

**ADVERSARIAL ATTACK AND ROBUST LEARNING IN MULTI-ARM BANDIT
PROBLEMS**

A Dissertation
Presented to
The Academic Faculty

By

Yinglun Xu

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
College of Engineering
School of Electrical and Computer Engineering

Georgia Institute of Technology

May 2021

© Yinglun Xu 2021

**ADVERSARIAL ATTACK AND ROBUST LEARNING IN MULTI-ARM BANDIT
PROBLEMS**

Thesis committee:

Dr. Jacob Abernethy
Computer Science
Georgia Institute of Technology

Dr. Vidya Muthukumar
Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Mark Davenport
Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Swati Gupta
Industrial and System Engineering
Georgia Institute of Technology

Date approved: April, 2021

ACKNOWLEDGMENTS

I want to thank my advisor Jacob Abernethy. He gave me a chance to do research in machine learning and I learned a lot from him. He is a smart professor and an honorable man that I admire. I also want to thank my coworkers. Bhuvesh Kumar and Thodoris Lykouris have helped me a lot with my research and my application, which I am truly grateful. Yeo Joon Youn, Zihao Hu, and Junkun Wang are all great friends and I really enjoy the time we spent together. My dear friends and mentors, I will miss you.

TABLE OF CONTENTS

Acknowledgments	iii
List of Tables	vii
List of Figures	viii
List of Acronyms	ix
Chapter 1: Introduction to Bandit Learning and Adversary	1
1.1 Bandit Learning Setting	2
1.2 Bandit Algorithms	3
1.2.1 Greedy algorithms	3
1.2.2 Optimistic in the face of uncertainty algorithms (OFU)	6
1.2.3 Thompson sampling algorithms	8
1.3 Adversaries in Bandit Learning	9
1.3.1 Contaminated setting and adversary from data corruption	9
1.3.2 Strategic setting and adversary from agents incentives	10
Chapter 2: Adversarial Attack on Stochastic Bandit Algorithms	12
2.1 Stochastic setting with data corruption	13
2.2 Criteria for attacks and algorithms	14

2.3	Mean based algorithms	14
2.4	Attack method	17
2.5	Vulnerability analysis on mean based algorithms	18
2.6	Attack on stochastic bandit algorithms	20
2.7	Experiment results	22
2.7.1	UCB1 Algorithm	23
2.7.2	ϵ -greedy Algorithm	24
2.7.3	Thompson sampling Algorithm	24
2.8	Observation free attack oblivious to mean rewards	26
2.9	Adversary that is not oblivious to algorithm's decision	27
Chapter 3: Bandit Algorithms Robust to Both Adversaries		28
3.1	Contextual PPC Auction Setting	28
3.2	More Notions for Contextual PPC Auctions	30
3.3	Truthful Mechanisms in Contextual PPC Auction	33
3.4	ϵ -Greedy for Contextual Bandit Mechanisms	36
3.5	Without the condition that advertisers may observe context	40
3.6	Robustness against data corruption	42
3.6.1	Warm up: corruption robustness without contexts	42
3.6.2	Corruption in Contextual Setting	43
Chapter 4: Algorithms Robust Against Data Corruption		45
4.1	Algorithm Decomposition: Estimator and Policy Maker	45
4.2	Estimators that minimize bias	46

4.3	Problems and Solutions	47
4.3.1	High Variance Problem	47
4.3.2	Solutions	50
Chapter 5:	Conclusion	52
Appendices	54
References	71

LIST OF TABLES

2.1	Corruption level parameters for different algorithms	22
-----	--	----

LIST OF FIGURES

2.1	Empirical behaviors of arms in different algorithms. (a), (b) is for UCB algorithm; (c), (d) is for ϵ -greedy algorithm; (e), (f) is for Thompson sampling algorithm. (a), (c), (e) focus on the time when the corruption is active. (b), (d), (f) focus on the time when the corruption stops.	25
2.2	The number of rounds the optimal arm get picked $n_{i^*}^t$ before the corruption ends. (a) is for UCB algorithm, (b) is for ϵ -greedy algorithm, and (c) is for Thompson sampling algorithm.	25
3.1	For arm a_1 , (a) is the relation between its bid and the probability that it receives a click, (b) is the payment for different bids if it is clicked, and (c) each line represent the utility it can get for different bid with an agent value.	36
3.2	When the payment rule is not exactly the same as the truthful payment rule, the relation between the bid for agent with different agent value.	37
4.1	The probability the sub-optimal arm will be picked for each iteration. The top plot is for special UCB algorithm; the bottom plot is for UCB1 algorithm.	49

SUMMARY

In this paper we study bandit learning problem with adversaries. Bandit learning setting is a typical sequential decisions making setting whose biggest challenge is to balance the exploration and exploitation. Models based on bandit learning setting are widely applied in a variety fields of applications, like marketing and advertising. However, the original bandit learning setting simplify the reward generation pattern as stochastic, which is not accurate in some important cases. In fact, there exists adversaries in a lot of important applications that severely influence the performance of most existing bandit learning algorithms. In this paper we consider two typical types of adversaries: incentive adversary and data corruption adversary. Incentive adversaries come from the fact that the bandits for the algorithm to deal with have their own incentives. An example is in an online advertising business, a company needs to sell its advertising slots to advertisers repeatedly, which could be a typical bandit problem except for the fact that advertisers also want to maximize their own interest and hence bid strategically. Data corruption adversaries come from a malicious third party that is able to manipulate data directly. An example is in recommendation systems, a platform recommend shops to users based on users' feedback, but some of the feedback may be generated by the shops themselves, i.e, false feedback. First we will show that most bandit algorithms for the original setting without adversaries are prone to both types of adversaries. In particular, proving that most algorithms are prone to data corruption adversary is of more interest. We characterize an important feature of most bandit algorithms, construct several instances with different adversary strategy, and show that all algorithms with this feature will fail in at least one of these cases. Particularly, we purpose an attack that is oblivious to algorithm's behavior and show that the attack can completely manipulate many typical bandit algorithms. Next we purpose an algorithm that is simultaneously robust to both types of adversaries and efficiently utilize side information (context) in the mean while. At last we study the case where only data corruption adversary exists, which has a higher requirement

on performance than the setting where both adversaries exist. We think the new challenge that the data corruption adversary bring to the bandit learning problem is bias on estimates oblivious to the algorithm. Based on this thinking we focus the estimators that is able to minimize the bias, and explore the conditions when such estimators could work. We show that these conditions can be found in both existing algorithms robust to data corruption adversary. In the end we come up with an open question whether there exist an estimator that strike a balance between bias and variance such that it can be combined with other bandit algorithms to make them robust to data corruption adversary.

CHAPTER 1

INTRODUCTION TO BANDIT LEARNING AND ADVERSARY

In the real world, people are making decisions all the time based on the information and knowledge they have. Whenever a decision is made, the consequence will be revealed, then one receives rewards from the decision. Intuitively, one could just make the optimal decision based on the current information we have. Such simple strategy makes perfect sense in the scenario where you only need to make a decision for once, or you can observe the consequence of all decisions no matter what decision you make. However, not all scenarios satisfies the such properties. In many important cases, a series of decisions need to be made sequentially, and you only observe the consequence of the decision you make. For example, when a company needs to make decisions on the price to sell its product, only the sale result for the decided price can be observed, and the company cannot observe the potential results if it had decided another price. In such cases, the fundamental trade-off between explore and exploit arises [1]. On one hand, if we greedily make the optimal decision based on current information, the interest of the current round is maximized with regards to our information; on the other hand, if we try a decision which we barely know before, we will get more information for the whole picture, thus we can make better decisions in the future. The former behavior corresponds to exploit, and the later behavior corresponds to explore. These two usually contradict each other, that is, usually usually one wouldn't make the empirically best decision if he wants to know more about every decisions. So it is crucial to find the balance between explore and exploit such that we can maximize the accumulated rewards over all decisions. The explore-exploit trade-off is one of the fundamental difficulties in reinforcement learning [2], influencing a lot of important control problems like autonomous driving [3], marketing and advertising [4], robotics [5], etc. In this thesis, we focus on stochastic bandit learning setting [6], which can be considered as the simplest reinforcement

learning task. To begin with, we formally introduce bandit setting and the corresponding bandit learning problem.

1.1 Bandit Learning Setting

The bandit problem is essentially a sequential decision making problem. There is an actions (or decisions, arms) set \mathcal{A} with cardinality K . A principal (or a bandit algorithm \mathcal{B}) needs to repeatedly decide an action $a \in \mathcal{A}$ over T time steps. Every time an action is decided, the principal would receive a reward $r \in [0, 1]$. Let the reward at time t as r^t , the goal of the principal is to maximize the accumulated reward $\sum_{t=1}^T r^t$. At any time t , the reward r^t depends on the decided action I^t . In the stochastic reward setting, each action $a \in \mathcal{A}$ is related to a fixed reward distribution $D(a)$, and if a is selected, the reward is drawn i.i.d from $D(a)$. Let $\mu(a)$ be the expectation of $D(a)$, then ideally the principal would like to always select the action with the highest expectation reward $a^* = \operatorname{argmax}_a \mu(a)$, which gives the benchmark accumulated reward $T \cdot \mu(a^*)$. To measure how good the bandit algorithm \mathcal{B} is, we introduce the concept of regret which is defined as the gap between the actual and benchmark accumulated reward $R(T) = T \cdot \mu(a^*) - \sum_{t=1}^T r^t$, and the goal of the algorithm is to minimize the expected regret $\mathbb{E}[R(\mathcal{T})] = T \cdot \mu(a^*) - \sum_{t=1}^T \mu(I^t)$. More specifically, the process that a bandit algorithm \mathcal{B} interacts with actions set \mathcal{A} for T rounds is as follows.

For each round $t = 1, 2, \dots, T$:

1. Environment generates a reward profile $\mathbf{r} \in [0, 1]^K$, and for any $a \in \mathcal{A}$, $r(a)$ is drawn from the fixed distribution $D(a)$.
2. \mathcal{B} selects an action $I^t \in \mathcal{A}$, and receives reward $r^t = \mathbf{r}(I^t)$.

Having presented the basic bandit learning setting, next we will introduce different ideas of bandit algorithms that performs well in this setting.

1.2 Bandit Algorithms

On the high level, the ideas of most bandit algorithms can be categorized into one of the three major ideas:

1. Greedy Algorithms [7, 8, 9, 10]
2. Optimistic in the face of uncertainty Algorithms [11, 12, 13, 14, 15]
3. Thompson Sampling Algorithms [16, 17, 18, 19, 20]

1.2.1 Greedy algorithms

The idea of greedy algorithms is that for most of the time or with high probability, the algorithm selects the action that is empirically the best, while for the rest of the time or probability, uniformly at random explores an action from the whole action set. With enough exploration for all actions, the empirical estimation on the reward for any action should be close to its true value, hence the greedy action is very likely to be the optimal action or at least very close to it. There are two typical greedy algorithms: explore-then-commit and ϵ -greedy [9], which are formally described as follows.

Greedy algorithms are simple and easy to be implemented, thus has been widely applied in reinforcement learning algorithms [21, 22, 23]. Convenient as it is, greedy algorithms usually explore more than necessary. The guarantee for regret in the worst cases is $O(T^{2/3})$ for both algorithms mentioned above, while the lower bound for regret in the worst case can be proved to be $O(T^{1/2})$. Moreover, let Δ be the gap between the expected reward of the best and second best arm, later we will show algorithms which can guarantee $O(\log T/\Delta)$ regret while greedy algorithms still only guarantee $O(T^{2/3})$. The problem for greedy algorithms is that at some point enough information has already been observed to conclude which arm must not be the best, but still the algorithm explore such arms as frequently as always. An easy improvement for greedy algorithm is active elimination [24], where the algorithm will

Algorithm 1: Explore then commit Algorithm

Parameters : Number of rounds T , Arms set \mathcal{A} which has cardinality K , Explore length L

Initialize : $\hat{\boldsymbol{\mu}} \leftarrow \mathbf{0}$, $\mathbf{n} \leftarrow \mathbf{0}$ where $\hat{\boldsymbol{\mu}} \in \mathbb{R}^K$ and $\mathbf{n} \in \mathbb{N}^K$ represents empirical mean and number of rounds getting selected for arms

```
1 for  $t = 1, \dots, T$  do
2   if  $t \leq L$  then
3      $I^t \leftarrow t \bmod K$       /* Explore every arm in rotation */
4     Receive reward  $r^t$ 
5      $\hat{\boldsymbol{\mu}}(I^t) \leftarrow \frac{\hat{\boldsymbol{\mu}}(I^t) \cdot \mathbf{n}(I^t) + r^t}{\mathbf{n}(I^t) + 1}$ 
6      $\mathbf{n}(I^t) \leftarrow \mathbf{n}(I^t) + 1$       /* Update Statistics for the
       selected arm */
7   end
8   else
9      $I^t \leftarrow \operatorname{argmax}_a \hat{\boldsymbol{\mu}}(a)$  /* Select the empirically best arm
       greedily */
10    Receive reward  $r^t$ 
11  end
12 end
```

stop explore an arm if it is very confident that some arm must be better than this arm. To formally present the algorithm, we need to introduce the concept of confidence interval [25]. Denote the expected reward of an arm as μ and an estimate on the reward (usually the empirical mean) as $\hat{\mu}$. A confidence interval centered at $\hat{\mu}$ with width $2 \cdot w$ should satisfy the following property: with probability at least $1 - \delta$, the expected reward is within the confidence interval, that is $\mu \in [\hat{\mu} - w, \hat{\mu} + w]$. The confidence width can be given by concentration inequalities [26], like Hoeffding inequality [27]. If $\hat{\mu}$ is the empirical mean calculated from N samples, then by Hoeffding inequality, we have $w = \sqrt{\frac{\log 1/\delta}{N}}$.

It is easy to conclude that if the lower bound of the confidence interval of one arm is greater than the higher bound of that of another arm, then this arm is very likely to be better than the other. This is the core idea of active elimination to decide at which point it should stop exploring an arm, and the formal description of active elimination algorithm is as follows.

By setting appropriate δ , e.g., $1/T^2$, the expected regret for active elimination algorithm

Algorithm 2: ϵ -greedy Algorithm

Parameters : Number of rounds T , Arms set \mathcal{A} which has cardinality K , Explore rate ϵ

Initialize : $\hat{\boldsymbol{\mu}} \leftarrow \mathbf{0}$, $\mathbf{n} \leftarrow \mathbf{0}$ where $\hat{\boldsymbol{\mu}} \in \mathbb{R}^K$ and $\mathbf{n} \in \mathbb{N}^K$ represents empirical mean and number of rounds getting selected for arms

```
1 for  $t = 1, \dots, T$  do
2    $\ell^t \leftarrow \begin{cases} 1 & \text{w.p. } \epsilon \\ 0 & \text{otherwise} \end{cases}$            /* Explore or exploit */
3   if  $\ell^t = 1$  then
4      $I^t \leftarrow j$  uniformly at random for  $j \in [K]$  /* Randomly explore an
      arm */
5     Receive reward  $r^t$ 
6      $\hat{\boldsymbol{\mu}}(I^t) \leftarrow \frac{\hat{\boldsymbol{\mu}}(I^t) \cdot \mathbf{n}(I^t) + r^t}{\mathbf{n}(I^t) + 1}$ 
7      $\mathbf{n}(I^t) \leftarrow \mathbf{n}(I^t) + 1$            /* Update Statistics for the
      selected arm */
8   end
9   else
10     $I^t \leftarrow \operatorname{argmax}_a \hat{\boldsymbol{\mu}}(a)$  /* Select the empirically best arm
      greedily */
11    Receive reward  $r^t$ 
12     $\hat{\boldsymbol{\mu}}(I^t) \leftarrow \frac{\hat{\boldsymbol{\mu}}(I^t) \cdot \mathbf{n}(I^t) + r^t}{\mathbf{n}(I^t) + 1}$ 
13     $\mathbf{n}(I^t) \leftarrow \mathbf{n}(I^t) + 1$ 
14  end
15 end
```

Algorithm 3: Active elimination algorithm

Parameters : Number of rounds T , Arms set \mathcal{A} which has cardinality K , confidence width multiplier α

Initialize : $\hat{\boldsymbol{\mu}} \leftarrow \mathbf{0}$, $\mathbf{n} \leftarrow \mathbf{0}$ where $\hat{\boldsymbol{\mu}} \in \mathbb{R}^K$ and $\mathbf{n} \in \mathbb{N}^K$ represents empirical mean and number of rounds getting selected for arms. Active set $\mathcal{S} \leftarrow \{1, \dots, K\}$ which includes all arms

```
1 for  $t = 1, \dots, T$  do
2    $I^t \leftarrow j$  uniformly at random for  $j \in \mathcal{S}$ 
3   Receive reward  $r^t$ 
4    $\hat{\boldsymbol{\mu}}(I^t) \leftarrow \frac{\hat{\boldsymbol{\mu}}(I^t) \cdot \mathbf{n}(I^t) + r^t}{\mathbf{n}(I^t) + 1}$ 
5    $\mathbf{n}(I^t) \leftarrow \mathbf{n}(I^t) + 1$ 
6   for  $a \in [S]$  do
7     for  $a' \in [S], a' \neq a$  do
8       if  $\boldsymbol{\mu}(a) + \alpha \cdot \sqrt{\frac{\log(T)}{\mathbf{n}(a)}} < \boldsymbol{\mu}(a') - \alpha \cdot \sqrt{\frac{\log(T)}{\mathbf{n}(a')}}$  then
9          $\mathcal{S} \leftarrow \mathcal{S}/a$  /* arm  $a$  is highly likely to be worse
10        than arm  $a'$ , so deactivate arm  $a$  from  $\mathcal{S}$  */
11       end
12     end
13 end
```

is bound by $O(\log T/\Delta)$.

There are also other ways to improve the performance of greedy algorithms. For example, one can adopt ϵ greedy algorithm by using dynamic explore rate ϵ that decays with time [9].

1.2.2 Optimistic in the face of uncertainty algorithms (OFU)

The idea of OFU algorithms adopts the concept of confidence interval. In OFU algorithms, each arm is represented by the upper bound of its confidence interval (or UCB, short for upper confidence bound), and the algorithm will always pick the arm with the highest upper bound. These algorithms are also called UCB algorithms, and they differ from each other by the way they use to calculate the confidence interval half width w . As the simplest example, UCB1 algorithm [11] runs as follow.

The lower bound of regret for UCB1 algorithm is $O(\log T/\Delta)$. There are other UCB

Algorithm 4: UCB1 Algorithm

Parameters : Number of rounds T , Arms set \mathcal{A} which has cardinality K , confidence width multiplier α

Initialize : $\hat{\boldsymbol{\mu}} \leftarrow \mathbf{0}$, $\mathbf{n} \leftarrow \mathbf{0}$ where $\hat{\boldsymbol{\mu}} \in \mathbb{R}^K$ and $\mathbf{n} \in \mathbb{N}^K$ represents empirical mean and number of rounds getting selected for arms

```
1 for  $t = 1, \dots, T$  do
2   if  $t \leq K$  then
3      $I^t \leftarrow K$  /* Pick each arm for once at the beginning
4       */
5     Receive reward  $r^t$ 
6      $\boldsymbol{\mu}(I^t) \leftarrow r^t$ 
7      $\mathbf{n}(I^t) \leftarrow 1$ 
8   end
9   else
10     $I^t \leftarrow \operatorname{argmax}_a \{ \boldsymbol{\mu}(a) + \alpha \sqrt{\frac{\log(T)}{\mathbf{n}(a)}} \}$  /* Pick the potentially
11      best arm */
12    Receive reward  $r^t$ 
13     $\hat{\boldsymbol{\mu}}(I^t) \leftarrow \frac{\hat{\boldsymbol{\mu}}(I^t) \cdot \mathbf{n}(I^t) + r^t}{\mathbf{n}(I^t) + 1}$ 
14     $\mathbf{n}(I^t) \leftarrow \mathbf{n}(I^t) + 1$ 
15  end
16 end
```

algorithms like KL-UCB which uses more sophisticated way to find the upper bound and achieve better regret bound (in terms of logarithm factor and constant factor). Just like greedy algorithms, the idea of OFU is easy to be extended to more sophisticated setting and works well [14, 15]. For example, Lin-UCB algorithm for contextual bandit problem has already been shown to be successful in online article recommendation applications [14]. [15] even develop an UCB algorithm based on a neural network model.

1.2.3 Thompson sampling algorithms

The idea of Thompson sampling algorithms also utilize confidence interval in some way, but unlike deterministic OFU (UCB) algorithms that always pick the potentially best arm, it picks arm in a random way and with more sophisticated probability distribution than ϵ -greedy. In Thompson sampling algorithms [17, 20], each arm a is represented by a distribution $D(a)$ which is the algorithm's belief on its mean reward distribution. Each round, (in practice) the algorithm will draw a value from distributions of each arm, then pick the arm with the highest value. The distribution will be continuously updated whenever a new reward from an arm is received. A general form of Thompson Sampling algorithm is described as follows.

Algorithm 5: Thompson sampling algorithm

Parameters : Number of rounds T , Arms set \mathcal{A} which has cardinality K , Prior distribution \mathbf{D} for each arm

Initialize : $\boldsymbol{\theta} \leftarrow \mathbf{0}$, where $\boldsymbol{\theta} \in \mathbb{R}^K$ is the parameters to represent arms

```

1 for  $t = 1, \dots, T$  do
2   | for  $a = 1, \dots, K$  do
3   |   |  $\theta(a)$  Randomly drawn from distribution  $D(a)$ 
4   | end
5   |  $I^t \leftarrow \operatorname{argmax}_a \{\theta(a)\}$ 
6   | Receive reward  $r^t$ 
7   | Update posterior distribution  $D(I^t)$  based on  $r^t$ 
8 end

```

Thompson sampling algorithms differs from each other by the forms of the distribution they use. Some common examples include Bernoulli distribution [17] and Gaussian distri-

bution [20]. These algorithms have $O(\log T/\Delta)$ upper bound on the expected regret, and in practice it works as well as the OFU algorithms. Essentially, Thompson sampling determines a relatively sophisticated probability for each arm to be picked, which differs from Greedy and OFU algorithms fundamentally. In practice, it may not always be possible to update the distribution easily in some problems, so people develop many other algorithms sharing the same features and requiring less computation power [28, 29, 30, 31]. For example, there are various kinds of boot-strapping algorithms [28], and Random-UCB algorithms [30], which determines the probability distribution over arms in ways different to Thompson sampling.

1.3 Adversaries in Bandit Learning

As we have already presented, the basic stochastic bandit learning problems have a lot of good solutions with strong theoretic guarantees. Can they directly be applied to the applications in the real world? In some circumstances where the assumption that rewards are randomly generated from an environment almost hold, the answer is yes and algorithms are indeed implemented to solve the problems in the real world [32, 33]. However, the situation in the real world is not always ideal, and in the cases where certain kinds of adversaries [34, 35] exist in the reward generation or observation, the algorithms mentioned before will fail tragically [36]. In this section we will introduce two settings as extensions to stochastic bandit setting. Each of them includes a form of adversary which makes the setting more realistic in real world applications and more difficult to solve.

1.3.1 Contaminated setting and adversary from data corruption

The contaminated setting consider potential data corruption in the rewards [36, 37, 38, 34, 39]. That is, the reward it receives at each round may not be the true reward randomly generated by the environment, and instead be an artificial value intentionally chose by a third party with probably malicious purpose, which is the adversary in this setting. As an example, consider an online recommendation problem, a platform like yelp needs to

recommend one of several restaurants to a customer, and then receives the feedback from the customer. This is a typical bandit learning setting where the platform is the principal, the restaurants are the arms, and the feedback's from the user are the reward. It is reasonable to assume that users' feedback's on a restaurant is random, which seems to be almost perfectly modeled by stochastic bandit learning setting. However, in the real world, there also exists fake feedback's which are generated by a malicious agent rather than a random customer. For example, one restaurant could leave fake bad comments on its competitor in hope to bring itself advantages, or it could manipulate the customers to leave fake good comments by bribing them (e.g, free dishes). Although it is also true that the ratio of fake feedback to all feedback is very small, the algorithms which works perfectly well in the stochastic setting may have no guarantee on regret in this setting because of this adversary even if with very limited power. In section 2 we will formally describe this setting and show a strategy for the adversary which could defeat most stochastic bandit algorithms.

1.3.2 Strategic setting and adversary from agents incentives

The strategic setting of Bandit problem is to consider the incentives of arms (or agents in this setting) in addition to the stochastic bandit problem [35, 40, 41, 42]. In this setting, the principal is interacting with players with their own interest rather than an environment indifferent to the principal's actions. The interest of the principal will not only depend on the its own decisions on arms, but also depend on the behaviors of these agents, which are the adversary in this setting. An important example of this setting is pay-per-click (PPC) auction which is widely applied in online advertising business [43]. A website like Google or Facebook sells advertisement slots to bidders (who is also the advertisers), and charge a certain price if the advertisement is clicked by a user. For the website, the reward from each advertiser is the click through rate of their advertisement times the price to charge, which depends on their bids. For the bidders, they also have rewards which are the interest they can get if their ad is clicked minus the price they pay, and the interests are private to

themselves only. Ideally, the website wants the bidders to bid their interest truthfully so that it could set a reasonable price to charge, but since the bidders also want to maximize their own rewards, they would only bid their interest truthfully if and only if this is or at least close to the optimal strategy for their own goal. For the principal in this problem, it has to face constraints on their learning algorithm due to the incentives of the agents. We will formally describe this setting in detail in section 3, where we will also provide a solution to the problem. In addition, an important fact we have to point out in PPC auction is that aside from the incentives adversary, there is also a famous problem called click fraud [44] that one should never overlook, which can be think of an adversary with data corruption. So it is practical to consider both adversaries in PPC auctions.

CHAPTER 2

ADVERSARIAL ATTACK ON STOCHASTIC BANDIT ALGORITHMS

Before thinking about new algorithms which can work well in the setting with adversaries, first we need to check whether the existing algorithms work or not. It may also be beneficial to think about how those algorithms don't work so that the problem is better understood and we could design algorithms robust to adversary easier.

First we briefly introduce the case of incentive adversaries. We restrict the scope of robust algorithms to the truthful algorithms which ensures that the best bidding strategy for every arm is to bid their own value truthfully. It has been proved that a necessary and sufficient condition for an algorithm to be truthful is *exploration separated* [35]. Exploration separated condition divides all rounds into explore rounds and exploit rounds. It requires that the decisions on explore rounds are independent of the bids (which implies independent of the reward), and the decisions on exploit rounds can only depend on the results from explore rounds. It is clear that algorithms like UCB and Thompson sampling are not exploration separated, because the decisions they make at each round are based on the rewards estimated from all rounds before. In this sense, greedy algorithms are very suitable for this problem. As a matter of fact, a simple explore then commit algorithm is robust against incentives. With a focus on truthful algorithms, we are only concerned about whether the best strategy for the agent is bidding truthful nor not, so usually people focus on the truthful algorithms rather than the behavior of the agents. Outside truthful algorithm, the behavior of the agents will be more interesting, and the interaction between the principal and the agents is more versatile [40, 45]. In this thesis we only focus on truthful algorithms, and in section 3 we will consider and solve a more meaningful setting of strategic bandit setting where context are also considered.

The focus of this chapter is data corruption attack. We will present an attack method

called observation free attack that can ensure big regret for most known algorithms designed for stochastic bandit learning problems, which are generalized in a class of algorithms called mean based algorithms. Through the attack, we not only are certain about the fact that most existing bandit algorithms are not reliable in the corrupted setting, but also gain some insight in why these algorithms fail.

First we formally introduce the bandit setting with data corruption in details. Then we will introduce our attack method: observation attack, and the definition of the class of algorithms that conclude most bandit algorithms: mean based algorithm. At last we will analyze the effect of observation free attack on mean based algorithms and some specific algorithms, while corresponding experiment results will also be presented.

2.1 Stochastic setting with data corruption

The goal of the adversary is to manipulate the behavior of the algorithm by corrupting the reward r generated by the environment. In each round t , when the reward profile r^t is generated by the environment, the adversary can apply corruption to the reward, and the reward becomes r_C^t . The adversary is limited by the number of rounds that it is allowed to change to rewards. This number is denoted as C and we call such adversary has corruption level of C . Another important constraint on the adversary is that it cannot observe the arm actually picked by the algorithm before it decides whether to corrupt this round or not. This constraint makes attack more meaningful in terms of understanding robust algorithm since without it, the attack [36, 38] would be trivial and too powerful as will be illustrated in the end of this chapter. Formally, the process of stochastic bandit with data corruption can be summarized as follow.

For each round $t = 1, 2, \dots, T$:

1. Environment generates a reward profile $r^t \in [0, 1]^K$, and for any $a \in \mathcal{A}$, $r(a)$ is drawn from the fixed distribution $D(a)$.

2. Adversary observes r , and modifies it to r_C^t .
3. \mathcal{B} selects an action $I^t \in \mathcal{A}$, and receives reward $r^t = r^t(I^t)$.

2.2 Criteria for attacks and algorithms

To measure how well an algorithm works in the setting with data corruption, we introduce the definition of *vulnerable* algorithms.

Definition 1 (Vulnerable). *We say a bandit algorithm is vulnerable if there exists an instance and an adversary such that the adversary with $C = o(T)$ corruption level can induce linear regret $R(T) = \Omega(T)$ on the bandit algorithm in expectation.*

If an algorithm is not vulnerable, then it can always guarantee sublinear regret $R(T) = o(T)$ in expectation under any adversary with sublinear corruption level $C = o(T)$, and we say such algorithm is *robust*.

For the attack methods, a powerful one should guarantee a stronger consequence than linear regret in expectation when attacking a specific algorithm. To characterize how “powerful” an attack method is, we measure how the attack manipulate an bandit algorithm. Typically, the goal of the attack is to guarantee that with high probability, the algorithm will select a target arm specified by the adversary rather than the optimal arm for all but only a few rounds. Formally, We say a bandit algorithm \mathcal{B} is *completely vulnerable* to an adversarial attack \mathcal{A} , if in most instances, with probability at least $1 - \delta(T)$, with $\delta(T) = o(1)$, the adversary can make the algorithm pick the target arm specified by the adversary for all but $o(T)$ rounds by using only $C = o(T)$ corruption level.

2.3 Mean based algorithms

Recall that one important task of thinking about adversarial attack is to show that most algorithms for stochastic setting fail in this setting. Instead of checking all existing bandit algorithm one by one through customized adversarial attacks, we want to generalize a

common feature of most bandit algorithms, and show that any algorithm having such feature must fail in at least one of several specific cases. In this section we introduce the notion of mean based algorithms. Most stochastic bandit algorithms as we have introduced fit this definition, and we will show in the next section that all mean based algorithms are vulnerable.

Recall that the major problem an algorithm need to deal with in the stochastic bandit setting is to balance the explore and exploit trade-off as we introduced at the beginning. Typically, an algorithm focus on two statistics: empirical mean and variance of the reward of each arm. The empirical means of rewards indicate that the arm with the highest empirical mean is most likely to be the optimal arm, and the variances indicate how likely this conclusion is true, or equivalently, how much one might lose if he selects the empirically best arm. An algorithm needs to pay a certain price to reduce the variance by trying out sub-optimal arms so that it could always select the optimal arm almost for sure after that. Since the variance can be calculated by the number of rounds that an arm is selected, a traditional bandit algorithm only need to focus on two statistics of arms: the empirical mean of reward and number of rounds being selected for each arm.

Before introducing the formal definition of mean based algorithm, we formally characterize the way a bandit algorithm selects arm through the notion of *policy*. In short, policy is a probability distribution over arms depends on the data received in the past. Denote $\mathcal{H}^t = \{(I^1, r_{I^1}^1), \dots, (I^{t-1}, r_{I^{t-1}}^{t-1})\}$ as the information the algorithm received by round t , where $(I^t, r_{I^t}^t)$ is the selected arm and the corresponding reward at round t . A bandit algorithm will generate a policy $\pi^t(\cdot | \mathcal{H}^t)$, where for each arm $a \in \mathcal{A}$, $\pi^t(a | \mathcal{H}^t)$ is the probability that the arm a is selected at round t .

Let $n^t(a) = \sum_{\tau=1}^t \mathbb{1}\{I^\tau = a\}$ denote the number of rounds arm i gets picked by the algorithm before round t , and let $\bar{\mu}^t(a) = \frac{\sum_{\tau=1}^t r_{I^\tau}^\tau \mathbb{1}\{I^\tau = a\}}{n^t(a)}$ be the empirical mean of the arm a by round t . The formal definition of *Mean based algorithms* is as follows.

Definition 2 (Mean based algorithms). *We say an algorithm is a mean based algorithm if it*

satisfies the following conditions:

1. Its policy depends only on the empirical means $\bar{\mu}^t$ and number times each arm i is selected n_i^t of all the arms. More specifically, for each arm a ,

$$\pi^t(a|H^t) = \pi^t(a|n^{t-1}(1), \bar{\mu}^{t-1}(1), \dots, n^{t-1}(K), \bar{\mu}^{t-1}(K))$$

2. For each arm a , the probability that it is selected is monotonically increasing in its empirical mean, i.e.

$$\pi^t(a|\dots, n^{t-1}(a), \bar{\mu}^{t-1}(a), \dots) \geq \pi^t(a|\dots, n^{t-1}(a), \bar{\mu}^{t-1}(a), \dots)$$

if $\bar{\mu}^{t-1}(a) \geq \bar{\mu}^{t-1}(a)$

3. For each arm a , the probability that it is selected is monotonically decreasing on number of selections for the empirically sub-optimal arms, i.e

$$\pi^t(a|\dots, n^{t-1}(a), \bar{\mu}^{t-1}(a), \dots) \leq \pi^t(a|\dots, n^{t-1}(a), \bar{\mu}^{t-1}(a), \dots)$$

if $n^{t-1}(a) \geq n^{t-1}(a)$ and $\bar{\mu}^{t-1}(a) < \max_{j \in \mathcal{A}} \bar{\mu}_j^{t-1}$.

In definition 2, the first condition implies that the a mean based algorithm makes decisions only depends on the two statistics, that is, empirical mean and the number of pulls of each arm. The second condition implies that if the empirical mean of the arm is higher while fixing other statistics, the probability that the arm gets selected will increase. The third condition implies that if the arm is empirically sub-optimal so far, then if the number of samples used to obtain that estimate increases, then the algorithm is more confident about the fact the arm is sub-optimal, then the probability that the arm gets selected can only decrease.

It is easy to verify that all the three types of algorithms, including UCB, Thompson sampling, and Greedy algorithms are mean based algorithms. Note that the greedy algorithms

mentioned here construct estimate on rewards based on data from all rounds rather than just explore rounds. Next, we will present our attack method, which can help us to show that all mean based algorithms are vulnerable.

2.4 Attack method

Here we introduce an attack method that we call the Observation-Free Attack which doesn't explicitly observe the behavior of the bandit algorithm while deciding how to corrupt rewards.

The attack is separated into three phases. The first phase lasts for the first C_1 rounds. The goal is to make the algorithm receive a lot of low rewards from the optimal arm so that the empirical estimate of the optimal arm's mean reward is low while the confidence over its estimate is high. To ensure that the optimal arm is picked for enough times, the attack makes all the arms look the same bad. Explicitly, we set reward to be 0 for all arms in all the rounds during the first phase.

The second phase lasts for the next C_2 rounds. The goal is to make the target arm distinguishable from the other arms. That is, it wants the algorithm to believe that the target arm is better than all other arms. To ensure this, the attack explicitly set the reward as 1 for that target arm (whose index is denoted as 1) and 0 for all other arms for all rounds during the second phase. By the end of the first two phases, the attack wants to ensure that empirical mean of all arms (especially the ones with rewards higher than the target arm) except the target arm is very low with high confidence and that the empirical mean of the target arm is much higher than the other arms.

In the third phase which is the rest of the rounds, the adversary does nothing and hopes that the algorithm selects the target arm for most of the rounds. So the attack only corrupts the initial $C_1 + C_2$ rounds and the corruption level is $C_1 + C_2$.

C_1 and C_2 are the two parameters that the adversary needs to tune based on the bandit algorithm under consideration and the rewards of the arms. To simplify the analysis, we

Algorithm 6: Observation-Free Attack

Parameters : Number of rounds T , Mean rewards vector $\bar{\mu}$, bandit algorithm A , target arm i

```
1 Compute parameters  $C_1$  and  $C_2$  for the given  $T, \bar{\mu}, A$ .
2 for  $t = 1, \dots, T$  do
3   Environment generates the reward vector  $\mathbf{r}$ 
4   if  $t \leq C_1$  then
5      $\mathbf{r} \leftarrow (0, \dots, 0)$  /* Set reward as 0 for all arms */
6   end
7   else if  $C_1 < t \leq C_1 + C_2$  then
8      $\mathbf{r} \leftarrow e_1$  /* Set reward as 0 for all arms but the
9       target arm. The reward for the target arm is 1
10      */
11   end
12   else
13      $\mathbf{r} \leftarrow \mathbf{r}$  /* No corruption is applied */
14   end
15   Bandit algorithm  $A$  selects arm  $I^t$  and receives reward  $I^t t$ 
16 end
```

assume that adversary has access to the mean reward for each of the arms at the beginning. In the case where the adversary does not have access to the mean rewards before the start of the process, we will show later that while corrupting the first few rounds, the adversary can observe the realized rewards to effectively estimate the mean rewards. Using the estimates, the adversary can set the parameters C_1 and C_2 of Algorithm in an adaptive manner.

2.5 Vulnerability analysis on mean based algorithms

In this section we will show that all mean based algorithms are vulnerable. Formally, we have the theorem below.

Theorem 1. *For any mean based bandit algorithm that achieves sub-linear regret in the absence of data-corruptions, there always exists an instance with two arms and an adversary such that the algorithm will suffer linear regret $R(T) = \Omega(T)$ in expectation.*

To prove the theorem, we constructs three instances with different attack methods such that the algorithm will suffer from linear regret in expectation in at least one of the

three instances. All instances are consists of two arms $\mathcal{A} = [2]$. In the first instance (1), $\mu^{(1)}(1) > \mu^{(2)}(2) = \mu_0$, and the attack method is the observation free attack as we just presented. In the second instance (2), $\mu^{(2)}(1) = 1, \mu^{(2)}(2) = 0$, and the attack method is to set rewards of both arms to be 0 for the first C_1 rounds with the same parameter used in the attack in instance (1). In the third instance (3), $\mu^{(3)}(1) = \mu/4, \mu^{(3)}(2) = \mu/2$, and there is no attack. We can show that if the algorithm guarantee sub-linear regret in instance (2) and (3), then it must suffer from linear regret in instance (1). In instance (1), what happened in the first $C_1 + C_2$ rounds is the same as that in instance (2). Note that in instance (2), the algorithm guarantee sub-linear regret, which means it will only select arm 1 for only a few rounds for the C_2 rounds after the first C_1 rounds. Then we can set C_2 long enough in instance (1) such that in the second phase, the algorithm will select arm 2 for many rounds and form an estimate on rewards much better than that of arm 1. Next compare the third phase in instance (1) to instance (3), at the beginning, the gap on estimates between arm 1 and 2 in instance (1) is much higher than that in instance (3). We also know that algorithm guarantee sub-linear regret in instance (3), that is, it will only select arm 1 for a few rounds. So if we choose $C_1 + C_2$ big enough such that after these many picks for arm 1, the gap between the two arms in instance (1) is still greater than that in instance (3), then we can ensure that the algorithm will not pull arm 1 more often than it does in instance (3), thus it will suffer from linear regret in instance (1).

Here we provide an intuition for why mean based algorithms are vulnerable. The estimates generated by such algorithms will suffer from big bias in the existence of data corruption. The adversary can easily manipulate these estimates with a small amount of corruption budget. When the estimates are far from the real value for most of time, the bandit algorithms which completely depend on these estimates will make poor decisions for most of the time and suffer from big regret.

Note that mean based algorithms don't have to be vulnerable to observation free attack, and we only show that the algorithms will suffer from linear regret in expectation in some

specific instances. Actually, the observation free attack is more powerful when attacking some specific mean based algorithms. In the next section we will show that some example UCB, greedy, and Thompson sampling algorithms are completely vulnerable to the attack, that is, as long as the target arm has $\Omega(1)$ mean reward, the adversary with low corruption level is able to manipulate the bandit algorithm to almost always pick the target arm with high probability.

2.6 Attack on stochastic bandit algorithms

In this section we analyze the performance of the Observation-Free attack on the three example bandit algorithms, including UCB1, ϵ -greedy, and Thompson sampling with Beta distribution, each for one of the three major types of bandit algorithms for stochastic setting. We show how we can tune the parameters C_1 and C_2 for each of the algorithm such that these algorithms are completely manipulated by the attack.

Attack on UCB1 Algorithm

The specific UCB algorithm we attack is UCB1 algorithm. UCB1 algorithm uses the confidence upper bound directly given by Hoeffding inequality as we described in chapter 1. The theorem below claims that UCB1 algorithm is completely vulnerable to observation free attack method.

Theorem 2. *When an adversary applies data corruption attack on UCB algorithm with the attack given by algorithm 6, by choosing appropriate C_1 and C_2 , with corruption level $C = O(\frac{\log T}{\mu^2})$ where μ is the mean reward of the target arm, the UCB algorithm pulls the target arm for all but $O(\frac{\log T}{\mu^2})$ rounds with probability at least $1 - 1/T$.*

The proof ideas for the analysis of attack on UCB1 algorithm and the other two algorithms mentioned later this section are similar. In the first phase of the attack, the algorithm will pick each arm for almost the same number of rounds; in the second phase of the attack, the algorithm will pick the target arms almost for every round; in the third phase of attack,

the algorithm is very confident that the target arm is better than all other arms, and will almost always select the target arm until the end. C_1 and C_2 are big enough such that even at the end, the algorithm will still believe that the target arm is the best.

Attack on greedy Algorithm

The consequence of applying observation free attack on explore-then-commit greedy algorithm is clear. By setting $C_1 = 0$ and C_2 equal to the explore length, the algorithm will always believe that the target arm is the best and always select it for all exploit rounds. The analysis on ϵ -greedy algorithm is more interesting. More specifically, the ϵ -greedy algorithm we discuss here generates estimate based on the data from every round. In the next section we will show that, surprisingly, the ϵ -greedy algorithm which generates estimate based on the explore rounds only is robust. Nevertheless, the theorem below claims that the ϵ -greedy algorithm we mentioned here is completely vulnerable to observation free attack method.

Theorem 3. *When an adversary applies data corruption attack on ϵ -greedy algorithm with the attack given by algorithm 6, by choosing appropriate C_1 and C_2 , with corruption level $C = \tilde{O}(T\epsilon + \frac{1}{\mu^2})$ where \tilde{O} hides $\log T$ terms and μ is the mean reward of the target arm, the ϵ -greedy algorithm pulls the target arm for all but $\tilde{O}(T\epsilon + \frac{1}{\mu^2})$ rounds with probability at least $1 - \frac{2K+2}{T}$.*

For ϵ -greedy algorithm, in the absence of corruption, appropriate choice of ϵ is important to ensure sub-linear regret. Since there are $T\epsilon$ rounds for explore in expectation, which contributes to unavoidable regret, an algorithm needs to ensure that $T\epsilon$ is sub-linear, therefore our corruption level is also sub-linear. To guarantee best performance in the worst case, a typical choice is $\epsilon = O(T^{2/3})$, then the corresponding level for the attack is $\tilde{O}(T^{2/3} + \frac{1}{\mu^2})$, and the target arm will be selected for all but $\tilde{O}(T^{2/3} + \frac{1}{\mu^2})$ rounds with probability at least $1 - \frac{2K+2}{T}$.

Attack on Thompson Sampling Algorithms

The specific Thompson sampling algorithm we analyze is the one using Beta Distribution for Bernoulli bandits. In this setting, the reward from picking an arm i in any round is a

Bernoulli random variable with mean μ_i .

Theorem 4. *When an adversary applies data corruption attack on the Thompson sampling algorithm with the attack given by algorithm , by choosing appropriate C_1 and C_2 , with corruption level $C = O(\frac{\log T}{\mu^2})$ where μ is the mean reward of the target arm, the Thompson sampling algorithm will pull the target arm for all but $O(\frac{\log T}{\mu^2})$ rounds with probability at least $1 - \frac{2K+1}{T}$.*

The theorems in this section conclude that as long as $\frac{1}{\mu^2}$ is sub linear in T where μ is the mean reward for the target arm, then an adversary using the observation free attack can ensure that the algorithms picks a target arm of their choice for all but $o(T)$ rounds with high probability. In the following section, we experimentally evaluate the performance of the difference algorithms when subjected to the observation free attack.

2.7 Experiment results

In this section, we experimentally explore the interaction between the observation free attack and the example algorithms selected from the last section, including UCB1, ϵ -greedy and Thompson Sampling with Beta distribution algorithms. For each algorithm, we perform the experiment on the same instance with 2 arms whose means are $\mu_1 = 0.9$ and $\mu_2 = 0.8$. In this instance, arm 1 is the optimal arm, so we set arm 2 as the target arm for the adversary. The length of each independent simulation is set as $T = 50000$ rounds, and the parameters (C_1, C_2) for each of the algorithm are set as listed by Table 2.1.

Table 2.1: Corruption level parameters for different algorithms

Algorithm	C_1	C_2
UCB	34	66
ϵ -greedy	150	150
Thompson Sampling	34	66

In Figure 4.1, we plot some key statistics about the arms versus index of rounds for intuitive understanding of the behaviour of the algorithms under the attacks. In Figure

Figure 2.2, we plot the number of times the optimal arm is pulled is chosen till round t , i.e. $n_{i^*}^t$ with the iteration t on the x axis in both the settings. We compare the two case with and without the attack. In both Figure 4.1 and Figure 2.2, the top row zooms in on the iterations in phase 1 and 2, i.e. the corrupted rounds whereas the bottom row shows the behaviour till the horizon T .

2.7.1 UCB1 Algorithm

In UCB1 algorithm, the most important statistics for the algorithm to make decisions is the upper confidence bounds on the arms' mean reward. UCB1 algorithm will always select the arm with the highest upper confidence bound. In sub-figures (a1) and (a2) in Figure 4.1, we plot the UCB values for both the target arm and optimal arm. Sub-figure (a1) shows that in the first phase, i.e. $t \leq C_1$, the UCB value for both the arms decreases to a value close to 0. In the next phase as the algorithm starts receiving high rewards from selecting the target arm, the UCB value for the target arm grows while that for the optimal arm stays close to 0. In the third phase when there is no corruption, sub-figure (a2) shows that till the end of the horizon, UCB value of the target arms is always greater than that of the optimal arm. As only true rewards generated by the environments are received, the UCB value of the target arm approaches to its mean reward, it never fall below the that of the optimal arm, so the optimal arm will never be selected in this phase. In sub-figures (a1) and (a2) of Figure 2.2, we plot the the number of cumulative times the optimal arm gets pulled by the round t . In sub-figure (a1) of Figure 2.2, we can see that in the second phase, as we start injecting higher rewards in the target arm, the algorithm completely stops choosing the optimal arm. After the second phase also, we can see in sub-figure (a2) of Figure 2.2 that the optimal arm never almost never gets pulled. In the absence of corruptions, UCB algorithm performs very well and the optimal arm is pulled almost always.

2.7.2 ϵ -greedy Algorithm

In ϵ -greedy Algorithm, the key parameters of arms for the algorithm to make decisions are their empirical mean. In the experiments where only two arms are considered, the arm with higher empirical mean will be picked with probability $1 - \epsilon/2$. In sub-figures (b1) and (b2) of Figure 4.1, we plot the empirical mean for both the target arm and optimal arm. Similar to UCB we see that in Phase 1, the empirical means concentrate around 0, then the empirical mean for target arm increases in phase 2, and in the last phase, the empirical mean of the target arm is always greater than that of the optimal arm till the end of horizon. Similar behaviour is seen in the number of times the optimal arm gets pulled. We see in sub-figures (b1) and (b2) of Figure 2.2 that under the attack, after Phase 1, the optimal arm gets picked very infrequently (only in explore rounds) whereas in the absence of corruptions, the optimal arm is picked almost always.

2.7.3 Thompson sampling Algorithm

In Thompson Sampling algorithm, the algorithm maintains a Beta distribution for each arm. Based on the Beta distribution for the two arms, in sub-figures (c1) and (c2) of Figure 4.1, we plot the approximate probability that a sample from the empirical Beta distribution associated with the optimal arm is greater than a sample from the empirical Beta distribution of the target arm. Again, similar to what happened in the UCB1 case, we can see that in sub-figure (c1) of Figure 4.1 that after Phase 1, the probability that the optimal arm is chosen drops close to zero. In sub-figure (c2) of Figure 4.1, we observe that the optimal arm can never recover from the corruption and the probability that it gets selected remains close to 0. This is reflected in sub-figures (c1) and (c2) of Figure 2.2 where we can see that under attack, after phase 1, the optimal arm never gets picked whereas in the absence of corruptions, the optimal arm is picked almost always.

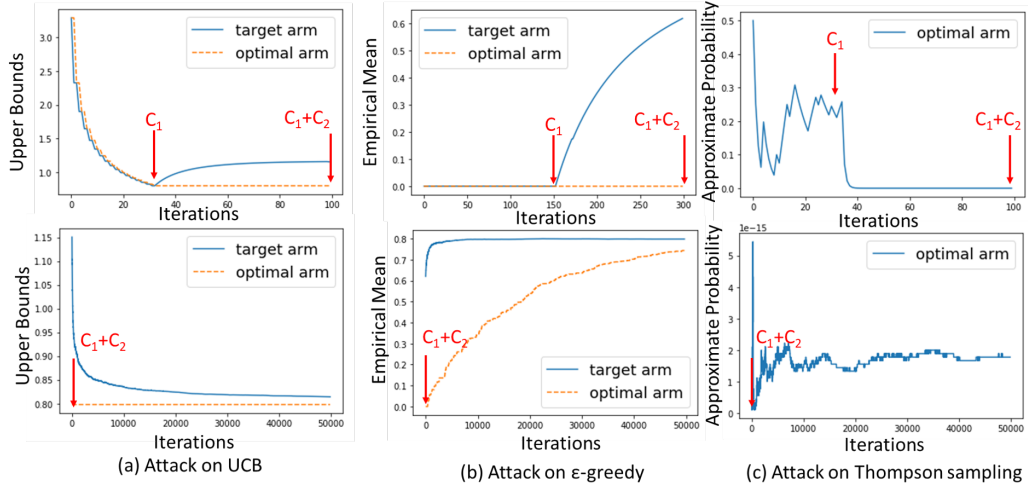


Figure 2.1: Empirical behaviors of arms in different algorithms. (a), (b) is for UCB algorithm; (c), (d) is for ϵ -greedy algorithm; (e), (f) is for Thompson sampling algorithm. (a), (c), (e) focus on the time when the corruption is active. (b), (d), (f) focus on the time when the corruption stops.

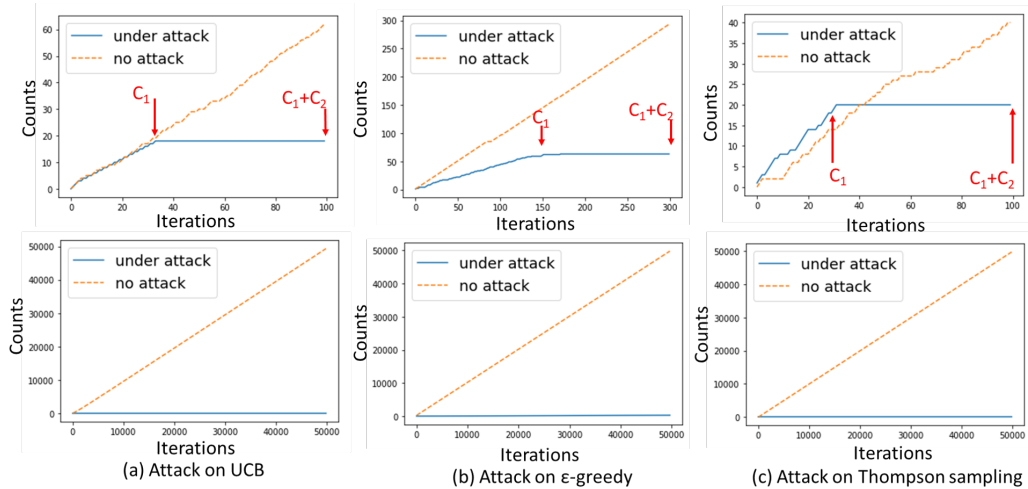


Figure 2.2: The number of rounds the optimal arm get picked $n_{i^*}^t$ before the corruption ends. (a) is for UCB algorithm, (b) is for ϵ -greedy algorithm, and (c) is for Thompson sampling algorithm.

2.8 Observation free attack oblivious to mean rewards

Here we introduce a slight modification on the original attack such that the new attack can be agnostic to the mean rewards while maintaining similar performance.

The modified observation free attack is still separated into three phases and applies corruption in the same way as before, but the length of each phase is determined during the first phase. Note that C_1 and C_2 only depends on the mean reward of the target arm, so to determine the length of each phase, the algorithm only needs to estimate the mean reward of the target arm by itself, and make the decision when the estimate is relatively accurate.

First set C_1 to be T temporarily, meaning that the attack will keep setting rewards for all arms as 0 until it makes the decisions on C_1 and C_2 formally. For the ease of notation, let n^t denote the number of rounds the target arm get selected by round t . By Hoeffding inequality, the adversary can have a lower confidence bound on the mean reward of the target arm as $\mu_{LCB} = \bar{\mu} - \sqrt{\frac{\log T}{n^t}}$. When $\mu_{LCB} = 2\sqrt{\frac{\log T}{n^t}}$, set C_1 and C_2 based on $\mu = \mu_{LCB}$. If the time τ to set C_1 is already greater than C_1 , then let $C_1 = \tau$ and determine new C_2 based on the new C_1 and μ_{LCB} correspondingly.

Lemma 5. *When attacking UCB1, ϵ -greedy, and Thompson sampling algorithms, with probability at least $1 - 2/T$, the new attack can have an estimation on the mean reward of the target arm before $\tau = \frac{16K \log T}{\mu^2} + K^2 \log T$, and the estimation satisfies $\hat{\mu} \in [\mu/2, \mu]$. Then the algorithm can set C_1 and C_2 based on $\hat{\mu}$.*

With probability at least $1 - 2/T$, the true mean reward of the target arm satisfies $\mu \in [\bar{\mu} - \sqrt{\frac{\log T}{n^t}}, \bar{\mu} + \sqrt{\frac{\log T}{n^t}}]$. If this is true, then when the adversary determines C_1 and C_2 , $\sqrt{\frac{\log T}{n^t}} \geq \mu/2$, which is equivalent to $n^t \leq \frac{4 \log T}{\mu^2}$. Note that in all the three algorithms mentioned above, n^t is at least $t/K - \sqrt{t \log T}$ with probability at least $1 - 1/T$. So the time the adversary determine C_1 and C_2 is at most $\frac{16K \log T}{\mu^2} + K^2 \log T$, and $\mu_{LCB} \geq \mu/2$.

When attacking algorithms like UCB, ϵ -greedy, and Thompson sampling algorithms, Theorem 5 implies that the adversary is able to have a good estimate on the mean reward of

the target arm during the first phase at an early time, based on which C_1 and C_2 could be determined. Then the attack could be applied in the same way given by observation free attack. Since the estimation on μ is at least $\mu/2$, C_1 and C_2 given by the estimation should be of the same order of that given by the real mean reward μ . If $\tau > C_1$ and C_1 is set as τ instead, then the new C_1 and C_2 are still sublinear.

2.9 Adversary that is not oblivious to algorithm's decision

At last we briefly explain why requiring adversary oblivious to algorithm's decision is important. If the adversary can observe the decision of the algorithm before it determines attack, then it only needs to apply corruption when the optimal arm is picked. Considering the fact that an algorithm has to pick non optimal arms for limited times to ensure low regret, so to make the algorithm always believe that the optimal arm is not the optimal, the adversary only needs to corrupt a small number of rounds. Further more, for any algorithm that has $o(T)$ guarantee regret in the stochastic, there is always an adversary with $C = o(T)$ corruption level to induce $\Omega(T)$ regret on the algorithm. These results has been shown in . Note that in the case where the bandit algorithm is deterministic, it doesn't matter whether the adversary observe the decision or not because in either case, the adversary may know what arm the algorithm will pick for sure. This also implies that all deterministic bandit algorithms are vulnerable.

CHAPTER 3

BANDIT ALGORITHMS ROBUST TO BOTH ADVERSARIES

In this chapter we will introduce algorithms that is robust against both types of adversaries. The motivation is that in PPC auctions, both adversaries actually exist in the form of bidders and click fraud. To make the setting to study even more practical, in addition to the adversaries we have mentioned, we introduce another element that can make bandit learning model much more suitable for the applications. The element is context [46], which can be think of as a side information that is relevant to the reward from each arm and can be utilized to get more rewards. As an example, consider online recommendation problem again where Yelp is recommending restaurants to users. This time, Yelp can observe users' information before making a recommendation. For example, now Yelp knows that this user is an Asian, then this user may favor Asian food more than other foods, so it may be better to recommend the best Asian restaurant to the user rather than a general best restaurant. In this chapter we will discuss PPC auction in the contextual bandit setting and present an algorithm (or mechanism) that is robust against incentives and can efficiently utilize the context. Moreover, the algorithm is also robust to data corruption adversary intrinsically. Next we formally introduce the contextual PPC auction setting and show that contextual bandit setting and stochastic PPC auction setting can be considered as two special cases.

3.1 Contextual PPC Auction Setting

In the contextual PPC auction setting, the action set \mathcal{A} is the set of advertisers (also referred to as *agents* or *arms*) who repeatedly compete for a single slot over T time steps (which could correspond to T users). If an advertiser $a \in \mathcal{A}$ is selected, then a click occurs with probability equal to the click-through-rate. Upon a click, the agent earns value equal to $\mu(a) \in [0, 1]$ (advertisers earn a fixed reward whenever a click happens, but different users

may have different probability of clicking their ad which depends on the context of the user). Contexts come from a context space \mathcal{X} and are identically and independently distributed (i.i.d.) across rounds. The click-through-rate function $\rho : \mathcal{A} \times \mathcal{X} \rightarrow [0, 1]$ maps agents and contexts to the probability of being clicked upon display.

In each round $t = 1, \dots, T$, a context $x^t \in \mathcal{X}$ is drawn i.i.d. from a fixed distribution. Each agent $a \in \mathcal{A}$ submits a bid $b^t(a)$, and in response the platform selects an agent a^t and displays her ad. The ad is clicked with probability $\rho(a^t, x^t)$. The platform receives the click result c^t and if the ad gets clicked, then it charges the agent a^t a payment p^t and the agent a^t gains utility $u^t(a^t) = \mu(a^t) - p^t$.

Following classical works on contextual bandits, we also assume a class of hypotheses (also referred to as experts) which we denote as \mathcal{H} with $|\mathcal{H}| = m$. Each expert $h : \mathcal{X} \rightarrow \mathcal{A}$ is a mapping from the context space to agents which the auctioneer can use to make decisions.

Just like the case without context, the goal of the platform is to maximize the cumulative social welfare defined as $\sum_t c^t \cdot \mu(a^t)$. Note that c^t is a random variable that depends on (possibly randomized selection of) agent a^t and on the randomness in the context x^t . However, with additional information from the context, it is possible that an algorithm could do better than picking the best arm based on the experts available. So the benchmark to evaluate how well the platform achieves this goal is different and depends on the experts. The typical benchmark is to compare against the expected social welfare of the best expert $h \in \mathcal{H}$ where the expectation is taken over the randomness in the contexts. When the value profile of the agents is $\vec{\mu} \in [0, 1]^K$ where $\mu(a)$ is the value of agent a , the expected welfare for an expert h is $R(h, \vec{\mu}) = \mathbb{E}_x[\mu(h(x)) \cdot \rho(h(x), x)]$. (Note that here we use $R(\cdot, \cdot)$ is the function to represent the expected reward for experts rather than regret as we used before. We will use REG to represent regret instead in this chapter.)

Let $h^* = \operatorname{argmax}_{h \in \mathcal{H}} \{R(h, \vec{\mu})\}$ be the expert with the highest expected welfare. The

performance of our algorithm against h^* is measured by the notion of regret:

$$\text{REG}(T) = T \cdot R(h^*, \vec{\mu}) - \sum_{t=1}^T c^t \cdot \mu(a^t).$$

Here we also point out that the contextual bandit problem and stochastic PPC auction problem are two special cases in this contextual PPC auction setting. If all the advertisers have the same value and no incentives, i.e., always bid their true value, then the problem is the same as the classical contextual bandit problem. If the experts set only consists of the experts which always recommend one arm regardless of the context, then the problem is the same as the stochastic PPC auction setting.

Now we have formally presented the contextual PPC auction setting without including the adversary that can apply data corruption. This is to let us focus on the incentives adversary with context problem for now, and later we will show that the algorithm (or mechanism in the case where arms have incentives) we suggest for this setting is intrinsically robust to the data corruption adversary. Next, we will introduce some additional notions for analysing the problem.

3.2 More Notions for Contextual PPC Auctions

[35] and [41] introduced the study of incentives in the multi-armed bandit setting by considering that agents bid strategically based aiming to maximize their utility defined as the difference between the total welfare they collect and the price they pay. To deal with such strategic agents, they restrict the allocation/payment design to mechanism that are *truthful*, i.e., agents cannot benefit by bidding something other than their value. They show that this truthfulness requirement poses additional challenges by providing $\Omega(T^{2/3})$ lower bound for this setting compared to the \sqrt{T} regret that is achievable in the absence of incentives. This $O(T^{2/3})$ regret can be achieved by classical algorithms that separate exploration from exploitation such as explore-then-commit and ϵ -greedy algorithms.

In this work, we introduce the study of truthful mechanism design in contextual bandit mechanisms. Our definition of truthfulness generalizes the one in multi-armed bandits:

Definition 3 (Truthful Mechanism). *A mechanism is truthful if for any sets of click-through-rates $\rho(\cdot, \cdot)$ and values $\vec{\mu}$, for every agent a , irrespective of the bidding strategies used by other other agent, a obtains the maximum expected utility over all T rounds by bidding her true value $b_a^t = \mu_a$ at every round t . In other words, if the mechanism is truthful, misreporting does not help any agent.*

For further analysis about truthfulness, more notions need to be introduced. First, we want to characterize how likely an agent will get a click if the mechanism follows an expert, which leads to the following definition:

Definition 4 (Click probability for an agent induced by an expert). *The click probability for an arm a induced by an expert h is*

$$C(a, h) := \sum_x p(x) \rho(a, x) \mathbb{1}\{h(x) = a\},$$

where $p(x)$ is the probability that the context is x .

Note that based on the definition of $C(a, h)$, the expected welfare of an expert can be written as $R(h, \boldsymbol{\mu}) = \sum_{a \in \mathcal{A}} \mu(a) \cdot C(a, h)$. Since mechanisms don't have access to agents' value, it is reasonable for mechanisms to treat bids as reported value, which leads to the definition of reported welfare of experts as follows.

Definition 5 (Reported welfare of an expert). *The reported welfare of an expert h when the bids profile is \mathbf{b} is:*

$$R(h, \mathbf{b}) = \sum_{a \in \mathcal{A}} b(a) \cdot C(a, h)$$

Note that when the mechanism is truthful, bids are expected to be the same as the values,

then the reported welfare of an expert will be identical to its expected welfare. In this case, the expert with the highest reported welfare is exactly the best expert $\operatorname{argmax}_{h \in \mathcal{H}} R(h, \mathbf{b}) = h^*$.

Next we want to characterize the incentives of agents. One important quantity which the incentives are based on is the relation between how much they bid and how likely they will get a click is important. The formal definition of such relation is as follows.

Definition 6 (Click probability of an agent). *Let A be the allocation rule used by the mechanism, H^t be the history till round t , and \mathcal{H} be the experts class. For an agent a , fixing the bids of the other agents as \mathbf{b}_{-a}^t , then the probability that the agent a get clicked in round t by bidding $\mathbf{b}(a) = b$ is given by*

$$g^t(a, b) = \mathbb{E}_x[\Pr\{A(\mathcal{H}, H^t, x, b, \mathbf{b}_{-a}^t) = a\}\rho(a, x)]$$

where $\rho(a, x)$ is the click through rate for agent a on context x .

Based on the definition of $g^t(a, b)$, we are able to formally write the expected utilization function of an agent a when her bid is b as:

$$u^t(a, b) = (\mu(a) - p^t(a, b))g^t(a, b),$$

where $p^t(a, b)$ to be is payment at round t if agent a is selected and clicked and her bid is b . Note that the function $p^t(\cdot, \cdot)$ can depend on the allocation rule, bids from other arms, data of history, and experts class.

Finally, we introduce the definition of exploration separated allocation rule. has shown the importance of such allocation rule for truthful mechanisms in stochastic setting.

Definition 7 (Exploration-separated allocation rule [35]). *An allocation rule is called **exploration-separated** if the allocation on any influential rounds does not depend on any bids, where influential rounds are the rounds whose click result and bids will influence the allocation in the future rounds.*

Our analysis on truthful mechanisms in contextual setting will focus on the mechanisms with such exploration separated allocation rules.

3.3 Truthful Mechanisms in Contextual PPC Auction

If contexts are indeed present and each experts predictions might change with the context, the requirement that agents are not allowed to observe the context becomes crucial to maintain truthfulness. In this section we will show that with such requirement, the ϵ -greedy mechanism can effectively utilize the contexts through experts while maintaining approximate truthfulness. Later we will show that when agents can see the context, without any assumptions on the expert class, it becomes harder to achieve truthfulness.

In the stochastic setting without contexts, since each expert is an arm and we are competing with the best arm in expectation, as shown in [35], a simple VCG style mechanism recovers the lower bounds for the regret. In the presence of a more complex class of experts which can utilize the context to make better decisions, the benchmark becomes harder as we are now competing with the best expert in expectation. To compete with the best expert, the platform has to consider expert predictions while making decisions and the interaction between the agents and the experts makes truthfulness harder. In this section, we require that the context is only available to the experts and cannot be observed by the agents. We show in section why such an assumption is necessary.

From now on we mostly focus on the mechanisms using exploration separated allocation rules. Under such allocation rules, agents only need to bid to maximize their utility for current round. In other words, for each a , if $u^t(a, b)$ is maximized at $b = \mu(a)$ for all $t \in [T]$, then all agent will always bid truthfully in every round. This fact gives a necessary and sufficient condition for a mechanism which uses an exploration separated allocation rule to be truthful.

Lemma 6. *If a mechanism uses an exploration separated allocation rule, then the mechanism is truthful if and only if for every round t and every agent a , the probability that she*

gets clicked in round t , i.e. $g^t(a, b)$ is monotonically increasing in her bid b and the payment rule satisfies

$$p^t(a, b) \cdot g^t(a, b) = b \cdot g^t(a, b) - \int_{y=0}^b g^t(a, y) dy$$

In fact, this payment rule can be viewed as the Pay-Per-Click auction version of the Myerson payment identity [47]. This theorem shows that for any mechanism using an exploration separated allocation rule, it can only be truthful for some payment rule if and only if $g^t(a, b)$ is always monotonically increasing in b .

Note that to have monotone $g^t(a, b)$, one simple allocation rule could be randomly picking an arm at each round, but clearly such allocation rule is far worse than the benchmark we want to achieve. We need more complicated allocation rule which can guarantee monotonicity of $g^t(a, b)$ while recovering the benchmark.

Perfect information. If assuming no learning is needed and $C(a, h)$ for all $a \in \mathcal{A}$ and $h \in \mathcal{H}$ is available at the beginning, to recover the benchmark, a straight forward idea is to always follow the expert with the highest reported $h^t = \operatorname{argmax}_h \{R(h, \mathbf{b})\}$ at each round. Mechanisms with such allocation rule could recover the benchmark if all agents will bid truthfully, and the lemma below shows that there exists payment rules to make the mechanism truthful.

Lemma 7. *In contextual PPC auction, there exists payment rules such that the mechanism which always follow the expert with the highest reported welfare $h^t = \operatorname{argmax}_h R(h, \mathbf{b}^t)$ is truthful.*

The intuition behind the proof is to show that the click probability of each agent a as a function of her bid is monotonically increasing. This is true because if an agent increases her bid, and the expert selected by the mechanism changes, then the the click probability for that agent induced by the new expert can only be higher.

Aside from the requirement of a monotone $g^t(a, b)$, there is a second difficulty for a mechanism to be truthful in contextual setting. The difficulty is originated from calculating

the payment. For the payment rule given in Theorem 6, when $g^t(a, b)$ is as simple as a step function, i.e., $g^t(a, b) = 0$ when b is less than a threshold and $g^t(a, b) = C$ where $C > 0$ is some constant, the corresponding payment rule will be just like a second price (the payment is the second highest bid) rule which can be found without knowing the exact value of C . However, if $g^t(a, b)$ is complicated and has multiple steps, the payment rule will be more complicated and require the exact knowledge of $g^t(a, b)$. If the payment rule a mechanism actually uses is only slightly deviated from the truthful payment rule, in some instances the maximum of the utility function of an agent could be achieved somewhere far from its real value. Here is an example to show when this could happen.

Example: Suppose there are three contexts x_1, x_2, x_3 , and the probability of each to show up is $1/3$. There are three experts h_1, h_2, h_3 and three arms a_1, a_2, a_3 , and the contents of each expert is summarized in the table. Let ρ be the 3 by 3 matrix where $\rho_{i,j}$ denote the click through rate of arm a_j when the context is x_i . Set ρ as

$$\rho = \begin{bmatrix} 0.7 & 0.4 & 0.9 \\ 0.2 & 0.2 & 0.5 \\ 0.6 & 0.8 & 0.3 \end{bmatrix}.$$

Denote \mathbf{b} be the bid profiles of all arms, and let $b_1 = 0.1, b_2 = 0.2$. Then the click probability and corresponding truthful payment rule of arm a_1 is given in Figure 3.1 (a), (b). For a_1 with any value, the utility maximizing option is to bid its value, and in Figure 3.1 (c) we show that for a_1 with different value, the maximal of the utility is always achieved at the value. However, if the payment rule is slightly deviated from the truthful one, then a_1 with some value will bid far from its value to optimize its utility. Let the payment deviation be 0.001, in Figure 3.2 (a), (b), we show that utility of a_1 with value 0.301 and 0.599 is not maximized at the corresponding value. To maximize the utility, a_1 could bid 0 instead of 0.301 and 1 instead of 0.599.

Due to this reason, we relax the requirement of truthful by introducing the definition

of α -rational agents, who will bid truthfully as long as bidding truthful is almost the best strategy.

Definition 8 (α -rational). *In mechanisms where the bids of agents of a round will not affect the future rounds, an α -rational agent will bid truthfully in round t if the extra utility in that round she can get by misreporting her value is bounded by α . That is, an α -rational agent a will bid truthfully in round t if*

$$\max_b u^t(a, b) - u^t(a, \mu(a)) \leq \alpha.$$

Note that $\alpha \in [0, 1]$.

Since estimating the optimal bids for the agents requires them to know information about the competition and other estimates of the mechanism, it is reasonable to assume that if the extra utility they get from trying to estimate their optimal bids is very small, then they can just bid truthfully.

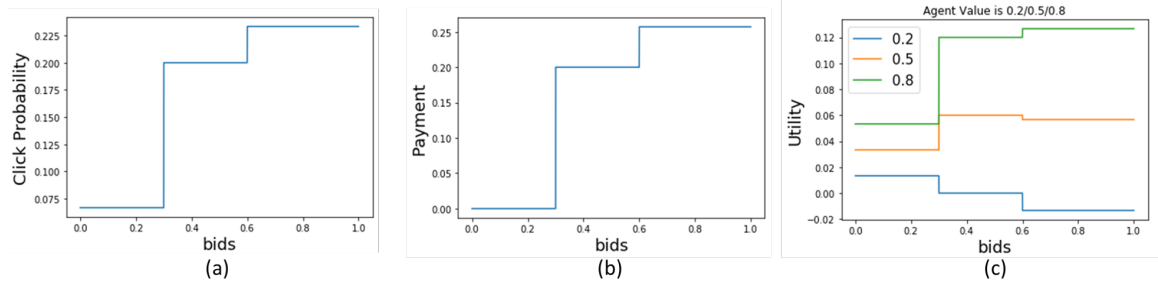


Figure 3.1: For arm a_1 , (a) is the relation between its bid and the probability that it receives a click, (b) is the payment for different bids if it is clicked, and (c) each line represent the utility it can get for different bid with an agent value.

3.4 ϵ -Greedy for Contextual Bandit Mechanisms

Now we present the ϵ -Greedy Mechanism algorithm 7 for the contextual multi-arm bandit PPC auction setting. Note that unlike the greedy algorithm we introduced before, the ϵ -Greedy mechanism here will not use information from exploit rounds to make decisions.

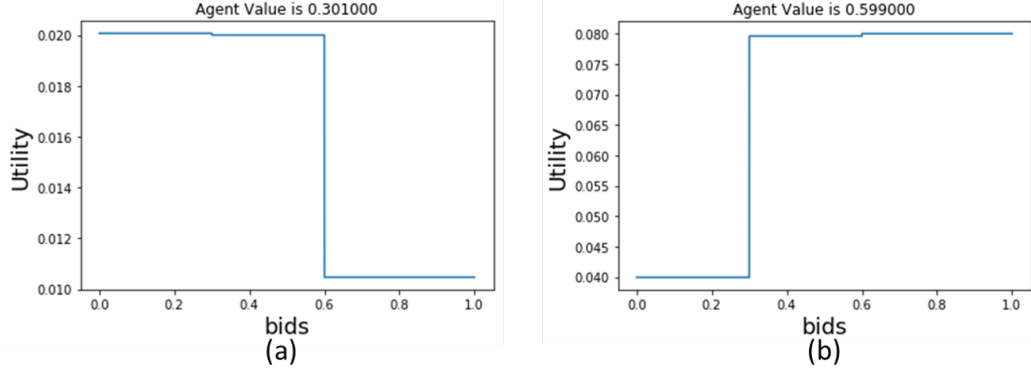


Figure 3.2: When the payment rule is not exactly the same as the truthful payment rule, the relation between the bid for agent with different agent value.

The mechanism works as follows. In each round, with a fixed probability ϵ , the mechanism uniformly at random select an arm. Such rounds are called *explore rounds* and the mechanism analyze the performance of each expert based on the results of Explore rounds only. The other rounds are called *exploit rounds* where the mechanism selects the empirically best expert under the current bids and then select the arm suggested by this expert. The mechanism also estimates the click probability of agents based on the result of the explore rounds, and use the payment rule given by Theorem 11 with the true click probabilities of agents substituted by the empirical estimation.

First we show that ϵ -greedy mechanism can maintain accurate empirical estimates of the click probability for agents induced by experts.

Lemma 8. *Let the number of experts in \mathcal{H} be m . With probability at least $1 - 3/T$, for any expert h , any arm a , at any round $t \geq 24 \log T / \epsilon$ the estimation error on click probability is bound by*

$$|\hat{C}^{t}(a, h) - C(a, h)| \leq K \sqrt{\frac{2 \log(T \cdot m \cdot K)}{t \epsilon}}.$$

proof sketch. Since in each round, we explore each arm with a fixed probability ϵ / K , thus for each expert we get a sample with this probability, and the result follows from a simple concentration bound. \square

Since we use empirical estimates to calculate the payments, in the initial few rounds,

Algorithm 7: ϵ -greedy Mechanism

Parameters : Number of arms K , Number of rounds T , exploration rate ϵ , A hypotheses class H

Initialize : $\hat{h} \leftarrow$ an arbitrary expert in H , $t_e \leftarrow \{\}$ /* \hat{h} is the empirical best expert, t_e is the index of explore rounds */

1 **for** $t = 1, \dots, T$ **do**

2 Receive bids b_j^t from arm j for all $j \in [K]$

3 $\ell^t \leftarrow \begin{cases} 1 & \text{w.p. } \epsilon \\ 0 & \text{otherwise} \end{cases}$ /* Explore or exploit */

4 **if** $\ell^t = 1$ **then**

5 /* ... **Explore Round** ... */

6 $t_e \leftarrow t_e \cup \{t\}$

7 $a^t \leftarrow j$ uniformly at random for $j \in [K]$ /* Select arm */

8 Receive click result c^t

9 Charge $p^t \leftarrow 0$ /* Payment */

10 **else**

11 /* ... **Exploit Round** ... */

12 **for** $h \in H$ **do**

13 **for** $i \in [K]$ **do**

14 $\hat{C}^t(i, h) \leftarrow \sum_{\tau \in t_e} \mathbb{1}(h(x) = i) \mathbb{1}(a^\tau = i) c^\tau \cdot K / |t_e|$

15 **end**

16 **end**

17 $h^* \leftarrow \operatorname{argmax}_h \sum_{i \in [K]} \hat{C}^t(i, h) b_i^t$ /* update the empirical optimal best expert */

18 h^* Receives context x^t /* only the experts have access to the context */

19 $a^t \leftarrow h^*(x^t)$ /* Select arm */

20 Receive click result c^t

21 **for** $y \in [0, 1]$ **do**

22 $b_{a^t}^t \leftarrow y$

23 $h \leftarrow \operatorname{argmax}_{h \in H} \sum_{i \in [K]} \hat{C}^t(i, h) b_i^t$

24 $\hat{g}^t(a^t, y) \leftarrow \hat{C}^t(a^t, h)$

25 **end**

26 Recover value of $b_{a^t}^t$

27 Charge price $p_i^t \leftarrow \begin{cases} b_i^t - \frac{\int_0^{b_i^t} \hat{g}^t(i, z) dz}{\hat{g}^t(i, b_i^t)} & \text{if } c^t = 1, a^t = i, \hat{g}^t(i, b_i^t) > 0 \\ 0 & \text{otherwise} \end{cases}$

 /* Payment */

28 **end**

29 **end**

agents might be able to take advantage of the loose estimates, but after a few rounds, we show that the the extra utility agents can get by misreporting is bounded. Thus if the agents are α -rational, after a few rounds, the agents will start bidding truthfully.

Theorem 9. *With probability at least $1 - 3/T$, an α -rational agent will bid truthfully in all $t \geq \frac{8K^2 \log(T \cdot m \cdot K)}{\epsilon \alpha^2}$ rounds under ϵ -Greedy Mechanism.*

The reasoning behind Theorem 15 show that if the estimation on $C(a, h)$ is accurate for all arms $a \in \mathcal{A}$ and experts $h \in \mathcal{H}$, then the allocation rule and the corresponding payment rule is more accurate, thus the extra utility from trying to exploit the errors in the estimate goes down as the rounds go by.

Finally, we show that ϵ -greedy mechanism can behave almost as well as the best expert when all agents are α -rational and present the main result in this setting.

Theorem 10. *If all agents are α -rational, the expected welfare regret of ϵ -Greedy Mechanism is bound by*

$$\text{REG}(T) \leq T\epsilon + K^2 \sqrt{\frac{8T \log(T \cdot m \cdot K)}{\epsilon}} + 3K + \frac{8K^2 \log(T \cdot m \cdot K)}{\epsilon \alpha^2}$$

Setting $\epsilon = 2T^{-1/3} \log(T \cdot m \cdot K)^{1/3} K^{4/3}$, we have

$$\text{REG}(T) \leq \tilde{O} \left(T^{2/3} K^{4/3} + T^{1/3} K^{2/3} / \alpha^2 \right)$$

The results come from the observation that regret can be decomposed into four parts: (a) explore rounds; (b) rare cases where estimation are not accurate at all; (c) the early few rounds when α rational agents do not bid truthfully; (d) exploit rounds when the estimates are accurate and agents are bidding truthfully. The first three parts of regret can be bound

straight forwardly. For the last part, we show that even if we follow the suggestions of a sub-optimal expert, its expected welfare can not be too far from that of the best expert.

Our results so far depended on the fact that the agents cannot see the realized context in each round before they submit their bids. In the following section, present a small discussion about this assumption and how things get much harder when agents can observe the contexts and strategize accordingly.

3.5 Without the condition that advertisers may observe context

When agents can observe the contexts, their behaviour is not with respect to the expectation of the context but can optimize for the specific realized contexts and can thus manipulate the mechanism who only has access to the contexts through the set of experts it has. More specifically, even if the mechanism knows the expected behaviour of experts, it cannot use the greedy best expert while maintaining incentive compatibility.

Theorem 11. *If a mechanism always follows the expert $h^t = \operatorname{argmax}_h R(h, \mathbf{b}^t)$, then there is no payment rule to make the mechanism truthful.*

When agents can observe the context, the function of click probability of an arm $g^t(a, b)$ is no longer an expectation over context but depends on context instead. So we can always find instances where $g^t(a, b)$ is not monotone on b for some specific contexts, and by Theorem 11, there is no payment rule to make such mechanism truthful. We give one such example in the appendix.

Although contexts create incentive issues in this setting, with some strong assumptions on the experts class \mathcal{H} , truthfulness can still be achieved. We give such class of experts where we can maintain truthfulness for the greedy mechanism even if the agents can observe the contexts.

Definition 9 (pre-ordered experts class). *An experts class \mathcal{H} is called pre-ordered if the following holds. For any two expert $h_1, h_2 \in \mathcal{H}$ and any agent $a \in \mathcal{A}$, at least one of the*

following must be true

(1): $\mathbb{1}\{h_1(x) = a\} \geq \mathbb{1}\{h_2(x) = a\}$ is true for all $x \in \mathcal{X}$

(2): $\mathbb{1}\{h_2(x) = a\} \geq \mathbb{1}\{h_1(x) = a\}$ is true for all $x \in \mathcal{X}$.

Examples: Although pre-ordering in the experts is a strong assumption, there are some reasonable expert classes which are pre-ordered. One example is when experts are agents where each expert always suggest an agent regardless of context, i.e. the stochastic bandit mechanism setting. Another example is when there are only two arms, the context space is $\{x : x \in [0, 1]\}$, and an expert h is associated with a threshold l such that it suggests arm 1 when $x \leq l$ and arm 2 when $x > l$. In more general cases, this assumption doesn't hold. For example, even if the number of arms in the second example becomes 3 and number of threshold becomes 2 for each expert, then without any assumptions on the threshold, this class of experts is not pre-ordered.

Theorem 12. *If the expert class is pre-ordered, then there exists a payment rule such that the mechanism which always follows the expert $h^t = \operatorname{argmax}_h R(h, \vec{b}^t)$ is truthful.*

A pre-ordered experts can guarantee that the probability of getting a click always increase when a bidder increases her bid, thus the *click allocation* for each agent can be monotonically increasing. Combining with the Theorem 11, we can obtain a payment rule that ensure truthfulness.

So far we have discussed the interplay between incentives and contexts in contextual multi arm bandit pay per click ad auctions. One more important aspect of online auctions is the potential presence of data corruptions. In the following section, we show that the ϵ -greedy is intrinsically robust to adversarial corruptions in the contextual multi arm bandit setting.

3.6 Robustness against data corruption

In this section, we show that, in addition, ϵ -greedy mechanism is intrinsically robust against data corruption.

3.6.1 Warm up: corruption robustness without contexts

As a warm up, we start with the stochastic setting where each expert is an arm itself. As discussed earlier, [35] show that in the absence of corruptions, an explore then commit approach can recover the $O(T^{2/3})$ lower bound of this setting. However when there is adversarial corruption, the simplest explore then commit mechanism is not robust against data corruption. Since the result from the explore phase determines the decisions in the exploit phase, if the adversary corrupt the whole initial explore phase of the mechanism, then the mechanism will make decisions totally based on the corrupted data where a sub-optimal arm may appears to be the optimal. The mechanism will then end up always selecting this arm and suffer from linear regret.

The ϵ -greedy mechanism is basically a randomized version of the explore then commit mechanism. Similar to the one introduced in the contextual setting, in each round, with a fixed probability ϵ , the mechanism uniformly at random selects an arm, and these rounds are called *explore rounds*. The mechanism learns the click through rate of each arm based on the results from these explore rounds only. The other others rounds are called *exploit rounds* where the mechanism select the empirically optimal arm given the current bids. The payment rule in the exploit rounds is a weighted second price rule, and the payment is always 0 in explore rounds. The formal form of the mechanism will be given in appendix.

ϵ -greedy mechanism is truthful given by [35], where a universal condition for a mechanism to be truthful in the stochastic setting is given, and ϵ -greedy is a specific mechanism that fits the condition. Basically, under ϵ -greedy mechanism, agents only need to bid to maximize the utility of current round, and a second price rule ensure that truthful bid can

achieve the highest utility from a round.

Unlike the deterministic explore then commit mechanism, the ϵ -greedy can always maintain an accurate estimation on arms' click through rate in the existence of data corruption, which provides an guarantee on the performance of ϵ -greedy mechanism in this setting

Theorem 13. *If the corruption level for the mechanism is C , let $C' = C + \max\{C, 6 \log(T)K/\epsilon\}$. The expected welfare regret of ϵ -greedy mechanism is bound by*

$$\text{REG}(T) \leq T\epsilon + \sqrt{\frac{8KT \log T}{\epsilon}} + \frac{24K \log T}{\epsilon} + 2(K + 1) + 4C' \log T$$

By taking $\epsilon = 2K^{1/3}(T \log T)^{-1/3}$, the welfare regret is bound by $\text{REG}(T) = \tilde{O}(K^{1/3}T^{2/3} + C)$.

The total regret can be decomposed into three parts: (a) regret from explore rounds; (b) regret in the rare case when the estimation is not accurate; (c) regret from exploit rounds when the estimation is accurate. The first two parts can be bound in straight forward ways. For the third part, we show that when the estimation is accurate, the expected welfare from the selected arm will not be far from that of the best arm.

We have shown that ϵ -greedy mechanism is simultaneously truthful and robust to the adversarial attack in the stochastic setting. Next we will show the ϵ -greedy mechanism is also robust against data corruption in the existence of context under the assumption that agents are α -rational.

3.6.2 Corruption in Contextual Setting

First we show that ϵ -greedy can accurately estimate the click probability of any expert on any arm.

Lemma 14. *Let the corruption level be C and denote $C' = C + \max\{C, \frac{6 \log T}{\epsilon}\}$. Let the number of experts in \mathcal{H} be m . With probability at least $1 - 3/T$, for any expert h , any arm*

a , at any round $t \geq 24 \log T / \epsilon$ the estimation error on click probability is bound by

$$|\hat{C}^t(a, h) - C(a, h)| \leq K \sqrt{\frac{2 \log(T \cdot m \cdot K)}{t \epsilon}} + \frac{2C'}{t}.$$

With an accurate estimation on click probability of experts, there would be guarantees for the behavior of α -rational agents and welfare regret when agents are α -rational which are similar to Theorem 15 and Theorem 10 with an extra dependence on the corruption.

Theorem 15. *With probability at least $1 - 3/T$, an α -rational agent will bid truthfully in all $t \geq \frac{8K^2 \log(T \cdot m \cdot K)}{\epsilon \alpha^2} + \frac{16C'}{\alpha}$ rounds under ϵ -Greedy Mechanism in context with experts setting.*

Theorem 16. *Let the corruption level be C and denote $C' = C + \max\{C, \frac{6 \log T}{\epsilon}\}$. Let the number of experts in \mathcal{H} be m . If all agents are α -rational, the expected welfare regret of ϵ -Greedy Mechanism is bound by*

$$\begin{aligned} \text{REG}(T) \leq & T\epsilon + K^2 \sqrt{\frac{8T \log(T \cdot m \cdot K)}{\epsilon}} + 3K + \\ & \frac{8K^2 \log(T \cdot m \cdot K)}{\epsilon \alpha^2} + \frac{16C'}{\alpha} + 4KC' \log T. \end{aligned}$$

By taking $\epsilon = 2T^{-1/3} \log(T \cdot m \cdot K)^{1/3} K^{4/3}$, we have $\text{REG}(T) \leq \tilde{O}(T^{2/3} K^{4/3} + T^{1/3} K^{2/3} / \alpha^2 + C' / \alpha)$

The proof is almost the same as that in the setting without corruption. The only difference is the error upper bound on estimation of click probability of experts.

CHAPTER 4

ALGORITHMS ROBUST AGAINST DATA CORRUPTION

In the last chapter we discuss the algorithms that only robust to data corruption without considering incentives. When arms have incentive, due to the fact that for truthful algorithms robust against incentives, the lower bound on regret is $O(T^{2/3})$, greedy algorithm becomes suitable for the corrupted contextual PPC auction setting. However, if there is only data corruption adversary and no incentives for each agent, greedy algorithms are not the best option. Some robust algorithms against data corruption have already been proposed. [34, 39] have designed algorithms that can guarantee $\tilde{O}(1/\Delta + C)$ upper bound on regret where \tilde{O} notation hides the logarithm factors. In practice, a modification on an existing algorithm will be more favored than a brand new algorithm, so it is interesting to ask if we could find a general way to modify existing bandit algorithms to be robust against data corruption. In this chapter, we present some exploratory work and some thinking about this topic.

4.1 Algorithm Decomposition: Estimator and Policy Maker

In this section we introduce a way to view bandit algorithms. On the high level, most bandit algorithm can be decomposed into two parts: an estimator and a policy maker. The estimator is to calculate the estimates on the statistics of arms, which usually includes mean and variance; the policy maker is to decide a probability distribution over arms based on the estimates returned by the estimator. The three main types of bandit algorithms, i.e., greedy, UCB, and Thompson sampling can be decomposed in this way. The estimator of all of them calculate mean and variance of an arm based on the data from the rounds where that arm is selected. The major difference between the three types of algorithms are their policy makers: greedy algorithms assign most probability into one arm and distribute the rest uniformly to other arms; UCB algorithms always put all probability on the potentially

best arm; Thompson sampling assign the probability in a more sophisticated way, yet usually most probability will concentrate on one arm.

In the stochastic setting, the estimators of the algorithms we just mentioned can form unbiased estimate on reward. However, when there exists data corruption, the estimate on mean could suffer from severe bias. For example, for UCB algorithms, the adversary could always set the reward of a specific arm to be 0 whenever it is picked, so the estimated mean of UCB algorithm on this arm is always 0 for all rounds while actually only $O(\log(T)/\Delta)$ rounds are corrupted. In fact, as we show through the observation free attack, and estimate formed by typical algorithms' estimators can be easily manipulated, which corresponds to high bias. Such high bias induce poor decisions made by reasonable policy makers. This fact motivates an idea to design robust algorithm, that is, we should use the estimators that can guarantee low bias on the estimates. In the next section we will introduce one kind of such estimator.

4.2 Estimators that minimize bias

The estimator we are going to introduce which aims at minimizing bias has already been adopted by EXP3 algorithms [9] and its variants [48], which are designed for the fully adversarial bandit setting. (The full adversarial setting can be considered as a special case of the corrupted stochastic setting with $C = T$, though the goal of the algorithm is different. Here we will not explain the fully adversarial setting in details and still focus the corrupted stochastic setting with $C = o(T)$.) Unlike the aforementioned estimators which estimate mean of an arm by averaging the rewards from the rounds when the arm is selected, this bias minimizing estimator estimate mean of an arm based on the data from all rounds and depends on the probability distribution given by the policy maker. Formally, the estimator estimate the mean reward of an arm a as follows.

Initialize $L = 0$. For round $t = 1, \dots, T$:

1. The policy maker decides that the probability this arm will be picked is p^t

2. Arm I^t is picked, and the reward is r^t . Define loss as $l^t = 1 - r^t$
3. Update $L = L + \frac{l^t}{p^t} \cdot \mathbb{1}\{I^t = a\}$
4. The mean of an arm is $\hat{\mu}^t(a) = 1 - L/t$

Let the true reward including corruption for arm a at round t as $r^t(a)$, then it is easy to verify that in expectation, $\mathbb{E}[\hat{\mu}^t(a)] = \sum_{\tau=1}^t r^\tau(a)/t$. Denote $C^t(a)$ as the accumulated corruption applied on arm a by round t , then the bias of the estimate on mean reward is $\frac{C^t(a)}{t}$. Considering that $C^t(a) \leq C = o(T)$, the bias is bound by $\frac{C}{t}$. The corresponding variance for the estimate is also easy to calculate $\sigma^t(a)^2 = \frac{\sum_{\tau=1}^t 1/p^\tau(a)}{t^2}$.

4.3 Problems and Solutions

Note that the error of the estimate on mean compared to the true value consists of both bias and variance. The variance can be directly evaluated through the formula $\sigma^t(a)^2 = \frac{\sum_{\tau=1}^t 1/p^\tau(a)}{t^2}$ while the bias is unknown to the estimator because the estimator doesn't have access to the corruption level C . In the last section we have shown that the bias is guaranteed to decrease with time, and here we first point out a problem induced by the variance.

4.3.1 High Variance Problem

The major problem for the estimator is that it is very sensitive to the probability distribution decided by the policy maker. As a simple example, if a $p^t(a)$ is set to be 0 for some arm a and round t , then the variance for that arm becomes unbounded. In the stochastic case, if an algorithm doesn't want to decrease the variance on an arm, then it just doesn't pick it for that round; in this case, to maintain the variance of an arm, the algorithm has to assign some probability on that arm. The price to maintain the variance is in fact not cheap as shown in the lemma below.

Lemma 17. *Assume at some round t , the variance on an arm is σ^2 . Then to keep the variance for the rest of the rounds, the algorithm will pick the arm for $(\log(T) - \log(t))/(2\sigma^2)$ rounds*

in expectation. Considering that $\sigma^2 \geq 1/t$, the algorithm will pick the arm for at least $t \cdot (\log(T) - \log(t))/2$ in expectation.

To prove Theorem 17, one only need to find the minimum p^t to keep the variance as σ^2 given that the variance before t is σ^2 , then sum the probabilities over t . In the stochastic case, an algorithm like UCB will stop picking an arm for a long while, and pick it from time to time when it is necessary to decrease its variance. In this case, the price to do the same thing is much more expensive as shown in the lemma below.

Lemma 18. *Assume at some round t , the variance on an arm is $\sigma^2 = 1/m$, and the algorithm wants to decrease the variance to $1/(m + 1)$ by selecting the arm for the next n rounds, then $n \approx \frac{t}{2m} = t \cdot \sigma^2/2$ if $m \ll t$.*

The proof for only needs to find the minimum number of n such that

$$\frac{t^2/m + n}{(t + n)^2} = \frac{1}{m + 1}$$

holds. The lemma generally claim that it is very hard to decrease the variance at a later round if the current variance is high. We verify this fact through an simple algorithm which is a combination of this estimator and UCB policy maker. The algorithm will focus most of the probability onto the potentially best arm, and assign the minimum probability to keep the variance for the other arms. Then we experimentally explore how will this algorithm run when there is no corruption. The case we consider consists of two arms with mean reward $\mu_1 = 0.8$, $\mu_2 = 0.6$, and top plot figure shows that probability that the sub-optimal arm will be picked versus iterations. As a comparison, the bottom plot shows the case where we test the same case for UCB1 algorithm. The top plot shows that in the middle of the game where the algorithm needs to decrease the upper bound of the sub-optimal arm, it has to select it for many rounds, which induces huge regret. In contrast, the bottom plot shows that in the stochastic case the algorithm usually only select the sub-optimal arm for once in the middle to reduce its variance.

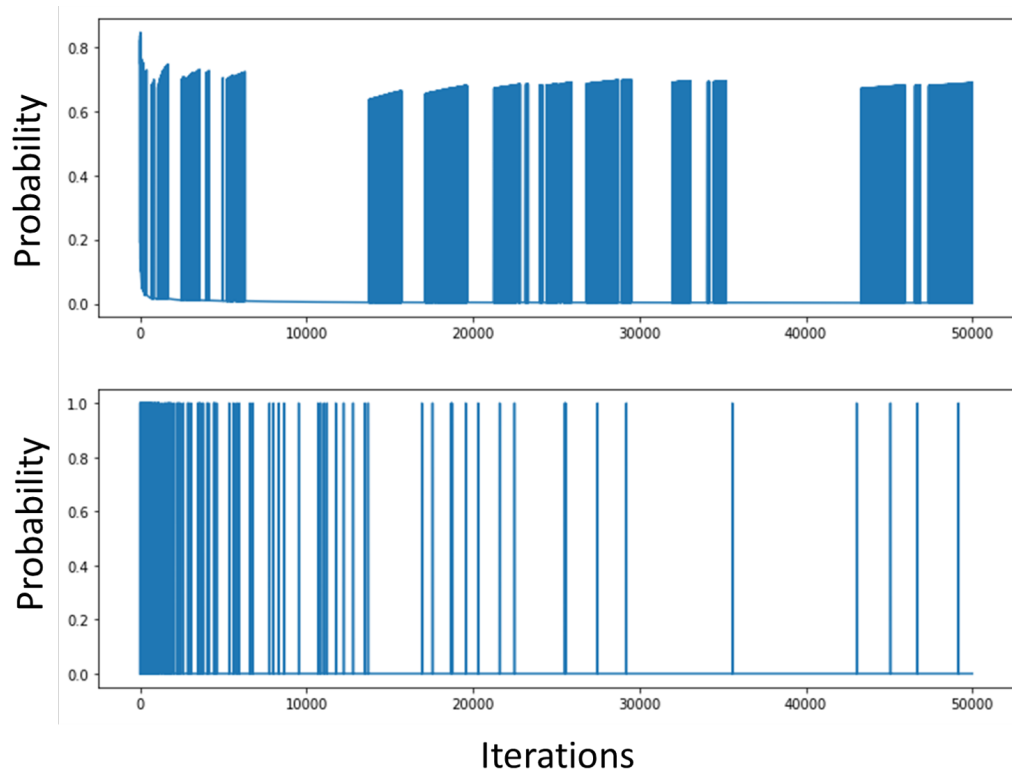


Figure 4.1: The probability the sub-optimal arm will be picked for each iteration. The top plot is for special UCB algorithm; the bottom plot is for UCB1 algorithm.

4.3.2 Solutions

A simple way to ensure the variance decreasing in a fast way is to fixing the probability to select an arm. It can be proved that if we are going to pick an arm for m rounds out of n rounds (in expectation), then the strategy that minimize the variance is to select that arm with fixed probability m/n .

Lemma 19. *If an arm is to be selected for m rounds from n rounds in expectation, then its variance is lower bound by $1/m$, and the corresponding strategy of the algorithm to achieve this is to select the arm with fixed probability m/n .*

The proof of Theorem 19 is to find the solution for a constrained optimization problem. The target function is $\min_x \sum_{i=1}^n 1/x^i$, and the constraint is $\sum_{i=1}^n x^i = m$. It is not hard to show that the solution is $x^i = m/n$ for all i . In fact, this lemma is closely related to lemma . lemma suggest that if one wants to decrease variance with a low price, then it should always keep the variance low, which is exactly what the algorithm does in lemma . ϵ -greedy algorithm (the robust one as we introduces in the last chapter) can be considered as a direct implementation of this way since it uses a fixed explore rate for each arm throughout the whole game.

Another way to control the variance is to trade bias for variance. When t is relatively large compared to C , the bias decrease in a slow rate while the price to decrease variance is high. So it is not economic to feed the estimator is a large number of rounds. One can just feed estimator with part of data to have a good estimation on the mean reward with limited price.

In fact, both ways can be found in the two existing algorithms robust to data corruption attack. Here we briefly introduce the core ideas of both algorithms.

The algorithm suggested by [34] consists of multiple pairs of estimators and policy makers. Each pair constructs an active eliminating algorithm with a modified confidence width which allow certain amount of corrupted data, and the probability that a pair is used

by the algorithm is fixed and distinct. For the pair with an appropriate probability to be used, the optimal arm will never be eliminated because the rounds that are corrupted are limited, so the modified active eliminating algorithm can work correctly with high probability. The main algorithm uses a tricky method to ensure that the right pair of estimator and policy maker will be used for most of time. So the algorithm actually explore each arm with an almost fixed rate, and some arms will not be explored at some point when the bias and variance are small enough to conclude that they are not the best.

The algorithm suggest by [39] roughly separated the whole game into several epochs. The length of an epoch is around twice the length of the last epoch. In each epoch, the probability distribution over arms is fixed and determined by the empirical mean reward calculate from the last epoch. Since the probability distribution is fixed in each epoch, the bias of estimate in each epoch is then limited. So each epoch starts with corrupted prior knowledge, based on which gathers information from every arm, and forward the estimates with bounded bias to the next epoch. By choosing appropriate probability distribution, the influence of the corruption on one epoch can only influence the regret from the next epoch. The regret guarantee from this algorithm is better than that last one in terms of logarithm factors.

The two ways we proposed may indeed be applied to constructing robust algorithms, however, both ways impose strong restrictions on the policy makers. It appears that such estimators could lead to efficient robust algorithms only for very limited policy makers. The question still remains how to implement the existing policy makers in a robust algorithm. A tentative idea is to finding another estimator that balance minimizing bias and controlling variance.

CHAPTER 5

CONCLUSION

In this work we introduce the basic ideas of bandit algorithms for the stochastic bandit settings, which have wide applications. We focus on the bandit settings which consider additional adversaries which greatly influence the performance of most typical bandit algorithms. We consider two types of adversaries: the first adversary is to consider the incentives from the agents; the second adversary is to consider data corruption on reward from a malicious third party. We show that both settings are important for its application in today's online business: online advertising auctions and recommendation systems. We first study how to efficiently manipulate the behavior of a bandit algorithm through data corruption adversary. We characterize most common bandit algorithms as mean based algorithms, and show that all of them are vulnerable to the data corruption adversary. We explicitly analyze the test how exactly the attack method we design manipulate some example bandit algorithms including UCB1, ϵ -greedy, and Thompson sampling with Beta distribution. We suggests that one reason why these algorithms are vulnerable is because their estimates are prone to suffer from high bias induced by data corruption. Next we introduce the contextual bandit setting which are more capable in applications, and study the contextual bandit setting with both types of adversaries. We analyze the setting and propose an algorithm that is simultaneously robust to both adversaries while efficiently utilizing the information from the context. At last we propose a question that how can we modify existing algorithms to be robust against data corruption in a general way. We propose a perspective to view bandit algorithms by decomposing them into estimators and policy makers. We analyze the application of an estimator which minimize the bias of estimates on bandit algorithms. We discover that this estimator could be applied in constructing robust algorithms, and indeed people discover robust algorithms that is in consistence of the features we propose,

though the estimator can work with some tight restrictions.

Appendices

Proof for Theorem 1

Proof. Denote the two arms in instances with two arms as a_1 and a_2 . Given an instance where the means of both arms are 0. For any constant C_1 , there is always at least one arm such that it gets at least $C_1/2$ picks with probability at least $1/2$, denote such arm as a_1 . We consider an instance (1) where a_1 is the optimal arm:

$$\mu_{a_2}^{(1)} = \mu,$$

$$\mu_{a_1}^{(1)} > \mu_{a_2}^{(1)}.$$

We will perform the observation free attack on instance (1). In the first phase of attack, the rewards are always 0 for any arm. By the end of the first case, for instance (1), from the way we set a_1 , we have with probability at least $1/2$, the following will happen:

$$n_{a_1}^{(1)} \geq \frac{C_1}{2}, \hat{\mu}_{a_1}^{(1)} = 0,$$

$$n_{a_2}^{(1)} \leq \frac{C_1}{2}, \hat{\mu}_{a_2}^{(1)} = 0.$$

Let G_1 be the event that the above is true in instance (1), we know that $\Pr\{G_1\} \geq 1/2$.

Next, consider another instance (2) where the mean reward of a_2 is 1, and the mean reward of other arm is 0:

$$\mu_{a_1}^{(2)} = 0,$$

$$\mu_{a_2}^{(2)} = 1.$$

For instance (2), we corrupt the first C_1 rounds and set the rewards to be 0 for all arms, then stop corruption. Let $N_1^{(2)}$ be the number of rounds when the algorithm pick arm 1 after the corruption ends. Let f_1 be the value such that $\Pr\{N_1^{(2)} \geq f_1\} = 1/2$. The expected regret of the algorithm is at least $R^{(2)}(T) \geq 1/2f_1$. So $f_1 \leq 2R^{(2)}(T)$, which has to be sublinear or otherwise the algorithm has linear expected regret in instance (2).

Next we focus on the second phase of attack in instance (1). Let $C_2 = f_1 + \alpha C_1$ where α is a parameter to be specified later. Up the end of this phase, what happened in (1) is the same as that in (2). So with probability $1/2$, a_1 is picked for less than f_1 rounds in this phase. Denote such Event as G_2 , then $\Pr\{G_2\} = 1/2$. If both G_1 and G_2 are true, by the end of the second phase of attack, the following is true :

$$n_{a_1}^{(1)} \geq \frac{C_1}{2}, \hat{\mu}_{a_1}^{(1)} = 0,$$

$$n_{a_2}^{(1)} \geq \alpha C_1, \hat{\mu}_{a_2}^{(1)} \geq \frac{2\alpha}{2\alpha + 1}.$$

Next we focus on the last phase of attack in instance (1) where the corruption is ended. For any value of n , if a_2 get picked for n times in this phase, then by Hoeffding inequality inequality, with probability at least $1 - 1/T$, the reward from these n rounds is at least $\mu n - \sqrt{\log(T)n}$ for any $n \leq T$. Set $\alpha = \frac{\log(T) + \mu}{\frac{2\mu C_1}{1-\mu/2} + 4}$, the corresponding empirical mean of a_2 satisfies

$$\bar{\mu} = \frac{C_1 \cdot \alpha + n\mu - \sqrt{n \log(T)}}{C_1(\alpha + 1/2) + n} \geq \mu/2.$$

That is, in the last phase, with probability at least $1 - 1/T$, the empirical mean of a_2 is always greater than $\mu/2$. Let G_3 denote the event where the above happens, so $\Pr\{G_3\} \geq 1 - 1/T$.

Before proceeding, we introduce an instance (3) where the reward of arm a_1 is always $\mu/4$ and the reward of a_2 is always $\mu/2$. Let n_1^t and n_2^t be the number of rounds a_1 and a_2 get selected by round t . Define random variables $\{Y_1, \dots, Y_{T/2}\}$ where Y_n is n_1^t if exists a t such that $n_2^t = n$, and $T - n$ if such t doesn't exists. It is clear that $\Pr\{Y_n < 0\} = 0$, $\Pr\{Y_n < T\} = 1$, and $\Pr\{Y_n < x\} \leq \Pr\{Y_n < x\} + 1$. So we could always find an integer k such that $\Pr\{Y_{T/2} < k\} = 1/2$, and such k must be sublinear in T or otherwise the regret in instance (3) will be linear. Y_n also satisfies $Y_n \in [Y_{n-1}, Y_{n-1} + 1, \dots, T - n]$, and $\Pr\{Y_n = Y_{n-1} + i | Y_{n-1} = y_{n-1}\} \geq \Pr\{Y_n = Y_{n-1} + j | Y_{n-1} = y_{n-1}\}$ for all $0 \leq i \leq j$ and y_{n-1} . The purpose of introducing instance (3) is to show that if the algorithm have sublinear regret in this instance, then with probability $1/2$, it won't pick a_1 for more than k

times. Then in stance (1), by choosing big enough C_1 and C_2 , with probability at least $1/2$, it won't pick a_1 for more than k times, so the algorithm will have linear regret in instance (1).

Now back to instance (1) and set $C_1 = (8/\mu - 2)k$, so at the beginning of the last phase, a_1 has already been picked for at least $(4/\mu - 1)k$ rounds. Then the empirical mean of a_1 will not exceed $\mu/4$ before it get at least k picks from this phase. Then by the definition of mean based algorithm, we know that before a_1 get its k^{th} pick, the probability a_1 get picked in instance (1) is always less than that in instance (3) for the same number of rounds a_2 get picked. Let n_1^t and n_2^t as the number of rounds arm a_1 and a_2 get picked in the last phase by round t . Define random variables $\{Z_1, \dots, Z_{T/2}\}$ in the same way as Y_n where $Z^n = n_1^t$ if exists t such that $n_2^t = n$ and $Z^n = T - n$ if such t doesn't exists. Z_n also satisfies $Z_n \in [Z_{n-1}, Z_{n-1} + 1, \dots, T - n]$, and $\Pr\{Z_n = Z_{n-1} + i | Z_{n-1} = z_{n-1}\} \geq \Pr\{Z_n = Z_{n-1} + j | Z_{n-1} = z_{n-1}\}$ for all $0 \leq i \leq j$ and z_{n-1} . The relation between Z_n and Y_n satisfies: $\Pr\{Z_n = x | Z_{n-1} = x\} \geq \Pr\{Y_n = x | Y_{n-1} = x\}$ for all $x \leq k$ and $\Pr\{Z_n = x + i | Z_{n-1} = x\} \leq \Pr\{Y_n = x + i | Y_{n-1} = x\}$ for all $i > 0$ and $x + i \leq k$. Intuitively, Z_n "grows" slower than Y_n before it exceeds k , so Y_n is more likely to reach k than Z_n . Next are we going to strictly prove that $\Pr\{Y_{T/2} \leq k\} \leq \Pr\{Z_{T/2} \leq k\}$.

Note that $\Pr\{Y_n \leq k\}$ depends on $\Pr\{Y_m | Y_{m-1}\}$ for all $m \leq n$. The idea of the proof is to show that by substituting each $\Pr\{Y_m | Y_{m-1}\}$ by $\Pr\{Z_m | Z_{m-1}\}$, the probability of $\Pr\{Y_n \leq k\}$ will increase. We introduce another series of random variables $\{F_1^1, \dots, F_{T/2}^1\}$ where $\{F_n^1\}$ is almost the same as $\{Y_n\}$ except that $\Pr\{F_m^1 | F_{m-1}^1\} = \Pr\{Z_m | Z_{m-1}\}$ for a specific m . We want to show that $\Pr\{Y_{T/2} \leq k\} \leq \Pr\{F_{T/2}^1 \leq k\}$. After that, we can construct $\{F_n^2\}$ which is almost the same as $\{F_n^1\}$ except for $\Pr\{F_{m'}^2 | F_{m'-1}^1\} = \Pr\{Z_{m'} | Z_{m'-1}\}$ where $m' \neq m$. For the same reason we will have $\Pr\{F_{T/2}^1 \leq k\} \leq \Pr\{F_{T/2}^2 \leq k\}$. Repeat this process until $\{F_n^{T/2}\}$ which is the same as $\{Z_n\}$, then we have $\Pr\{Y_{T/2} \leq k\} \leq \Pr\{F_{T/2}^1 \leq k\} \leq \Pr\{F_{T/2}^2 \leq k\} \leq \dots \leq \Pr\{F_{T/2}^{T/2} \leq k\} = \Pr\{Z_{T/2} \leq k\}$. Next we will prove that $\Pr\{Y_{T/2} \leq k\} \leq \Pr\{F_{T/2}^1 \leq k\}$.

First, we can write $\Pr\{Y_{T/2} \leq k\}$ as

$$\begin{aligned}
& \Pr\{Y_{T/2} \leq k\} \\
&= \sum_{x=0}^k \Pr\{Y_{T/2} \leq k | Y_{m-1} = x\} \cdot \Pr\{Y_{m-1} = x\} \\
&= \sum_{x=0}^k \Pr\{Y_{m-1} = x\} \cdot \sum_{y=x}^k \Pr\{Y_n \leq k | Y_m = y, Y_{m-1} = x\} \cdot \Pr\{Y_m = y | Y_{m-1} = x\} \\
&= \sum_{x=0}^k \Pr\{F_{m-1}^1 = x\} \cdot \sum_{y=x}^k \Pr\{F_n^1 \leq k | F_m^1 = y\} \cdot \Pr\{Y_m = y | Y_{m-1} = x\}
\end{aligned}$$

The difference between $\Pr\{Y_{T/2} \leq k\}$ and $\Pr\{F_{T/2}^1 \leq k\}$ can be written as

$$\begin{aligned}
& \Pr\{Y_{T/2} \leq k\} - \Pr\{F_{T/2}^1 \leq k\} \\
&= \sum_{x=0}^k \Pr\{F_{m-1}^1 = x\} \cdot \sum_{y=x}^k \Pr\{F_n^1 \leq k | F_m^1 = y\} \cdot (\Pr\{Y_m = y | Y_{m-1} = x\} - \Pr\{F_m^1 = y | F_{m-1}^1 = x\})
\end{aligned}$$

$$\begin{aligned}
& \sum_{y=x}^k \Pr\{F_n^1 \leq k | F_m^1 = y\} \cdot (\Pr\{Y_m = y | Y_{m-1} = x\} - \Pr\{F_m^1 = y | F_{m-1}^1 = x\}) \\
&= \Pr\{Y_n \leq k | Y_m = y\} \cdot (\Pr\{Y_m = x | Y_{m-1} = x\} - \Pr\{F_m^1 = x | F_{m-1}^1 = x\}) \\
&+ \sum_{y=x+1}^k \Pr\{F_n^1 \leq k | F_m^1 = y\} \cdot (\Pr\{Y_m = y | Y_{m-1} = x\} - \Pr\{F_m^1 = y | F_{m-1}^1 = x\}) \\
&= \Pr\{Y_n \leq k | Y_m = y\} \cdot \sum_{z=x+1}^{T-m} (\Pr\{F_m^1 = z | F_{m-1}^1 = x\} - \Pr\{Y_m = z | Y_{m-1} = x\}) \\
&+ \sum_{y=x+1}^k \Pr\{F_n^1 \leq k | F_m^1 = y\} \cdot (\Pr\{Y_m = y | Y_{m-1} = x\} - \Pr\{F_m^1 = y | F_{m-1}^1 = x\}) \\
&\leq \Pr\{Y_n \leq k | Y_m = y\} \cdot \sum_{z=x+1}^y (\Pr\{F_m^1 = z | F_{m-1}^1 = x\} - \Pr\{Y_m = z | Y_{m-1} = x\}) \\
&+ \sum_{y=x+1}^k \Pr\{F_n^1 \leq k | F_m^1 = y\} \cdot (\Pr\{Y_m = y | Y_{m-1} = x\} - \Pr\{F_m^1 = y | F_{m-1}^1 = x\}) \\
&= \sum_{y=x+1}^k (\Pr\{F_n^1 \leq k | F_m^1 = x\} - \Pr\{F_n^1 \leq k | F_m^1 = y\}) (\Pr\{F_m^1 = y | F_{m-1}^1 = x\} \\
&- \Pr\{Y_m = y | Y_{m-1} = x\})
\end{aligned}$$

We can directly have $\Pr\{F_m^1 = y | F_{m-1}^1 = x\} - \Pr\{Y_m = y | Y_{m-1} = x\} \leq 0$, for the other term, we have:

$$\begin{aligned}
& \Pr\{F_n^1 \leq k | F_m^1 = x\} \\
&= \Pr\{F_n^1 \leq k | F_{m+1}^1 \leq k, F_m^1 = x\} \cdot \Pr\{F_{m+1}^1 \leq k | F_m^1 = x\} \\
&= \Pr\{F_n^1 \leq k | F_{m+1}^1 \leq k\} \cdot \Pr\{F_{m+1}^1 \leq k | F_m^1 = x\} \\
&\geq \Pr\{F_n^1 \leq k | F_{m+1}^1 \leq k\} \cdot \Pr\{F_{m+1}^1 \leq k | F_m^1 = y\} \\
&= \Pr\{F_n^1 \leq k | F_m^1 = y\}
\end{aligned}$$

So eventually we have

$$\begin{aligned}
& \Pr\{Y_{T/2} \leq k\} - \Pr\{F_{T/2}^1 \leq k\} \\
&= \sum_{x=0}^k \Pr\{F_{m-1}^1 = x\} \cdot \sum_{y=x}^k \Pr\{F_n^1 \leq k | F_m^1 = y\} \cdot (\Pr\{Y_m = y | Y_{m-1} = x\} - \Pr\{F_m^1 = y | F_{m-1}^1 = x\}) \\
&\leq \sum_{x=0}^k \Pr\{F_{m-1}^1 = x\} \cdot \sum_{y=x+1}^k (\Pr\{F_n^1 \leq k | F_m^1 = x\} \\
&\quad - \Pr\{F_n^1 \leq k | F_m^1 = y\}) (\Pr\{F_m^1 = y | F_{m-1}^1 = x\} - \Pr\{Y_m = y | Y_{m-1} = x\}) \\
&\leq 0
\end{aligned}$$

As discussed before, by the same process we have $\Pr\{F_{T/2}^1 \leq k\} - \Pr\{F_{T/2}^2 \leq k\} \leq 0$ and so on. So $\Pr\{Y_{T/2} \leq k\} \leq \Pr\{F_{T/2}^1 \leq k\} \leq \Pr\{F_{T/2}^2 \leq k\} \leq \dots \leq \Pr\{F_{T/2}^{T/2} \leq k\} = \Pr\{Z_{T/2} \leq k\}$. Next we will prove that $\Pr\{Y_{T/2} \leq k\} \leq \Pr\{F_{T/2}^1 \leq k\}$. That is, with probability at least $1/2$, in instance (1), a_2 will be picked for more than $T/2$ rounds and by that time a_1 is picked for less than k rounds.

Suppose the algorithm guarantee sublinear regret in instances (2) and (3). Let $\mu = 1/2$ and the mean reward of the optimal arm as 1, set $C_1 = 14k$ and $C_2 = f_1 + \frac{3}{4} \log(T) + \frac{7}{3}k$, the expected regret for the algorithm in instance (1) is at least $T/16$. \square

Proof for Theorem 2

Proof. Let 1 as the index of the target arm. Under the adversarial attack, in the first phase of attack when $t \leq C_1$, the empirical mean of any arm will always be 0, so the empirical upper confidence for each arm j satisfies

$$\text{UCB}_j^t = \hat{\mu}_j^t + \sqrt{\frac{\log T}{n_j^t}} = \sqrt{\frac{\log T}{n_j^t}}.$$

It is clear that $\arg\max_j \text{UCB}_j^t = j$ n_j^t . So an arm could get its $n + 1^{\text{th}}$ pick only after all

other arms get selected at least n times. That is, arms will be selected in turn. Hence, when $t = C_1 + 1$, all arms will be selected for C_1/K times.

In the second phase of attack When $C_1 < t < C_1 + C_2$, the empirical mean of the target arm is increasing whenever it get selected while that of the others remain 0. If we choose $C_1 \geq \frac{4 \log T}{K}$, then the upper confidence bound of the target arm 1 when it gets n picks at this period satisfies:

$$\begin{aligned} \text{UCB}_1^t &= \hat{\mu}_1^t + \sqrt{\frac{\log T}{n_1^t}} \\ &= \frac{n}{n + 4 \log T / K^2} + \sqrt{\frac{\log T}{C_1/K + n}} \\ &\geq \sqrt{\frac{\log T}{C_1/K}} = \text{UCB}_{i \neq 1}^t \end{aligned}$$

So the target arm will get all the C_2 picks at this period. We choose $C_1 = \max\{\frac{K \log(T)}{\mu_1^2}, \frac{4 \log T}{K}\}$, so that the upper bound of other arms at the end of the second phase will be no greater than μ_1 . Then we choose $C_2 = \frac{\mu_1}{1-\mu_1} C_1$ so that at the end of the second phase, the empirical mean of the second arm is its true mean μ_1

In the last phase of attack when $t > C_1 + C_2$, we will show that the target arm will be picked for all rounds with a high probability. When the target arm get n picks in this phase, by Hoeffding inequality, with probability at least $1 - 1/T$, the total reward generated from these n rounds is greater than $\mu_1 n - \sqrt{n \log T}$ for any value of $n < T$. Denote the number of rounds the target arm get picked before $t = C_1 + C_2$ as m , then the upper bound of the target arm satisfies

$$\text{UCB}_j^t = \hat{\mu}_1^t + \sqrt{\frac{\log T}{n_1^t}} \geq \mu_1 - \frac{\sqrt{n \log T}}{n + m} + \sqrt{\frac{\log T}{n + m}} > \mu_1.$$

Therefore the target arm's upper confidence bound is always the highest no matter how many

times it get picks in the last phase, which means it will always get picked with probability at least $1 - 1/T$.

In conclusion, to defeat UCB1 algorithm, the observation free attack corrupt the first $\max\{\frac{K \log(T)}{\mu_1^2}, \frac{4 \log T}{K}\}/(1 - \mu_1)$ rounds, and the number of rounds arm other than the target get selected is less than $\frac{(K-1) \log(T)}{\mu_1^2}$ with probability at least $1 - \frac{1}{T}$. \square

Proof for Theorem 3

Proof. We refer to the rounds where the algorithm randomly pick an arm from all arms as “explore” rounds. Under the corruption from adversary algorithm, in the first phase of attack when $t < C_1$, all arms have the same probability to get picked because their empirical means are all 0. So each arm will get picked no less than

$$n_1 = C_1/K - \sqrt{C_1 \log T}$$

rounds and no more than

$$n_2 = C_1/K + \sqrt{C_1 \log T}$$

with probability at least $1 - \frac{K}{T}$ given by Hoeffding inequality. Next we will discuss the case where the above is true.

In the second phase of attack when $C_1 < t \leq C_1 + C_2$, once the target arm get one pick, its empirical mean will be the highest, and it will be selected with probability at least $1 - \epsilon$. With probability at least $1 - 1/T$, the target arm will get its first pick after $K \log(T)$ rounds. After that, with probability at least $1 - 1/T$, the target arm will get picked for at least

$$n(C_2) = (C_2 - K \log(T))(1 - \epsilon) - \sqrt{C_2 \log(T)}$$

times. Denote μ as the empirical mean of the target arm, to simplify the analysis, we choose C_2 big enough such that the target arm can get picked at least $n_3 = \max\{\frac{\log T}{\mu^2}, n_2 \frac{\mu}{1-\mu}\}$ times during this period. The reason we choose this n_3 is to give some guarantee on the

empirical mean of target arm when $t > C_1 + C_2$. To make sure $n(C_2) \geq n_3$, we can choose

$$C_2 = K \log T + \frac{\log(T) + 4n_3(1 - \epsilon)}{(1 - \epsilon)^2}$$

In the last phase of attack when $t > C_1 + C_2$, we want to find a lower bound on empirical mean of the target arm. Note that $n_3 \geq \frac{4 \log T}{\mu^2}$, so that empirical mean of the target arm at the beginning of this phase $t = C_1 + C_2 + 1$ is greater than μ . Denote the number of rounds the target arm get picked after $t = C_1 + C_2$ as m , the empirical mean of the target arm satisfies:

$$\begin{aligned} \hat{\mu} &\geq \frac{\mu n_3 + \mu m - \sqrt{m \log T}}{n_3 + m} \\ &= \mu - \frac{\sqrt{m \log T}}{n_3 + m} \\ &\geq \mu - 0.5 \sqrt{\frac{\log T}{n_3}} \\ &= 0.5\mu \end{aligned}$$

Therefore, before an arm other than the target arm has its empirical mean greater than 0.5μ , the probability it get picked is ϵ/K . We want C_1 to be big enough such that the empirical means of other arm are always less than $\mu/2$ in the last phase. From Hoeffding inequality, with probability at least $1 - 1/T$, an arm will get picked from explore rounds for at most $T \log T \epsilon / K$ rounds. If the arm never get picked from the exploit rounds, its empirical mean satisfies:

$$\hat{\mu}_i \leq \frac{T \log T \epsilon / K}{T \log T \epsilon / K + n_1}.$$

Set

$$C_1 = \log T K (\log T + \frac{4}{K}) T \epsilon (1 + \frac{2}{\mu}),$$

such that

$$n_1 = (T \log T \epsilon / K) \left(1 + \frac{2}{\mu}\right),$$

then we have $\hat{\mu}_i \leq \mu/2$. So with this C_1 , with probability at least $1 - K/T$, the empirical mean of other arms never exceed that of the target arm hence get not picks from the explore rounds. Based on such C_1 , the corresponding C_2 is

$$C_2 = \left(K + \frac{1}{(1 - \epsilon)^2}\right) \log T + \frac{4}{1 - \epsilon} \max\left\{\frac{\log T}{\mu^2}, \frac{\mu}{1 - \mu} (C_1/K + \sqrt{C_1 \log T})\right\}.$$

With such C_1 and C_2 , the ϵ -greedy algorithm will pick arms other than the target arm by at most $C_1 + T\epsilon + \sqrt{C_2 \log T}$ times with probability at least $1 - (2K + 2)/T$.

□

Proof for Theorem 4

Proof. Let 1 be the index of the target arm. When $t < C_1$, we want to show that all arms will get picked for around C_1/K rounds. Let's start with the case where $K = 2$. Denote Δ^t as the difference of number of rounds the other get picked, and $\Delta^{t+1} - \Delta^t$ as δ^t . The probability that the arm which get more picked before get picked this round is no greater than 1/2. That is, if $\Delta^t \geq 0$, $\Pr\{\delta^t = 1\} \leq 1/2$ and $\Pr\{\delta^t = -1\} \geq 1/2$; if $\Delta^t \leq 0$, $\Pr\{\delta^t = 1\} \geq 1/2$ and $\Pr\{\delta^t = -1\} \leq 1/2$. Since $\Delta^{t=C_1+1} = \sum_{t=1}^{C_1} \delta^t$, with probability at least $1 - 1/T$, $\Delta^{t=C_1+1} \leq \sqrt{C_1 \log T}$. In the case where $K > 2$, we can define $\Delta_{i,j}^t$ and $\delta_{i,j}^t$ as the Δ^t and $\delta_{i,j}^t$ arm i and j , and by similar argument we have with probability at least $1 - 1/T$, $\Delta_{i,j}^{t=C_1+1} \leq \sqrt{C_1 \log T}$. This means at round $t = C_1 + 1$, with probability at least $1 - K/T$, the number of rounds any arm get picked is no less than

$$n_1 = \frac{C_1 - (K - 1)\sqrt{C_1 \log T}}{K}$$

, and no greater than

$$n_2 = \frac{C_1 + (K-1)\sqrt{C_1 \log T}}{K}.$$

When $C_1 < t \leq C_1 + C_2$, denote X^j as the number of rounds between the target arm get its $(j-1)^{th}$ and j^{th} pick. After the target arm get its $(j-1)^{th}$ pick before its j^{th} pick, in the worst case, its beta distribution is $B(j, 1+n_2)$, and that of any other arm is $B(1, 1+n_1)$. By simple arithmetic calculation, we have when $j=1$, $\Pr\{\theta_1 < \theta_i\} = \frac{\beta}{1+\beta}$, and when $j \geq 2$, $\Pr\{\theta_1 < \theta_i\} \leq \frac{1}{j\beta}$ where $\beta = \frac{n_1+1}{n_2+1}$, so $\Pr\{\theta_1 > \theta_{i \neq 1}\} \geq (1 - \frac{1}{j\beta})^{K-1}$. When $j=1$, we have $\Pr\{\theta_1 > \theta_{i \neq 1}\} \geq (\frac{\beta}{1+\beta})^{K-1}$. The probability that the target arm be selected is at least $1/2^{K-1}$, When $j < \frac{1}{\beta(1-2^{1-K})} := n_3$, and at least $1/2^{K-1}$. when $j \geq n_3$. With probability at least $1 - 1/T$, the target arm will be picked for at least $(C_2 - n_3(\frac{\beta}{1+\beta})^{1-K} \log T)/2 - \sqrt{C_2 \log T}$ rounds.

We select C_1 and C_2 to be large enough such that with high probability, when $t > C_1 + C_2$, $\theta_1 > \mu/2$ and $\theta_{i \neq 1} < \mu/2$, so that the target arm will get all the picks. We set $C_1 = \frac{4 \log T}{\mu^2}$, and $C_2 = \max\{\frac{4 \log T}{\mu^2}, n_3(\frac{\beta}{1+\beta})^{1-K} \log T + 2\frac{\mu}{1-\mu}C_1\}$, then by $t = C_1 + C_2$, arms other than the target arm is picked for at least n_1 times, and the target arm's is picked for at least n_2 times with mean no less than μ . By result from [17], this can ensure that with probability at least $1 - K/T$, $\theta_{i \neq 1}^t < \mu/2$ and $\theta_1^t > \mu/2$ true for all rounds. So with probability at least $1 - (2K+1)/T$, the target arm will get all picks when $t > C_1 + C_2$, with C_1 and C_2 as given above. □

Proof for Theorem 8

Proof. With probability at least $1/T^2$, for every $t \geq 24 \log T/\epsilon$, the number of explore rounds is at least $|t_e| \geq t\epsilon/2$. Denote $r_i^t(h) = \mathbb{1}(h(x^t) = a_i) \mathbb{1}(a^t = a_i) \cdot c^t$. We have $[r_i^t(h)] = \frac{C_i(h)}{K}$ and $\hat{C}_i^t(h) = K \frac{\sum_{t \in t_e} x_i^t}{|t_e|}$. By Hoeffding inequality and union bound, for any

expert h , any arm i , at any round t , with probability at least $2/T$, we have

$$\frac{\sum_{t \in t_e} x_i^t}{|t_e|} \leq \sqrt{\frac{\log(T \cdot m \cdot K)}{|t_e|}}.$$

Combing both facts, with probability at least $1 - 3/T$, for any expert h , any arm i , at any round $t \geq 24 \log T/\epsilon$, we have

$$|\hat{C}_i^t(h) - C_i(h)| \leq K \sqrt{\frac{2 \log(T \cdot m \cdot K)}{t\epsilon}}.$$

□

proof for Theorem 15

Proof. Define a good event G that for any expert h , any arm i , at any round $t \geq 24 \log(T)/\epsilon$, the following is true

$$\Delta_C^t := |\hat{C}_i^t(h) - C_i(h)| \leq K \sqrt{\frac{2 \log(T \cdot m \cdot K)}{t\epsilon}}.$$

Denote $\hat{g}_i^t(y)$ as the function that used as the click probability function to decide payment.

Let $\mu_i^{t^*}$ be the bid that is actually the best bidding for agent i , we have:

$$(\mu_i - p_i^t(\mu_i)) \hat{g}_i^t(\mu_i) > (\mu_i - p_i^t(\mu_i^{t^*})) \hat{g}_i^t(\mu_i^{t^*}),$$

so the utility gap between bidding truthfully and best bidding Δ_u^t satisfies

$$\begin{aligned}
\Delta_u^t &= (\mu_i - p_i^t(\mu_i^t))g_i^t(\mu_i^t) - (\mu_i - p_i^t(\mu_i))g_i^t(\mu_i) \\
&\leq (\mu_i - p_i^t(\mu_i^t))(\hat{g}_i^t(\mu_i^t) + \Delta_c^t) - (\mu_i - p_i^t(\mu_i))(\hat{g}_i^t(\mu_i) - \Delta_c^t) \\
&\leq [(\mu_i - p_i^t(\mu_i^t))\hat{g}_i^t(\mu_i^t) - (\mu_i - p_i^t(\mu_i))\hat{g}_i^t(\mu_i)] + (\mu_i - p_i^t(\mu_i^t) + \mu_i - p_i^t(\mu_i))\Delta_c^t \\
&\leq 2\Delta_c^t
\end{aligned}$$

When G is true, that is, with probability at least $1 - 3/T$, for any $t \geq 24 \log(T)/\epsilon$ we have

$$\Delta_u^t \leq 2K \sqrt{\frac{2 \log(T \cdot m \cdot K)}{t\epsilon}}.$$

Therefore, when G is true, $\Delta_u^t \leq \alpha$ is true for all $t \geq \max\{24 \log(T)/\epsilon, \frac{8K^2 \log(T \cdot m \cdot K)}{\epsilon\alpha^2}\}$.

Considering the fact that $\alpha \leq 1$ and $K \geq 2$, $\max\{24 \log(T)/\epsilon, \frac{8K^2 \log(T \cdot m \cdot K)}{\epsilon\alpha^2}\} = \frac{8K^2 \log(T \cdot m \cdot K)}{\epsilon\alpha^2}$.

So with probability at least $1 - 3/T$, an α -truthful agent will bid truthfully for any round $t \geq \frac{8K^2 \log(T \cdot m \cdot K)}{\epsilon\alpha^2}$.

□

proof for Theorem 10

Proof. Define a good event G as all α -truthful agent will bid truthfully when $t \geq \frac{8K^2 \log(T \cdot m \cdot K)}{\epsilon\alpha^2} := t^*$. The probability that G is true is at least $1 - 3K/T$. The regret from the cases where G is not true is bound by $3K$. The regret from the rounds where agents might not be truthful is therefore bound by t^* . The regret from the explore rounds is bound by $T\epsilon$ in expectation. Next we focus on the regret from the exploit rounds after $t \geq t^*$.

The empirical estimation on expert reward function is given by $\hat{R}^t(h) = \sum_{i \in [K]} b_i^t c_i^t$. So the difference between the empirical estimation and true reward function of expert h

satisfies:

$$\Delta_R^t = |\hat{R}^t(h, \vec{b}^t) - R(h, \vec{b}^t)| \leq \sum_{i \in [K]} b_i^t \Delta_C^t \leq K \Delta_C^t.$$

Then the difference between the expected reward of the best expert h^* and the empirical best expert h^t satisfies:

$$\begin{aligned} R(h^*, \vec{b}^t) - R(h^t, \vec{b}^t) &\leq (\hat{R}(h^*, \vec{b}^t) + \Delta_R^t) - (\hat{R}(h^t, \vec{b}^t) + \Delta_R^t) \\ &= (\hat{R}(h^*, \vec{b}^t) - \hat{R}(h^t, \vec{b}^t)) + 2\Delta_R^t \\ &\leq 2\Delta_R^t \\ &\leq 2K\Delta_C^t \\ &\leq 2K^2 \sqrt{\frac{2 \log(T \cdot m \cdot K)}{t\epsilon}} \end{aligned}$$

So the total regret from these exploit rounds is bound by

$$R_{|t \geq t^*}(T) \leq \sum_{t=t^*}^T 2K^2 \sqrt{\frac{2 \log(T \cdot m \cdot K)}{t\epsilon}} \leq 2K^2 \sqrt{\frac{2T \log(T \cdot m \cdot K)}{\epsilon}}.$$

In total, the regret is bound by

$$\text{REG}(T) \leq T\epsilon + K^2 \sqrt{\frac{8T \log(T \cdot m \cdot K)}{\epsilon}} + 3K + \frac{8K^2 \log(T \cdot m \cdot K)}{\epsilon \alpha^2}$$

□

proof for Theorem 13

Proof. Recall that $\text{REG}(T) = T\rho_1\mu_1 - \sum_t \rho_{a^t}\mu_{a^t}$ where a^t is the arm picked at round t . Let's divide the $\text{REG}(T)$ into the regret from exploration rounds $\text{REG}(T)$ and the regret from exploitation rounds $\text{REG}(T)$.

The expected number of explore rounds is $T\epsilon$. Thus $\text{REG}(T) \leq T\epsilon$.

Define "good" event G such that at every round $t \geq 24K \log T/\epsilon$, for every arm j ,

$\hat{\rho}_j^t \in [\rho_j - w^t, \rho_j + w^t]$ is true. G happens with probability at least $1 - \frac{2(K+1)}{T}$. The regret from rounds where $t < 24K \log T/\epsilon$ can be bound by $24K \log T/\epsilon$, and henceforth we only focus on the rounds $t \geq 24K \log T/\epsilon$ where event G is defined.

When G is true and $t \geq 24K \log T/\epsilon$, the regret from this round is bound by:

$$\begin{aligned}
R^t &= \rho_1 \mu_1 - \rho_{a^t} \mu_{a^t} \\
&\leq (\hat{\rho}_1 + w^t) \mu_1 - (\hat{\rho}_{a^t} - w^t) \mu_{a^t} \\
&\leq w^t (\mu_1 + \mu_{a^t}) \\
&\leq 2w^t
\end{aligned} \tag{1}$$

Summing it over t gives an upper bound on the regret from exploit rounds when G is true and $t > 24K \log T/\epsilon$:

$$\text{REG}_{|G}(T) \leq \sum_{t=1}^T 2w^t \leq 2\left(\sqrt{\frac{2KT \log T}{\epsilon}} + 2C' \log T\right).$$

When event G doesn't occur, $R_{|\bar{G}}(T)$ is at most T . We get

$$\begin{aligned}
\text{REG}(T) &\leq 24K \log T/\epsilon + \Pr\{G\} \cdot \left(2\left(\sqrt{\frac{2KT \log T}{\epsilon}}\right) + 2C' \log T\right) + \Pr\{\bar{G}\}T \\
&\leq 24K \log T/\epsilon + 2\left(\sqrt{\frac{2KT \log T}{\epsilon}}\right) + 2C' \log T + 2(K+1).
\end{aligned} \tag{2}$$

Adding $\text{REG}(T)$ and $\text{REG}(T)$, we get

$$\text{REG}(T) \leq T\epsilon + 24K \log T/\epsilon + 2\sqrt{\frac{2KT \log T}{\epsilon}} + 4C' \log T + 2(K+1).$$

By taking $\epsilon = 2(K \log T)^{1/3}T^{-1/3}$, the welfare regret is bound by

$$\text{REG}(T) \leq 4(K \log T)^{1/3}T^{2/3} + 4C' \log T + 12K^{2/3}T^{1/3}(\log T)^{4/3} + 2(K + 1).$$

□

REFERENCES

- [1] J. G. March, “Exploration and exploitation in organizational learning,” *Organization science*, vol. 2, no. 1, pp. 71–87, 1991.
- [2] P. Dayan and N. D. Daw, “Decision theory, reinforcement learning, and the brain,” *Cognitive, Affective, & Behavioral Neuroscience*, vol. 8, no. 4, pp. 429–453, 2008.
- [3] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, “Deep reinforcement learning framework for autonomous driving,” *Electronic Imaging*, vol. 2017, no. 19, pp. 70–76, 2017.
- [4] H. Cai, K. Ren, W. Zhang, K. Malialis, J. Wang, Y. Yu, and D. Guo, “Real-time bidding by reinforcement learning in display advertising,” in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, 2017, pp. 661–670.
- [5] J. Kober, J. A. Bagnell, and J. Peters, “Reinforcement learning in robotics: A survey,” *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.
- [6] T. Lattimore and C. Szepesvári, *Bandit algorithms*. Cambridge University Press, 2020.
- [7] A. Garivier, T. Lattimore, and E. Kaufmann, “On explore-then-commit strategies,” *Advances in Neural Information Processing Systems*, vol. 29, pp. 784–792, 2016.
- [8] J. Langford and T. Zhang, “The epoch-greedy algorithm for contextual multi-armed bandits,” *Advances in neural information processing systems*, vol. 20, no. 1, pp. 96–1, 2007.
- [9] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, “The nonstochastic multiarmed bandit problem,” *SIAM journal on computing*, vol. 32, no. 1, pp. 48–77, 2002.
- [10] O. Caelen and G. Bontempi, “Improving the exploration strategy in bandit algorithms,” in *International Conference on Learning and Intelligent Optimization*, Springer, 2007, pp. 56–68.
- [11] P. Auer, N. Cesa-Bianchi, and P. Fischer, “Finite-time analysis of the multiarmed bandit problem,” *Machine learning*, vol. 47, no. 2, pp. 235–256, 2002.
- [12] S. Bubeck and N. Cesa-Bianchi, “Regret analysis of stochastic and nonstochastic multi-armed bandit problems,” *arXiv preprint arXiv:1204.5721*, 2012.

- [13] A. Garivier and O. Cappé, “The kl-ucb algorithm for bounded stochastic bandits and beyond,” in *Proceedings of the 24th annual conference on learning theory*, JMLR Workshop and Conference Proceedings, 2011, pp. 359–376.
- [14] L. Li, W. Chu, J. Langford, and R. E. Schapire, “A contextual-bandit approach to personalized news article recommendation,” in *Proceedings of the 19th international conference on World wide web*, 2010, pp. 661–670.
- [15] D. Zhou, L. Li, and Q. Gu, “Neural contextual bandits with ucb-based exploration,” in *International Conference on Machine Learning*, PMLR, 2020, pp. 11 492–11 502.
- [16] O. Chapelle and L. Li, “An empirical evaluation of thompson sampling,” *Advances in neural information processing systems*, vol. 24, pp. 2249–2257, 2011.
- [17] S. Agrawal and N. Goyal, “Analysis of thompson sampling for the multi-armed bandit problem,” in *Conference on learning theory*, JMLR Workshop and Conference Proceedings, 2012, pp. 39–1.
- [18] ———, “Thompson sampling for contextual bandits with linear payoffs,” in *International Conference on Machine Learning*, PMLR, 2013, pp. 127–135.
- [19] ———, “Further optimal regret bounds for thompson sampling,” in *Artificial intelligence and statistics*, PMLR, 2013, pp. 99–107.
- [20] J. Honda and A. Takemura, “Optimality of thompson sampling for gaussian bandits depends on priors,” in *Artificial Intelligence and Statistics*, PMLR, 2014, pp. 375–383.
- [21] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.
- [22] M. Tokic, “Adaptive ε -greedy exploration in reinforcement learning based on value differences,” in *Annual Conference on Artificial Intelligence*, Springer, 2010, pp. 203–210.
- [23] C. Painter-Wakefield and R. Parr, “Greedy algorithms for sparse reinforcement learning,” *arXiv preprint arXiv:1206.6485*, 2012.
- [24] E. Even-Dar, S. Mannor, Y. Mansour, and S. Mahadevan, “Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems,” *Journal of machine learning research*, vol. 7, no. 6, 2006.
- [25] D. W. Hosmer and S. Lemeshow, “Confidence interval estimation of interaction,” *Epidemiology*, pp. 452–456, 1992.

- [26] S. Boucheron, G. Lugosi, and O. Bousquet, “Concentration inequalities,” in *Summer school on machine learning*, Springer, 2003, pp. 208–240.
- [27] S. A. van de Geer, “On hoeffding’s inequality for dependent random variables,” in *Empirical process techniques for dependent data*, Springer, 2002, pp. 161–169.
- [28] B. Kveton, C. Szepesvari, S. Vaswani, Z. Wen, T. Lattimore, and M. Ghavamzadeh, “Garbage in, reward out: Bootstrapping exploration in multi-armed bandits,” in *International Conference on Machine Learning*, PMLR, 2019, pp. 3601–3610.
- [29] S. Vaswani, B. Kveton, Z. Wen, A. Rao, M. Schmidt, and Y. Abbasi-Yadkori, “New insights into bootstrapping for bandits,” *arXiv preprint arXiv:1805.09793*, 2018.
- [30] B. Kim and A. Tewari, “On the optimality of perturbations in stochastic and adversarial multi-armed bandit problems,” *arXiv preprint arXiv:1902.00610*, 2019.
- [31] S. Vaswani, A. Mehrabian, A. Durand, and B. Kveton, “Old dog learns new tricks: Randomized ucb for bandit problems,” *arXiv preprint arXiv:1910.04928*, 2019.
- [32] K. Misra, E. M. Schwartz, and J. Abernethy, “Dynamic online pricing with incomplete information using multiarmed bandit experiments,” *Marketing Science*, vol. 38, no. 2, pp. 226–252, 2019.
- [33] C. Schumann, Z. Lang, J. S. Foster, and J. P. Dickerson, “Making the cut: A bandit-based approach to tiered interviewing,” *arXiv preprint arXiv:1906.09621*, 2019.
- [34] T. Lykouris, V. Mirrokni, and R. Paes Leme, “Stochastic bandits robust to adversarial corruptions,” in *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, 2018, pp. 114–122.
- [35] M. Babaiouff, Y. Sharma, and A. Slivkins, “Characterizing truthful multi-armed bandit mechanisms,” *SIAM Journal on Computing*, vol. 43, no. 1, pp. 194–230, 2014.
- [36] K.-S. Jun, L. Li, Y. Ma, and X. Zhu, “Adversarial attacks on stochastic bandits,” *arXiv preprint arXiv:1810.12188*, 2018.
- [37] F. Liu and N. Shroff, “Data poisoning attacks on stochastic bandits,” in *International Conference on Machine Learning*, PMLR, 2019, pp. 4042–4050.
- [38] E. Garcelon, B. Roziere, L. Meunier, J. Tarbouriech, O. Teytaud, A. Lazaric, and M. Pirota, “Adversarial attacks on linear contextual bandits,” *arXiv preprint arXiv:2002.03839*, 2020.

- [39] A. Gupta, T. Koren, and K. Talwar, “Better algorithms for stochastic bandits with adversarial corruptions,” in *Conference on Learning Theory*, PMLR, 2019, pp. 1562–1578.
- [40] M. Braverman, J. Mao, J. Schneider, and M. Weinberg, “Selling to a no-regret buyer,” in *Proceedings of the 2018 ACM Conference on Economics and Computation*, 2018, pp. 523–538.
- [41] N. R. Devanur and S. M. Kakade, “The price of truthfulness for pay-per-click auctions,” in *Proceedings of the 10th ACM conference on Electronic commerce*, 2009, pp. 99–106.
- [42] K. Kandasamy, J. E. Gonzalez, M. I. Jordan, and I. Stoica, “Mechanism design with bandit feedback,” *arXiv preprint arXiv:2004.08924*, 2020.
- [43] K. K. Kapoor, Y. K. Dwivedi, and N. C. Piercy, “Pay-per-click advertising: A literature review,” *The Marketing Review*, vol. 16, no. 2, pp. 183–202, 2016.
- [44] K. C. Wilbur and Y. Zhu, “Click fraud,” *Marketing Science*, vol. 28, no. 2, pp. 293–308, 2009.
- [45] J. Liu, Z. Huang, and X. Wang, “Learning optimal reserve price against non-myopic bidders,” in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31, Curran Associates, Inc., 2018.
- [46] W. Chu, L. Li, L. Reyzin, and R. Schapire, “Contextual bandits with linear payoff functions,” in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, JMLR Workshop and Conference Proceedings, 2011, pp. 208–214.
- [47] R. B. Myerson, “Optimal auction design,” *Mathematics of operations research*, vol. 6, no. 1, pp. 58–73, 1981.
- [48] Y. Seldin and A. Slivkins, “One practical algorithm for both stochastic and adversarial bandits,” in *International Conference on Machine Learning*, PMLR, 2014, pp. 1287–1295.