-group pair deviates from its exploration rate by more than $\epsilon_{i,g}(t)$ in either direction. This means we can bound the failure probability of event $\mathcal{E}$ as

$$\delta_E \leq 2|A \times G|\sqrt{e}\zeta(\eta(1 - \frac{1}{2x}))(\frac{\sqrt{x}}{2\sqrt{2}} + 1)^\eta \exp(-x).$$

We use the exploration rate of

$$\beta(N(t), \delta_E) = \log(1/\delta_E) + b\log\log(1/\delta_E) + 3\log\log(eN(t)),$$

where $b > 6$, and $\delta_E < 0.049$. This means that if we plug in $\eta = 3$ and $x = a + b\log(a)$ into our bound, where $a = \log(1/\delta_E)$, we get

$$2|A \times G|\sqrt{e}\zeta(3(1 - \frac{1}{2(a + b\log(a))}))(\frac{\sqrt{a + b\log(a)}}{2\sqrt{2}} + 1)^3 \exp(-(a + b\log(a))) \quad \text{(A.4)}$$

For any $a > 3$ (which is true when $\delta_E < 0.049$) and $b > 6$, we know that $\zeta(3(1 - \frac{1}{2(a+b\log(a))})) \leq \zeta(2) = \pi^2/6$, so we can upper bound the Riemann zeta function as $\pi^2/6$. We rearrange Equation A.4 to get an upper bound of

$$|A \times G|\sqrt{e}(\frac{\pi^2}{3})(\frac{\sqrt{a + b\log(a)}}{2\sqrt{2}} + 1)^3 \frac{\delta_E}{a^b}.$$

It can be shown that empirically, for $a > 3$, and $b > 6$, then we get that

$$f_b(\delta_E) = |A \times G|\sqrt{e}(\frac{\pi^2}{3})(\frac{\sqrt{a + b\log(a)}}{2\sqrt{2}} + 1)^3 \frac{1}{a^b} \leq 1,$$

which means that with probability $1 - f_b(\delta_E)\delta_E$ we know that event $\mathcal{E}$ holds.

However, the failure probability is dependent on the number of arm-group pairs, which means we need to increase the value of b or decrease δ_E if we need a guarantee for more pairs. For example, if $\delta_E < 0.049$, event $\mathcal{E}$ holds for 14 arm-group pairs with $b > 6$, for 40 arm-group pairs with $b > 7$, for 111 arm-group pairs with $b > 8$, for 310 arm-group pairs with $b > 9$, and for 869 arm-group pairs with $b > 10$. If we need a theoretical guarantee for more arm-group pairs than this, then we would need to either shrink δ_E or increase b.

Now we need to connect the bound from Lemma 7 to our confidence radius in Equation (4.2). It is important to note that our bound is for the average reward, not sum, so we divide by $N(t)$. This leaves us with

$$\frac{\sqrt{2N(t)\sigma^2\beta(N(t), \delta_E)}}{N(t)} = \sqrt{\frac{2\sigma^2\beta(N(t), \delta_E)}{N(t)}}.$$

We know that for sub-gaussian random variables that $\sigma^2 \leq \frac{b-a}{4}$, where for these trials $b = 1$ and $a = 0$. Therefore $\sigma^2 \leq 1/4$. If we plug in $\sigma^2 = 1/4$ we get

$$\sqrt{\frac{\beta(N(t), \delta_E)}{2N(t)}} \leq \sqrt{\frac{\beta(N(t), \delta_E)}{2N_{a,g}(t)}},$$

which is our confidence radius from (4.2).

To get an explicit bound on the number of pulls from MD-LUCB, we need a bound such that

$$16H(\boldsymbol{\mu})(\log(1/\delta_E) + b\log\log(1/\delta_E) + 3\log\log(eN(t))) < N(t).$$

First this to simplify notation we set $c_1 = 16H(\boldsymbol{\mu})$. This has already been split up into 3 separate components. So if we find the biggest component of the three, times that by 3 and we have a bound on $N(t)$. This means we need the max of

$$\begin{aligned} c_1\log(1/\delta_E) &< N(t)/3, \\ c_1 \cdot b\log\log(1/\delta_E) &< N(t)/3, \\ c_1 \cdot 3\log\log(eN(t)) &< N(t)/3. \end{aligned}$$

We set $c_2 = 3c_1 \cdot b$ and $c_3 = 9c_1$. The first two cases are easy, as we just need to make sure $N(t) > c_2\log(1/\delta_E)$. The third case, we can use the fact that for the equation $C\log\log(ex) < x$, we know this is true for $x = 3C\log\log(C)$, for any $C > 3e$. This can be verified with

$$\begin{aligned} C\log\log(e(3C\log\log(c))) &< 3C\log\log(C) \\ C\log\log(C) + C\log\log(3e) + C\log\log\log\log(C) &< 3C\log\log(C) \\ C\log\log(3e) + C\log\log\log\log(C) &< 2C\log\log(C) \\ C\log\log(3e) &< C\log\log(C). \end{aligned}$$

Using this fact, we can set $C = c_3$, and then we get that $c_3\log\log(ec_3) < N(t)$. This gives us an explicit bound using this exploration rate, where

$$N \leq 48H(\boldsymbol{\mu})\max\left(\log(\frac{1}{\delta_E}), b\log\log(\frac{1}{\delta_E}), 3\log\log(3e \cdot 48H(\boldsymbol{\mu})\right).$$

□

A.2 Tables

Tables A.1, A.2, and A.3 contain the average number of pulls of MD-LUCB and INC-EG in the 1000s for settings 3 and 3V over 1000 repetitions with $\delta = 0.001$.

	$\tau_{1,1}$	$\tau_{2,1}$	$\tau_{3,1}$	$\tau_{1,2}$	$\tau_{2,2}$	$\tau_{3,2}$	$\tau_{3,1}$	$\tau_{3,2}$	$\tau_{3,3}$
N1 3	3.97	0.486	188	145	0.590	2.10	34.7	9.39	3.44
N1 3V	4.10	1.11	699	622	1.30	2.81	9.76	71.5	2.53

Table A.1: Number of pulls from MD-LUCB with Beta1 ($N1$) in units of 1000's for Settings 3 and 3V.

	$\tau_{1,1}$	$\tau_{2,1}$	$\tau_{3,1}$	$\tau_{1,2}$	$\tau_{2,2}$	$\tau_{3,2}$	$\tau_{3,1}$	$\tau_{3,2}$	$\tau_{3,3}$
N2 3	0.33	0.04	15.2	11.8	0.05	0.20	2.82	0.81	0.30
N2 3V	0.31	0.09	53.6	48.0	0.11	0.23	0.74	5.42	0.23

Table A.2: Number of pulls from MD-LUCB with Beta2 ($N2$) in units of 1000's for Settings 3 and 3V.

	$\tau_{1,1}$	$\tau_{2,1}$	$\tau_{3,1}$	$\tau_{1,2}$	$\tau_{2,2}$	$\tau_{3,2}$	$\tau_{3,1}$	$\tau_{3,2}$	$\tau_{3,3}$
M 3	109	287	15.7	15.7	417	560	42.0	15.7	78.6
M 3V	412	1150	15.7	15.7	1702	2300	111	15.7	145

Table A.3: Number of pulls from INC-EG (M) in units of 1000's for Settings 3 and 3V.

Glossary

Notation	Description	Page
A	Set of arms	17
a	An arm	17
G	Set of groups	17
g	A group	17
$\mu_{a,g}$	Highest mean reward of an arm in group g.	17
a_g^*	The arm with the highest mean reward in group g	17
μ_g^*	Highest mean reward of an arm in group g	17
$D_{a,g}$	Disappointment of arm a in group g.	18
C_a	Cost of disappointment of arm a	18
δ	Failure probability of MD-LUCB algorithm	18
$r_g(t)$	How many rounds of INC-EG have been called for group g at time t.	20
$\hat{\nu}_g(r)$	Empirical mean reward of the highest mean reward arm in group g in round r	20
$U_g(r)$	Upper confidence bound of $\hat{\nu}_g(r)$ after r rounds.	20
$L_g(r)$	Lower Confidence bound of $\hat{\nu}_g(r)$ after r rounds.	20
$N_{a,g}(t)$	Number of pulls from MD-LUCB from arm a in group g at time t	22
$\epsilon_{a,g}(t)$	Confidence radius for $\hat{\mu}_{a,g}(t)$.	23
$U_{a,g}(t)$	Upper confidence bound on the mean reward from arm a in group g at time t	23
$L_{a,g}(t)$	Lower confidence bound on the mean reward from arm a in group g at time t	23

Notation	Description	Page
$\hat{D}_{a,g}(t)$	Empirical disappointment of arm a in group g at time t.	23
$U_{D_{a,g}}(t)$	Upper confidence bound on the disappointment of arm a in group g at time t	23
$L_{D_{a,g}}(t)$	Lower confidence bound on the disappointment of arm a in group g at time t	23
$g_a(t)$	Group that maximizes potential disappointment for arm a at time t	23
$\hat{C}_a(t)$	Empirical cost of disappointment of arm a at time t	23
$U_{C_a}(t)$	Upper confidence bound on the cost of disappointment of arm a in group g at time t	23
$L_{C_a}(t)$	Lower confidence bound on the cost of disappointment of arm a in group g at time t	23
t_s	The last time step of epoch s	24
$a_1(t_{s-1})$	Best arm at time t_{s-1}	24
$a_2(t_{s-1})$	Second best arm at time t_{s-1}	24
$g_{a_1(t_{s-1})}(t_{s-1})$	The group that maximizes potential disappointment for arm $a_1(t)$ at time t_{s-1}	24
$g_{a_2(t_{s-1})}(t_{s-1})$	The group that maximizes potential disappointment for arm $a_2(t)$ at time t_{s-1}	24
$P_1(t_{s-1})$	The arm-group pair we associate as the best arm at time t_{s-1}	24
$P_2(t_{s-1})$	The arm-group pair we associate as the second best arm at time t_{s-1}	24
δ_E	Failure probability of event $\mathcal{E}$	28
δ_H	Failure probability of event $\mathcal{H}$	28
τ_δ	Stopping time of MD-LUCB	31

Notation	Description	Page
$N(t)$	Number of total pulls from MD-LUCB at time t	32
$M(t)$	Number of total pulls from INC-EG at time t	32
ρ_{δ_E}	Stopping epoch of MD-LUCB	34
ϵ_g	The level of precision needs around the highest mean reward from group g at stopping time τ_δ.	37

Bibliography

Jean-Yves Audibert, Sébastien Bubeck, and Rémi Munos. Best arm identification in multi-armed bandits. In *COLT*, pages 41–53. Citeseer, 2010.

Djallel Bouneffouf, Irina Rish, and Charu Aggarwal. Survey on applications of multi-armed and contextual bandits. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE, 2020.

Sébastien Bubeck, Rémi Munos, and Gilles Stoltz. Pure exploration in multi-armed bandits problems. In *International conference on Algorithmic learning theory*, pages 23–37. Springer, 2009.

Deepayan Chakrabarti, Ravi Kumar, Filip Radlinski, and Eli Upfal. Mortal multi-armed bandits. *Advances in neural information processing systems*, 21, 2008.

Yifang Chen, Alex Cuellar, Haipeng Luo, Jignesh Modi, Heramb Nemlekar, and Stefanos Nikolaidis. Fair contextual multi-armed bandits: Theory and experiments. In *Conference on Uncertainty in Artificial Intelligence*, pages 181–190. PMLR, 2020.

Audrey Durand, Charis Achilleos, Demetris Iacovides, Katerina Strati, Georgios D Mitsis, and Joelle Pineau. Contextual bandits for adapting treatment in a mouse model of de novo carcinogenesis. In *Machine learning for healthcare conference*, pages 67–82. PMLR, 2018.

Eyal Even-Dar, Shie Mannor, Yishay Mansour, and Sridhar Mahadevan. Action

elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *Journal of machine learning research*, 7(6), 2006.

Aurélien Garivier, Emilie Kaufmann, and Wouter M Koolen. Maximin action identification: A new bandit framework for games. In *Conference on Learning Theory*, pages 1028–1050. PMLR, 2016.

Kevin Jamieson, Matthew Malloy, Robert Nowak, and Sébastien Bubeck. lil'UCB: An optimal exploration algorithm for multi-armed bandits. In *Conference on Learning Theory*, pages 423–439. PMLR, 2014.

Shivaram Kalyanakrishnan, Ambuj Tewari, Peter Auer, and Peter Stone. PAC subset selection in stochastic multi-armed bandits. In *ICML*, volume 12, pages 655–662, 2012.

Zohar Karnin, Tomer Koren, and Oren Somekh. Almost optimal exploration in multi-armed bandits. In *International Conference on Machine Learning*, pages 1238–1246. PMLR, 2013.

Emilie Kaufmann, Olivier Cappé, and Aurélien Garivier. On the complexity of best-arm identification in multi-armed bandit models. *The Journal of Machine Learning Research*, 17(1):1–42, 2016.

Aria Khademi and Vasant Honavar. Algorithmic bias in recidivism prediction: A causal perspective (student abstract). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13839–13840, 2020.

Tze Leung Lai, Herbert Robbins, et al. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22, 1985.

Natalia Martinez, Martin Bertran, and Guillermo Sapiro. Minimax pareto fairness: A multi objective perspective. In *International Conference on Machine Learning*, pages 6755–6764. PMLR, 2020.

Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)*, 54(6):1–35, 2021.

Ziad Obermeyer, Brian Powers, Christine Vogeli, and Sendhil Mullainathan. Dissecting racial bias in an algorithm used to manage the health of populations. *Science*, 366 (6464):447–453, 2019.

Sandeep Pandey, Deepak Agarwal, Deepayan Chakrabarti, and Vanja Josifovski. Bandits for taxonomies: A model-based approach. In *Proceedings of the 2007 SIAM International Conference on Data Mining*, pages 216–227. SIAM, 2007.

Vishakha Patil, Ganesh Ghalme, Vineet Nair, and Y Narahari. Achieving fairness in the stochastic multi-armed bandit problem. *J. Mach. Learn. Res.*, 22:174–1, 2021.

Herbert Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58(5):527–535, 1952.

Samira Samadi, Uthaipon Tantipongpipat, Jamie H Morgenstern, Mohit Singh, and Santosh Vempala. The price of fair PCA: One extra dimension. *Advances in neural information processing systems*, 31, 2018.

Shubhanshu Shekhar, Greg Fields, Mohammad Ghavamzadeh, and Tara Javidi. Adaptive sampling for minimax fair classification. *Advances in Neural Information Processing Systems*, 34, 2021.

Weiwei Shen, Jun Wang, Yu-Gang Jiang, and Hongyuan Zha. Portfolio choices with orthogonal bandit learning. In *Twenty-fourth international joint conference on artificial intelligence*, 2015.

Patrick D Smith and Maureen H McDonough. Beyond public participation: Fairness in natural resource decision making. *Society & natural resources*, 14(3):239–249, 2001.

Heather P Whitley and Wesley Lindsey. Sex-based differences in drug activity. *American family physician*, 80(11):1254–1258, 2009.

Robert Williamson and Aditya Menon. Fairness risk measures. In *International Conference on Machine Learning*, pages 6786–6797. PMLR, 2019.