Probabilistic Sequential Shrinking: A Best Arm Identification Algorithm for Stochastic Bandits with Corruptions

Zixin Zhong [1] **Wang Chi Cheung** [2 3] **Vincent Y. F. Tan** [1 3 4]

Abstract

We consider a best arm identification (BAI) problem for stochastic bandits with adversarial corruptions in the fixed-budget setting of T steps. We design a novel randomized algorithm, PROBABILISTIC SEQUENTIAL SHRINKING(u) (PSS(u)), which is agnostic to the amount of corruptions. When the amount of corruptions per step (CPS) is below a threshold, PSS(u) identifies the best arm or item with probability tending to 1 as $T \to \infty$. Otherwise, the optimality gap of the identified item degrades gracefully with the CPS. We argue that such a bifurcation is necessary. In PSS(u), the parameter u serves to balance between the optimality gap and success probability. The injection of randomization is shown to be essential to mitigate the impact of corruptions. To demonstrate this, we design two attack strategies that are applicable to any algorithm. We apply one of them to a deterministic analogue of PSS(u) known as SUCCESSIVE HALVING (SH) by Karnin et al. (2013). The attack strategy results in a high failure probability for SH, but PSS(u) remains robust. In the absence of corruptions, PSS(2)'s performance guarantee matches SH's. We show that when the CPS is sufficiently large, no algorithm can achieve a BAI probability tending to 1 as $T \to \infty$. Numerical experiments corroborate our theoretical findings.

1. Introduction

Consider a drug company D_1 that wants to design a vaccine for a certain illness, say COVID-19. It has a certain number of options, say $L = 10$, to design a near-optimal vaccine. Because D_1 has a limited budget, it can only test vaccines for a fixed number of times, say $T = 1000$. Using the limited number of tests, it wants to find the option that will lead to the "best" outcome, e.g., the shortest average recovery time of certain model organisms. However, vaccine trials usually assume that every test subject satisfies a certain set of criteria, such as having no prior related illnesses. If a subject who violated the criteria is tested, the observed recovery time would be *corrupted*. We assume the total corruption budget is bounded as a function of the number of tests. How can D_1 find a near-optimal drug design in the presence of the corruptions and uncertainty of the efficacy of the drugs? We will show that the utilization of a suitably *randomized* algorithm is assumed to be key.

To solve D_1's problem, we study the *Best Arm Identification* (BAI) problem for stochastic bandits with adversarial corruptions. We note that the effect and mitigation of corruptions were studied for the *Regret Minimization* problem by Lykouris et al. (2018) and others. While most existing works study the BAI problem for stochastic bandits *without* corruptions (Auer et al., 2002; Audibert & Bubeck, 2010; Carpentier & Locatelli, 2016), Altschuler et al. (2019) considers a variation of the classical BAI problem and aims to identify an item with high *median* reward, while Shen (2019) assumes that the amount of corruption per step (CPS) *diminishes* as time progresses. Therefore, these studies are not directly applicable to D_1 as we are interested in obtaining a near-optimal item in terms of the mean and we assume that the CPS does not diminish with time. Our setting dovetails neatly with company D_1's problem and D_1 can utilize our algorithm to sequentially and adaptively select different design options to test the vaccines and to eventually choose a near-optimal design that results in a short recovery time even in the presence of adversarial corruptions.

Beyond any specific applications, we believe that this problem is of fundamental theoretical importance in the broad context of BAI in multi-armed bandits (MAB) and adversarial machine learning. In particular, Gupta et al. (2019) advanced the theory of regret minimization in MAB; this work complements Gupta's work in the BAI setting.

Main Contributions. In stochastic bandits with adversarial

[1]Department of Mathematics, National University of Singapore, Singapore [2]Department of Industrial Systems and Management, National University of Singapore, Singapore [3]Institute of Operations Research and Analytics, National University of Singapore, Singapore [4]Department of Electrical and Computer Engineering, National University of Singapore, Singapore. Correspondence to: Zixin Zhong <zixin.zhong@u.nus.edu>, Wang Chi Cheung <isecwc@nus.edu.sg>, Vincent Y. F. Tan <vtan@nus.edu.sg>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

corruptions, there are L items with different rewards distributions. At each time step, a random reward is generated from each item's distribution; this reward is observed and arbitrarily corrupted by the adversary. The learning agent selects an item based on corrupted observations in previous steps, and only observes the pulled items' corrupted rewards. Given $T \in \mathbb{N}$, the agent aims to identify a near-optimal item with high probability over T time steps. Our first main contribution is the PROBABILISTIC SEQUENTIAL SHRINKING(u) (PSS(u)) algorithm. PSS(u) is agnostic to the amount of adversarial corruption. The parameter u can be adjusted to trade-off between the optimality gap of the identified item and the success probability.

The key challenge lies in mitigating the impact of corruptions. For this purpose, upon observing pulled items' corrupted rewards in previous time steps, PSS(u) pulls subsequent items probabilistically. By comparing PSS(u) to the state-of-the art for BAI with fixed horizon, namely SUCCESSIVE HALVING (SH) by Karnin et al. (2013), we argue that randomization is beneficial, and indeed necessary, for BAI under adversarial corruption. On one hand, PSS(2)'s failure probability in BAI (at least in the exponent) matches that of SH when there is no corruption. On the other hand, the largest possible CPS under which PSS(2) succeeds in BAI with probability $1-\exp(-\Theta(T))$ is a factor of L larger than that for SH. En route, we identify a term in the exponent of the failure probability of PSS(u) that generalizes the ubiquitous H_2 term for BAI under the fixed-budget setting. Finally, when CPS is so large that BAI is impossible, the sub-optimality gap of the identified item degrades gracefully with the CPS. In complement, we provide lower bound examples to show that BAI is impossible when CPS is sufficiently large. The examples involve judiciously chosen attack strategies, which corroborate the tightness of our performance guarantee for PSS(u). Numerical experiments on various settings further corroborate our theoretical findings.

Novelty. (i) We identify randomization as a key tool in mitigate corruption in BAI, and identify an achievable sub-optimality gap for PSS(u). (ii) The analysis of PSS(u) shows how our designed randomization "confuses the adversary", which results in the improvement over SH, and yields (suboptimality gap and failure probability exponent) results that are almost *tight* with respect to the lower bounds. (iii) The design of the attack strategies, which involves a randomized adversary, and their analysis are novel.

Literature review. The BAI problem has been studied extensively for both stochastic bandits (Audibert & Bubeck, 2010) and bandits with adversarial corruptions (Shen, 2019). There are two complementary settings for BAI: (i) Given $T \in \mathbb{N}$, the agent aims to maximize the probability of finding a near-optimal item in at most T steps; (ii) Given $\delta > 0$, the agent aims to find a near-optimal item with the probability of at least $1-\delta$ in the smallest number of steps. These settings are respectively known as the *fixed-budget* and *fixed-confidence* settings. Another line of studies aims to prevent the agent from achieving the above desiderata and thus to design strategies to *attack* the rewards efficiently (Jun et al., 2018; Liu & Lai, 2020). We now review some works.

First, we review related work in stochastic bandits. Both the fixed-budget setting (Audibert & Bubeck, 2010; Karnin et al., 2013; Jun et al., 2016) and the fixed-confidence setting (Audibert & Bubeck, 2010; Chen et al., 2014; Rejwan & Mansour, 2020; Zhong et al., 2020) have been extensively studied. However, as previously motivated, we need to be cognizant that the agent may encounter corrupted rewards and thus must design appropriate strategies to nullify or minimize the effects of these corruptions.

Regret minimization on stochastic bandits with corruptions was first studied by Lykouris et al. (2018), and has attracted extensive interest recently (Zimmert & Seldin, 2019; Li et al., 2019; Gupta et al., 2019; Lykouris et al., 2020; Liu & Lai, 2020; Krishnamurthy et al., 2020; Bogunovic et al., 2020). Pertaining to the BAI problem in the presence of corruptions, Altschuler et al. (2019) studies a variation of the classical fixed-confidence setting and aims to find an item with a high median reward. In contrast, Shen (2019) proposes an algorithm under the fixed-budget setting, whose theoretical guarantee requires a number of stringent conditions. In particular, Shen (2019) assumes that CPS diminishes as time progresses. However, it may be hard to verify in practice whether these conditions are satisfied. In spite of the many existing works, the classical BAI problem has not been analyzed when the rewards suffer from general corruptions. Our work fills in this gap in the literature by proposing and analyzing the PSS(u) algorithm under the fixed-budget setting. The randomized design of our algorithm is crucial in mitigating the impact of corruptions.

Another concern is how an adversary can corrupt the rewards to prevent the agent from obtaining sufficient information from the corrupted observations. Many studies aim at attacking certain algorithms, such as UCB, ϵ-greedy or Thompson sampling, using an adaptive strategy (Jun et al., 2018; Zuo, 2020). Liu & Shroff (2019) design offline strategies to attack a particular algorithm and also an adaptive strategy against any algorithm. All these strategies aim to corrupt the rewards such that the agent can only receive a small cumulative reward in expectation. The design and analysis of attack strategies pertaining to the BAI problem have been unexplored. Our analysis fills in this gap by proposing two offline strategies for Bernoulli instances and proving that when the total corruption budget is sufficiently large (i.e., of the order[1] $\Omega(T)$), *any* algorithm will fail to identify any

[1] A (non-negative) function $f(T) = O(T)$ if there exists a constant $0 < c < \infty$ (dependent on w but not T) such that

near-optimal item with constant probability.

2. Problem Setup

For brevity, for any $n \in \mathbb{N}$, we denote the set $\{1, \dots, n\}$ as $[n]$. Let there be $L \in \mathbb{N}$ ground items, contained in $[L]$. Each item $i \in [L]$ is associated with a reward distribution $\nu(i)$ supported in $[0, 1]$ with mean $w(i)$. The distributions $\{\nu(i)\}_{i\in[L]}$ and means $\{w(i)\}_{i\in[L]}$ are not known to the agent. Over time, the agent is required to identify the best or close-to-best ground item by adaptively pulling items. The agent aims to identify an optimal item, which is an item of the highest mean reward, after a fixed time budget of $T \in \mathbb{N}$ time steps, whenever possible in the presence of corruptions. More precisely, at each time step $t \in [T]$,

(i) A stochastic reward $W_t(i) \in [0, 1]$ is drawn for each item i from $\nu(i)$.

(ii) The adversary observes $\{W_t(i)\}_{i\in[L]}$, and corrupts each $W_t(i)$ by an additive amount $c_t(i) \in [-1, 1]$, leading to the corrupted reward $\tilde{W}_t(i) = W_t(i) + c_t(i) \in [0, 1]$ for each $i \in [L]$.

(iii) The agent pulls an item $i_t \in [L]$ and observes the corrupted reward $\tilde{W}_t(i_t)$.

For each $i \in [L]$, the random variables in $\{W_t(i)\}_{t=1}^T$ are i.i.d. When determining $\{c_t(i)\}_{i\in[L]}$ at time step t, the adversary cannot observe the item i_t going to be pulled, but he can utilize the current observations consisting of $\{W_q(1), \dots, W_q(L)\}_{q=1}^t$, $\{c_q(1), \dots, c_q(L)\}_{q=1}^{t-1}$, and $\{i_q\}_{q=1}^{t-1}$. We assume that the total amount of adversarial corruptions during the horizon is bounded:

$$\sum_{t=1}^{T} \max_{i\in[L]} |c_t(i)| \le C.$$

The *corruption budget* C is not known to the agent.

We focus on instances with a unique item of the highest mean reward, and assume that $w(1) > w(2) \ge \dots \ge w(L)$, so that item 1 is the unique optimal item. To be clear, the items can, in general, be arranged in any order; the ordering that $w(i) \ge w(j)$ for $i < j$ is just to ease our discussion. We denote $\Delta_{1,i} = w(1) - w(i)$ as the *optimality gap* of item i. An item i is ϵ-*optimal* ($\epsilon \ge 0$) if $\Delta_{1,i} \le \epsilon$.

The agent uses an *online algorithm* π to decide the item i_t^π to pull at each time step t, and the item $i_{\text{out}}^{\pi,T}$ to output as the identified item eventually. More formally, an online algorithm consists of a tuple $\pi := ((\pi_t)_{t=1}^T, \phi^{\pi,T})$, where

- the *sampling rule* π_t determines, based on the observation history, the item i_t^π to pull at time step t. That is, the random variable i_t^π is $\mathcal{F}_{t-1}$-measurable, where $\mathcal{F}_t := \sigma(i_1^\pi, \tilde{W}_1(i_1^\pi), \dots, i_t^\pi, \tilde{W}_t(i_t^\pi))$;
- the recommendation rule $\phi^{\pi,T}$ chooses an item $i_{\text{out}}^{\pi,T}$, that is, by definition, $\mathcal{F}_T$-measurable.

We denote the probability law of the process $\{\tilde{\mathbf{W}}_t = (\tilde{W}_t(1), \dots, \tilde{W}_t(L))\}_{t=1}^T$ by $\mathbb{P}$. This probability law depends on the agent's online algorithm π, which influences the adversarial corruptions.

For fixed $\epsilon_C, \delta \in (0, 1)$, an algorithm π is said to be (ϵ_C, δ)-*PAC (probably approximately correct)* if

$$\mathbb{P}\big[\Delta_{1,i_{\text{out}}^{\pi,T}} > \epsilon_C\big] \le \delta.$$

Our overarching goal is to design an (ϵ_C, δ)-PAC algorithm π such that both ϵ_C and δ are small. In particular, when $\epsilon_C < \Delta_{1,2}$, an (ϵ_C, δ)-PAC algorithm π identifies the optimal item with probability at least $1 - \delta$. For BAI with no corruption, existing works (Audibert & Bubeck, 2010; Karnin et al., 2013) provide $(0, \exp[-\Theta(T)])$-PAC algorithms. In the presence of corruptions, unfortunately it is impossible to achieve a $(0, \exp[-\Theta(T)])$-PAC performance guarantee, as we discuss in the forthcoming Section 4.2. We investigate the trade-off between ϵ_C and δ, and focus on constructing $(\epsilon_C, \exp[-\Theta(T)])$-PAC algorithms with ϵ_C as small as possible. We abbreviate i_t^π as i_t and $i_{\text{out}}^{\pi,T}$ as i_{out} when there is no ambiguity.

Finally, in anticipation of our main results, we remark that given a failure probability δ, the smallest possible ϵ_C is, in general, a function of the *corruption per step* (CPS) C/T and *possibly* the total number of items L.

3. Algorithm

Our algorithm PROBABILISTIC SEQUENTIAL SHRINKING (u) (PSS(u)) is presented in Algorithm 1. The algorithm involves randomization in order to mitigate the impact of adversarial corruptions.

The agent partitions the whole horizon into $\lceil \log_u L \rceil$ phases of equal length. During each phase, PSS(u) classifies an item as *active* or *inactive* based on the empirical averages of the corrupted rewards. Initially, all ground items are active and belong to the *active set* A_0. Over phases, the active sets A_m shrink, and an item may be eliminated from A_m and consequently it may become *inactive*.

During phase m:

(i) at each time step, the agent chooses an active item uniformly at random and pulls it;

(ii) at the end, the agent finds $\hat{w}_m(i)$, the corrupted empirical mean during phase m for each active item i;

(iii) the agent utilizes the $\hat{w}_m(i)$'s of active items $i \in A_{m-1}$ to shrink the active set.

$f(T) \le cT$ for sufficiently large T. Similarly $f(T) = \Omega(T)$ if there exists $c > 0$ such that $f(T) \ge cT$ for sufficiently large T. Finally, $f(T) = \Theta(T)$ if $f(T) = O(T)$ and $f(T) = \Omega(T)$.

Algorithm 1 PROBABILISTIC SEQUENTIAL SHRINKING

1: **Input:** time budget T, size of ground set of items L, parameter $u \in (1, L]$.
2: Set $M = \lceil \log_u L \rceil$, $N = \lfloor T/M \rfloor$, $T_0 = 0$, $A_0 = [L]$.
3: **for** phase $m = 1, 2, \ldots, M$ **do**
4: Set $T_m = T_{m-1} + N$, $q_m = 1/|A_{m-1}|$, $n_m = q_m N$.
5: **for** $t = T_{m-1} + 1, \ldots, T_m$ **do**
6: Choose item $i \in A_{m-1}$ with probability q_m, pull it and observe corrupted reward $\tilde{W}_t(i)$.
7: **end for**
8: For all $i \in A_{m-1}$, set
$$S_m(i) = \sum_{t=T_{m-1}+1}^{T_m} \tilde{W}_t(i_t) \cdot \mathbb{I}\{i_t = i\},\ \hat{w}_m(i) = \frac{S_m(i)}{n_m}.$$
9: Let A_m contain the $\lceil L/u^m \rceil$ items with the highest empirical means $\hat{w}_m(i)$'s in A_{m-1}.
10: **end for**
11: Output the single item $i_{\text{out}} \in A_M$.

By the end of the last phase M, we show that $|A_M| = 1$ (see Lemma 5.1 in Section 5), and the agent outputs the single active item.

The effectiveness of Algorithm 1 is manifested in four different aspects: (i) the agent only utilizes information from the *current* phase to shrink the active set, which ensures that any corruption has a limited impact on her decision; (ii) the injection of randomization by the agent to decide on which item to pull nullifies the ability of the adversary from corrupting rewards of *specific* items; (iii) the agent can handle the adversarial attacks even though she does not know the total corruption budget C; (iv) the agent can choose any $u \in (1, L]$ to trade off between ϵ_C and δ in its (ϵ_C, δ)-PAC performance guarantee. A smaller ϵ_C leads to a higher failure probability δ. We would like to emphasize that though the agent can choose any $u \in (1, L]$, this parameter is a fixed constant and cannot vary with the horizon T after PSS(u) is initialized.

When $u = L$, PSS(L) regards the horizon T as a single phase. Each item is pulled with probability $1/L$ at each step, and is expected to be pulled for T/L times in T steps. We can regard PSS(L) as a randomized version of the naïve UNIFORM PULL (UP) algorithm, which pulls each item for $\lfloor T/L \rfloor$ times according to a deterministic schedule.

When $u = 2$, PSS(2) is a randomized analogue to the SEQUENTIAL HALVING (SH) algorithm proposed in Karnin et al. (2013). Both PSS(2) and SH divide the whole horizon into $\lceil \log_2 L \rceil$ phases and halve the active set during each phase, i.e., $A_m = \lceil L/2^m \rceil$. However, the differences between them are as follows:

- at each time step of phase m, PSS(2) chooses item $i \in A_{m-1}$ with probability $1/|A_{m-1}|$ and pulls it (Line 6 of Algorithm 1);
- during phase m, SH pulls each item in A_{m-1} for *exactly* $\lfloor T/(\lceil \log_2 L \rceil \cdot |A_{m-1}|) \rfloor$ times according to a deterministic schedule.

Therefore, though PSS(2) and SH pull each active item for about an equal number of times in expectation, PSS(2) involves more randomness in the pulls.

4. Main Results

4.1. Upper Bound

Theorem 4.1. *For any $u \in (1, L]$, the* PROBABILISTIC SEQUENTIAL SHRINKING(u) *algorithm, as presented in Algorithm 1, outputs an item i_{out} satisfying*

$$\begin{aligned}
&\mathbb{P}\left[\Delta_{1,i_{\text{out}}} > \frac{8C\lceil \log_u L \rceil}{T}\right] \\
&\le 4\lceil \log_u L \rceil (L-1) \exp\left[-\frac{1}{192\tilde{H}_2(w,L,u)} \cdot \left\lfloor \frac{T}{\lceil \log_u L \rceil} \right\rfloor\right] \\
&= O\left(L(\log_u L) \exp\left[-\frac{T}{192\tilde{H}_2(w,L,u)\log_u L}\right]\right), \quad (4.1)
\end{aligned}$$

where

$$\tilde{H}_2(w, L, u) = \max_{i \neq 1} \frac{\min\{u \cdot i,\ L\}}{\Delta_{1,i}^2}. \quad (4.2)$$

Theorem 4.1 shows that PSS(u) is (ϵ_C, δ)-PAC for any $u \in (1, L]$, where

$$\epsilon_C = O\left(\frac{C \log L}{T}\right) \quad \text{and} \quad \delta = \exp[-\Theta(T)].$$

We remark that only ϵ_C, but not δ, depends on C. The dependence of ϵ_C on the CPS C/T is, in general, unavoidable in view of our lower bounds (see Section 4.2).

The upper bound on the failure probability δ involves the parameter $\tilde{H}_2(w, L, u)$, which quantifies the difficulty of identifying the best item in the instance. The parameter $\tilde{H}_2(w, L, u)$ generalizes its analogue

$$H_2(w) = \max_{i \neq 1} \frac{i}{\Delta_{1,i}^2}$$

proposed by Audibert & Bubeck (2010), in the sense that

$$\lim_{u \to 1^+} \tilde{H}_2(w, L, u) = H_2(w), \quad \forall\, w \in [0, 1]^L.$$

We propose to consider the more general version $\tilde{H}_2(w, L, u)$ in order to analyze the randomized versions of SH and UP under one unified framework.

Function of parameter u. Theorem 4.1 implies that when u increases, the upper bound ϵ_C on $\Delta_{1,i_{\text{out}}}$ decreases. However, the quantity $\tilde{H}_2(w, L, u)$ increases, which leads to a

larger upper bound on the failure probability. Specifically,

$$\tilde{H}_2(w, L, u_2) \geq \frac{u_2}{u_1}\tilde{H}_2(w, L, u_1), \quad \forall\, 1 < u_1 \leq u_2 \leq L.$$

Meanwhile, as presented in Algorithm 1, PSS(u) with a larger u separates the whole horizon into fewer phases and shrinks the active set faster. (i) The fewer number of phases leads to a longer duration of each phase, which is beneficial for bounding the impact of corruptions (see Lemma 5.2). (ii) Besides, the faster the active sets shrink, the larger $\tilde{H}_2(w, L, u)$ is. See Section C.4 for details.

Altogether, Theorem 1 provides a bound on learning an ϵ_C-optimal item and implies that PSS(u) allows the agent to trade off between the bound on $\Delta_{1,i_{\text{out}}}$ and the failure probability by adjusting u. When the CPS is so low that

$$\frac{C}{T} < \frac{\Delta_{1,2}}{8\lceil \log_u L\rceil}, \tag{4.3}$$

Theorem 4.1 implies that PSS(u) identify the optimal item with probability at least $1-\delta$, where $\delta = \exp(-\Theta(T))$ is as shown in (4.1). When the CPS is so large such that

$$\frac{C}{T} \geq \frac{\Delta_{1,L}}{8\lceil \log_u L\rceil}, \tag{4.4}$$

Theorem 4.1 is vacuous, since all the items are $\Delta_{1,L}$-optimal. In the extreme case in which

$$\frac{C}{T} \geq \sup_{u\in(1,L]} \frac{\Delta_{1,L}}{8\lceil \log_u L\rceil} = \frac{\Delta_{1,L}}{8},$$

Theorem 4.1 is vacuous for all $u \in (1, L]$. Indeed, we show in Section 4.2 that this bifurcation on the learnability holds true not only to our algorithms. No algorithm can achieve BAI when C/T is above a certain threshold. In passing, our characterization of the threshold is tight up to log factors.

BAI on stochastic setting without corruptions. In the setting without adversarial corruptions, i.e., $C = 0$, Theorem 1 upper bounds the probability that PSS(u) outputs i_{out} with $\Delta_{1,i_{\text{out}}} > 0$. We compare Theorem 4.1 on PSS(2) with the performance guarantee of SH by Karnin et al. (2013):

$$\mathbb{P}[\Delta_{1,i_{\text{out}}} > 0] = O\left((\log_2 L)\exp\left[-\frac{T}{8H_2(w)\log_2 L}\right]\right).$$

Disregarding constants, the bound on $\mathbb{P}[\Delta_{1,i_{\text{out}}} > 0]$ of PSS(2) is worse than that of SH by a factor of L, which is a multiplicative factor we incur due to the impact of corruptions. Apart from that, our bound involves $\tilde{H}_2(w, L, 2)$ while Karnin et al. (2013) involves $H_2(w)$, and notice that

$$\tilde{H}_2(w, L, 2) \leq 2H_2(w).$$

As a result, our exponent matches that by Karnin et al. (2013) up to an absolute constant (which is 48).

Table 1. Comparison of PSS(u) to Other Algorithms

Algorithm	Order of ϵ_C	Order of δ
PSS(u)	$\frac{C\log_u L}{T}$	$L(\log_u L)\exp\left[-\frac{T}{192\tilde{H}_2(w,L,u)\log_u L}\right]$
PSS(2)	$\frac{C\log_2 L}{T}$	$L(\log_2 L)\exp\left[-\frac{T}{192\tilde{H}_2(w,L,2)\log_2 L}\right]$
SH	$\frac{CL\log_2 L}{T}$	$L(\log_2 L)\exp\left[-\frac{T}{192\tilde{H}_2(w,L,2)\log_2 L}\right]$
PSS(L)	$\frac{C}{T}$	$L\exp\left(-\frac{T}{192L/\Delta_{1,2}^2}\right)$
UP	$\frac{CL}{T}$	$L\exp\left(-\frac{T}{192L/\Delta_{1,2}^2}\right)$

Next, we compare Theorem 4.1 on PSS(L) with the performance guarantee of UP, which is folklore. We use the following in Section 33.3 of Lattimore & Szepesvári (2020):

$$\begin{aligned}\Pr[\Delta_{1,\text{out}} > 0] &\leq \sum_{i=2}^{L}\exp\left[-\frac{\lfloor T/L\rfloor\cdot\Delta_{1,i}^2}{4}\right]\\ &\leq (L-1)\exp\left[-\frac{\lfloor T/L\rfloor\cdot\Delta_{1,2}^2}{4}\right],\end{aligned} \tag{4.5}$$

where (4.5) is tight when $\Delta_{1,2} = \Delta_{1,i}$ for all $i \neq 1$. For PSS(L), $\tilde{H}_2(w, L, L) = L/\Delta_{1,2}^2$, and the failure probability bound in (4.1) specializes to

$$O\left(L\exp\left[-T\cdot\frac{\Delta_{1,2}^2}{192\cdot L}\right]\right),$$

which matches (4.5) up to multiplicative factors in the exponent and the $O(\cdot)$ notation.

Comparisons in the corrupted setting. Though the SH and the UP algorithms can be directly applied to the setting with corruptions, we propose PSS(u) to inject randomness in order to mitigate the impact of corruptions. Intuitively, for an adversary with the knowledge of the algorithm, the fact that a deterministic algorithm such as SH or UP pulls each active item according to a deterministic schedule fixed at the start of a phase allows the adversary to corrupt rewards of the items to be pulled. However, PSS(u) pulls items *probabilistically*, which prevents the adversary from identifying the items to be pulled even when the semantics of the algorithm are known to the adversary.

We analyse SH and UP using a similar analysis to our proof of Theorem 4.1, and we tabulate the (ϵ_C, δ)-PAC performance guarantee in Table 1. While SH and UP have similar performance guarantees on δ compared to their randomized counterparts, namely PSS(u), the upper bounds on ϵ_C for SH and UP are *larger* than their randomized counterparts by a multiplicative factor of L. Consequently, the

randomization in PSS(u) allows us to mitigate the adversarial corruptions and leads to a better performance guarantee on ϵ_C compared to its deterministic counterparts.

4.2. Lower bounds

In the previous section, we observed that the performance guarantee of PSS(u) on ϵ_C deteriorates as the CPS increases. Interestingly, the deterioration is, in fact, fundamental to any online algorithm. Here, we demonstrate that no online algorithm is able to identify the optimal item with vanishing failure probability when C/T is above a certain threshold. In fact, one attack strategy we design is shown to cause SH to fail miserably; in contrast, PSS(2) remains robust to it. The impossibility result is further generalized to the identification of an ϵ-optimal item for any $\epsilon \in [0, \Delta_{1,L})$.

Bernoulli instance. We focus on instances where each item $i \in [L]$ follows Bern$(w(i))$, and $1 > w(1) > w(2) \geq w(3) \geq \ldots \geq w(L) > 0$. For any $\epsilon \in (0,1)$, we use $L_\epsilon := |\{i \in [L] : \Delta_{1,i} \leq \epsilon\}|$ to count the number of items with mean reward at most ϵ worse than that of the optimal item.

Corruption strategy against general BAI algorithms. Abbreviate $\Delta_{1,2}$ as Δ. Assume that $w(2) - \Delta > w(3)$. In this strategy, essentially, the adversary solely corrupts the reward of item 1, so that $\tilde{W}_t(1) \sim$ Bern$(w(2) - \Delta)$, different from $W_t(1) \sim$ Bern$(w(1))$, as long as there is enough corruption budget (Figure 1). We describe the corruption strategy in full in Appendix C.5.

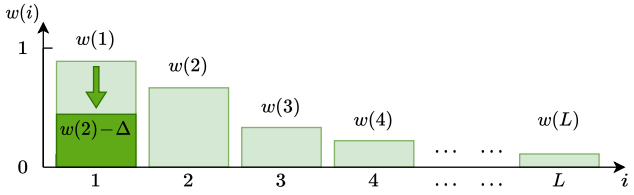


Figure 1. $\tilde{W}_t(1) \sim$ Bern$(w(2) - \Delta)$

For a BAI with adversarial corruptions instance, we say that the instance has an optimality gap $\Delta > 0$ if $\Delta = \Delta_{1,2} > 0$.

Theorem 4.2. *Fix $\lambda \in (0,1)$ and $\Delta \in (0, 1/2)$. For any online algorithm, there is a BAI with an adversarial corruption instance in T steps, corruption budget $C = 1 + (1+\lambda)2\Delta T$, and optimality gap Δ, such that*

$$\mathbb{P}[\Delta_{1,i_{\text{out}}} > 0] = \mathbb{P}[\Delta_{1,i_{\text{out}}} \geq \Delta] = \mathbb{P}[i_{\text{out}} \neq 1] \geq \frac{1}{2} \cdot \left[1 - \exp\left(-\frac{2\lambda^2 \Delta T}{3}\right)\right].$$

In particular, Theorem 4.2 implies that, if the CPS satisfies

$$C/T > 2\Delta_{1,2}, \tag{4.6}$$

then it is impossible to identify the best item with probability $1 - \exp[-\Theta(T)]$. The upper bound in (4.3) and the lower bound in (4.6) differ by a multiplicative factor of $16\lfloor \log_u L \rfloor$. Consequently, the upper bound in (4.3) is within a factor of $O(\log_u L)$ away from the largest possible upper bound on CPS C/T, under which it is possible to identify the best item with probability at least $1 - \exp[-\Theta(T)]$.

Robustness of PSS(2) with respect to SH. Consider Theorem 4.2's attack strategy (see Figure 1 and Appendix C.5), but applied to SH only in phase 1. We can show that SH will fail to identify the best item with probability at least $1/2$.

Theorem 4.3. *Fix $L > 1$, $\lambda \in (0,1)$ and $\Delta \in (0, 1/4)$. For the SH algorithm, there is a BAI with adversarial corruption instance with T time steps, corruption budget $C = (1+\lambda)2\Delta T/(L \log_2 L)$, and optimality gap Δ, such that if T is sufficiently large,*

$$\mathbb{P}[\Delta_{1,i_{\text{out}}} > 0] = \mathbb{P}[\Delta_{1,i_{\text{out}}} \geq \Delta] = \mathbb{P}[i_{\text{out}} \neq 1] \geq 1/2.$$

Consequently, if $C/T \geq \Delta_{1,2}/(L \log_2 L)$, SH fails to identify the best item with probability at least $1/2$ for large T. In contrast, PSS(2) identifies the best item with probability at least $1 - \exp(-\Theta(T))$ as long as $C/T = O(\Delta_{1,2}/\log_2 L)$ (see (4.3)). Lastly, according to Table 1, SH would succeed with high probability if $C/T \leq O(\Delta_{1,L}/(L \log_2 L))$. Hence, the upper and lower bounds of the CPS for SH are tight, *even up to log factors in L*.

The failure of SH is due to the fact that according to the observation history, the adversary knows the item to pull at each time step. In contrast, for PSS(2), when determining $\{c_t(i)\}_{i \in [L]}$, *the adversary only knows $\{W_t(i)\}_{i \in [L]}$, but does not know i_t*. Rather, he only knows the *distribution* of i_t. This uniform distribution facilitates exploration, while minimizing the leakage of the identity of i_t to the adversary; this leads to an improvement by a factor of $\tilde{O}(L)$ on ϵ_C.

Corruption strategy against identifying an ϵ-optimal item. We extend the previous strategy in order to impede the identification of an ϵ_C-optimal item for any $\epsilon_C \in [0, \Delta_{1,L})$. Consider the following two offline strategies:

(I) at each time step, if the random reward is 1, the adversary shifts it to 0 until the corruption amount is depleted (see Figure 2);

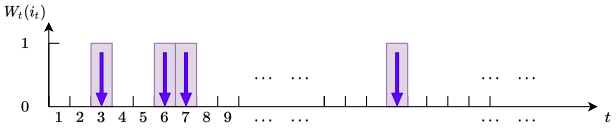


Figure 2. Shift $W_t(i_t)$ to 0 When $W_t(i_t) = 1$

(II) at each time step, if the random reward is 0, the adversary shifts it to 1 until the corruption amount is depleted.

The design of either strategy aims to make the agent obtain the same random reward at all time steps. As a result, the

agent fails to get any information from the observations. In this case, the best thing she can do is to output any item with a uniform probability of $1/L$ after T time steps.

Theorem 4.4. *Fix any $\lambda, \epsilon \in (0,1)$. If $C \geq L \cdot \{1-(1-\lambda)[1-w(1)]\} \cdot T$, Strategy (I)'s attack results in*

$$\mathbb{P}[\Delta_{1,i_{\text{out}}} > \epsilon] \geq 1 - \frac{L_\epsilon}{L} - \exp\left[-\frac{\lambda^2 TL[1-w(1)]}{2}\right].$$

If instead $C \geq L \cdot [1-(1-\lambda)w(L)] \cdot T$, Strategy (II)'s attack results in

$$\mathbb{P}[\Delta_{1,i_{\text{out}}} > \epsilon] \geq 1 - \frac{L_\epsilon}{L} - \exp\left[-\frac{\lambda^2 TLw(L)}{2}\right].$$

When $\epsilon < \Delta_{1,2}$ so that $L_\epsilon = 1$, Theorem 4.4 provides lower bounds for the probability of identifying the optimal item under corruption strategies (I) and (II) respectively. In this case, when $T \to \infty$, the failure probability is asymptotically lower bounded by $1 - 1/L$.

Although the adversary can use adaptive strategies to attack random rewards, i.e., design a strategy to add corruptions according to past observations, Theorem 4.4 shows that when the corruption is sufficiently large, even an *offline* strategy, i.e., one that is *fixed* before the algorithm runs, prevents the agent from identifying a satisfactory item with high probability. Thus, if $C = \Omega(T)$, any algorithm will fail to identify a near-optimal item with high probability. Therefore, PSS(u) is tight up to a factor that differs from $\log_u L$ in Theorem 4.1 to L in Theorem 4.4.

5. Proof Sketch of Theorem 4.1

We provide the proof sketch for Theorem 4.1. The detailed proof and those of Theorem 4.2–4.4 are deferred to the supplementary material.

Feasibility. We first show that our algorithm is feasible in the sense that the M phases proceed within T steps, and A_M is a singleton.

Lemma 5.1. *It holds that $NM \leq T$ and $|A_M| = 1$.*

Lemma 5.1 ensures that i_{out} is well-defined. Moreover, it implies that $\{1 \neq i_{\text{out}}\} = \{1 \notin A_M\}$.

Concentration. At the end of phase m ($1 \leq m \leq M$), the agent shrinks the active set A_{m-1} according to the $\hat{w}_m(i)$'s, the corrupted empirical means of the active items. Intuitively, we expect that if $\hat{w}_m(i)$ and $w(i)$ are sufficiently close, we can identify item i with small $\Delta_{1,i}$. To this end, we define the amount of corruptions during phase m as

$$C_m := \sum_{t=T_{m-1}+1}^{T_m} \max_{i \in [L]} |c_t(i)|.$$

To estimate the gap between $\hat{w}_m(i)$ and $w(i)$, we define a class of "nice events" for all $i \in A_{m-1}$ and $a \in (0,1)$:

$$\mathcal{E}_{m,i}^{(\text{U})}(a) := \left\{\hat{w}_m(i) < w(i) + \frac{2C_m}{N} + 2a\right\},$$
$$\mathcal{E}_{m,i}^{(\text{L})}(a) := \left\{\hat{w}_m(i) > w(i) - \frac{2C_m}{N} - 2a\right\}.$$

We utilize Theorem B.1 and B.2 to show that all these events hold with high probability. In particular, Theorem B.2 allows us to bound the impact of corruptions.

Lemma 5.2. *Let $\overline{\mathcal{E}}$ denote the complement of any event $\mathcal{E}$. For any fixed $m, i \in A_{m-1}$ and $a \in (0,1)$,*

$$\mathbb{P}\left[\overline{\mathcal{E}_{m,i}^{(\text{U})}(a)}\right] \leq 2\exp\left[-\tfrac{a^2 n_m}{3}\right],\ \mathbb{P}\left[\overline{\mathcal{E}_{m,i}^{(\text{L})}(a)}\right] \leq 2\exp\left[-\tfrac{a^2 n_m}{3}\right].$$

Note that n_m is the expected number of pulls of each active item $i \in A_{m-1}$ during phase m. Lemma 5.2 implies that we are able to bound the gap between $\hat{w}_m(i)$ and $w(i)$ for each active item $i \in A_{m-1}$ with high probability.

Technique. In light of the importance of randomization for the regret minimization problem (Lykouris et al., 2018; Gupta et al., 2019; Zimmert & Seldin, 2019), we inject randomness in PSS(u) and derive Lemma 5.2, which explains the necessity of Line 6 in Algorithm 1 in order to mitigate the impact of adversarial corruptions. While an active item is pulled probabilistically in PSS(u), it is pulled for a *fixed* number of times in SH. Though the expected number of pulls of one active item is of the same order for PSS(2) and SH, the absence of randomization in SH does not allow Theorem B.2 to bound the gap between $\hat{w}_m(i)$ and $w(i)$ in the same way as for PSS(2). For SH, we can only show that

$$\mathbb{P}\left[\hat{w}_m(i) < w(i) + \frac{C_m|A_{m-1}|}{N} + a\right] \leq \exp\left[-\frac{a^2 \cdot n_m}{3}\right],$$

and similarly for the upper tail. Disregarding constants, the difference between these bounds and those for PSS(2) in Lemma 5.2 is that the term involving C_m is worse by a factor of $|A_{m-1}|$ for SH. As a result, the bound on $\Delta_{1,i_{\text{out}}}$ for SH turns out to be $O(CL\log_2 L/T)$, which is worse than that for PSS(2) by a factor of L (see Table 1). A similar explanation is also applicable to explain the difference between the bounds for UP and PSS(L).

Elimination of the optimal item. When the agent fails to output item 1 (the optimal item), i.e.,$1 \neq i_{\text{out}}$, item 1 is inactive by the end of the last phase of the algorithm. Let $m_1 := \min\{m \in [M] : 1 \notin A_m\}$, where $\min\emptyset = \infty$. Since $1 \neq i_{\text{out}}$, we have $m_1 \leq M$. The index m_1 labels the phase during which item 1 turns from active to inactive. Next, any item i that belongs to the active set A_{m_1} satisfies $w(i) < w(1)$ and $\hat{w}_{m_1}(i) \geq \hat{w}_{m_1}(1)$. Conditioning on $\mathcal{E}_{m_1,1}^{(\text{L})}(a)$ and $\mathcal{E}_{m_1,i}^{(\text{U})}(a)$, we have

$$(-\infty, w(i) + \tfrac{2C_m}{N} + 2a] \cap [w(1) - \tfrac{2C_m}{N} - 2a, +\infty) \neq \emptyset.$$

To facilitate our analysis, we set $a_i := \Delta_{1,i}/8$ for all $2 \le i \le L$. We let j_1 be the item in A_{m_1} with the smallest mean reward, i.e., $j_1 := \arg\min_{i \in A_{m_1}} w(i)$. Lemma 5.2 implies that with probability $1 - 4\exp(-\Delta_{1,j_1}^2 \cdot n_{m_1}/192)$, we have $\Delta_{1,j_1} \le 8C_{m_1}/N$. Since $i_{\text{out}} \in A_{m_1}$, we have $w(j_1) \le w(i_{\text{out}})$. This allows us to bound $\Delta_{1,i_{\text{out}}}$ as follows:

$$\Delta_{1,i_{\text{out}}} \le \Delta_{1,j_1} \le \frac{8C_{m_1}}{N} \le \frac{8C}{N}.$$

Note that m_1, j_1 are random variables that depend on the dynamics of the algorithm. For any realization of m_1, j_1, we formulate the observation above in Lemma 5.3. The complete proof of Lemma 5.3 is postponed to Section C.3.

Lemma 5.3. *Conditioned on $\mathcal{E}^{(\mathrm{L})}_{m,1}(a_i)$ and $\mathcal{E}^{(\mathrm{U})}_{m,i}(a_i)$, where $a_i = \Delta_{1,i}/8$ for each $2 \le i \le L$, we have*

$$\{1 \in A_{m-1}, 1 \notin A_m, i \in A_m\} \subset \left\{\Delta_{1,i} \le \frac{8C_m}{N}\right\}.$$

Bounds. When $\mathcal{E}^{(\mathrm{L})}_{m_1,1}(a_{j_1})$ and $\mathcal{E}^{(\mathrm{U})}_{m_1,j_1}(a_{j_1})$ hold, we can apply Lemma 5.3 to bound $\Delta_{1,i_{\text{out}}}$ with the total corruption budget C, i.e., for any realization of m_1, j_1,

$$\mathbb{P}\left[\Delta_{1,i_{\text{out}}} > \frac{8C}{N}\right] \le \mathbb{P}\left[\overline{\mathcal{E}^{(\mathrm{L})}_{m_1,1}(a_{j_1}) \bigcap \mathcal{E}^{(\mathrm{U})}_{m_1,j_1}(a_{j_1})}\right].$$

In addition, the definitions of j_1 and A_m indicate that

$$j_1 \ge |A_{m_1}| = \left\lceil \frac{L}{u^{m_1}} \right\rceil \ge \frac{\left\lceil \frac{L}{u^{m_1-1}} \right\rceil}{u} = \frac{|A_{m_1-1}|}{u},$$

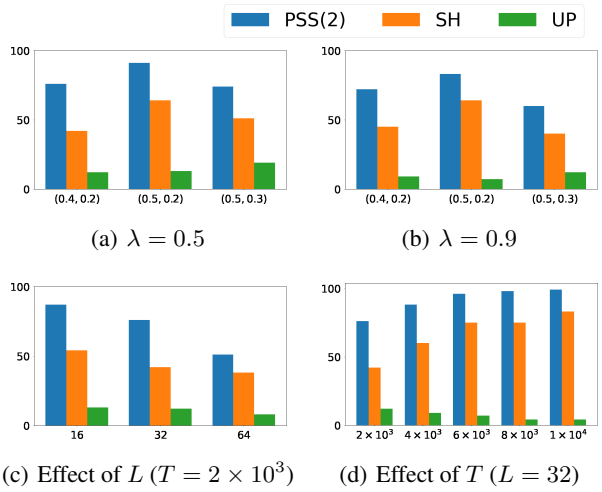
and $|A_{m_1-1}| \le L$. These inequalities, along with Lemma 5.2, the definitions of a_i, N_m and $\tilde{H}_2(w, L, u)$, imply that for all $1 \le m \le M$ and $2 \le i \le L$,

$$\mathbb{P}\left[\left(\overline{\mathcal{E}^{(\mathrm{L})}_{m_1,1}(a_{j_1}) \bigcap \mathcal{E}^{(\mathrm{U})}_{m_1,j_1}(a_{j_1})}\right) \bigcap \{m_1 = m, j_1 = i\}\right] \le 4\exp\left[-\frac{N}{192\tilde{H}_2(w, L, u)}\right].$$

Altogether,

$$\begin{aligned}
&\mathbb{P}\left[\Delta_{1,i_{\text{out}}} > \frac{8C}{N}\right] \\
&\le \sum_{m=1}^{M}\sum_{i=2}^{L} \mathbb{P}\left[\left\{\Delta_{1,i_{\text{out}}} > \frac{8C}{N}\right\} \bigcap \{m_1 = m, j_1 = i\}\right] \\
&\le 4M(L-1)\exp\left[-\frac{N}{192\tilde{H}_2(w, L, u)}\right].
\end{aligned}$$

We complete the proof of Theorem 4.1 with $N = \lfloor T/M \rfloor$, $M = \lceil \log_u L \rceil$. We elaborate on the details in Section C.4.



(a) $\lambda = 0.5$ (b) $\lambda = 0.9$

(c) Effect of L ($T = 2 \times 10^3$) (d) Effect of T ($L = 32$)

Figure 3. Percentage of correct BAI of PSS(2), SH and UP. We fix $T = 2 \times 10^3$, $L = 32$ and vary w^*, w' in (a) and (b). We fix $\lambda = 0.5, w^* = 0.4, w' = 0.2$ in (c) and (d).

6. Numerical Experiments and Conclusion

We compare the performances of PSS(2), SH and UP under the corruption strategy considered in Theorem 4.2. In the experiments, we set the mean of the optimal item to be $w^* \in \{0.4, 0.5\}$, the mean of $L-2$ suboptimal items to be $w' = 0.2$. We set $\Delta = (w^* - w')/3$ and the mean of the remaining item to be $w^* - \Delta$. The CPS $C/T = (1+\lambda)2\Delta/(L\log_2 L)$ (cf. Theorem 4.3). For each algorithm and instance, we ran 100 independent trials and report the percentage of trials each algorithm succeeds in identifying the optimal item. Further experiments are provided in Appendix D. The codes to reproduce all the experiments can be found at `https://github.com/zixinzh/2021-ICML.git`.

Overall, Figure 3 implies that PSS(2) always outperforms SH and UP for a BAI problem that is attacked by the strategy of Theorem 4.2, underscoring the importance of randomization. Next, we observe from Figures 3(a)-3(b) that a larger Δ means that the difference between the optimal and suboptimal items is more pronounced, resulting in better performances across all algorithms, even if the CPS increases. Since the CPS increases with λ, each algorithm identifies the best item less often when λ increases (see Table A.1). Figure 3(c) shows that the agent identifies the best item less often when L increases. This implies that even if we let the CPS decrease with L (per Theorem 4.3), the larger size of the ground set still makes the instance more difficult. Lastly, Figure 3(d) shows that when the CPS is fixed, a larger T increases the success probabilities of PSS(2) and SH.

Summary and Future Work. This paper has deepened our understanding the fundamental performance limits of BAI algorithms in their ability to cope with adversarial corruptions that are added on to the random rewards. We

designed PSS(u), an algorithm that can be regarded as a robustification of the SH algorithm by Karnin et al. (2013). Due to PSS(u)'s inherent randomized nature, it can successfully mitigate the adversarial corruptions. Furthermore, we showed by way of constructing several adversarial corruption strategies that the optimality gap of PSS(u) is $O(\log L)$-competitive vis-à-vis *any* corruption-tolerant algorithm. These attack strategies are shown to break SH but PSS(u) remains robust to the corruptions.

Inspired by Liu & Shroff (2019), Jun et al. (2018), and Zuo (2020), it would be fruitful to devise optimal corruption strategies for algorithm-specific and algorithm-independent settings to uncover whether the dependence of the smallest optimality gap ϵ_C on $\log L$ is fundamental. We conjecture that the smallest ϵ_C does not depend on L. More ambitiously, we would like to close the gap between the upper and lower bounds in (4.3) and (4.6).

Acknowledgements

This work is partially funded by a National University of Singapore Start-Up Grant (R-266-000-136-133), a Singapore National Research Foundation (NRF) Fellowship (R-263-000-D02-281) and a Singapore Ministry of Education AcRF Tier 1 Grant (R-263-000-E80-114).

References

Altschuler, J., Brunel, V.-E., and Malek, A. Best arm identification for contaminated bandits. *Journal of Machine Learning Research*, 20(91):1–39, 2019.

Audibert, J.-Y. and Bubeck, S. Best arm identification in multi-armed bandits. In *Proceedings of the 23th Conference on Learning Theory*, pp. 41–53, 2010.

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

Beygelzimer, A., Langford, J., Li, L., Reyzin, L., and Schapire, R. Contextual bandit algorithms with supervised learning guarantees. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, pp. 19–26, 2011.

Bogunovic, I., Krause, A., and Scarlett, J. Corruption-tolerant Gaussian process bandit optimization. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, pp. 1071–1081, 2020.

Carpentier, A. and Locatelli, A. Tight (lower) bounds for the fixed budget best arm identification bandit problem. In *Proceedings of the 29th Conference on Learning Theory*, pp. 590–604, 2016.

Chen, S., Lin, T., King, I., Lyu, M. R., and Chen, W. Combinatorial pure exploration of multi-armed bandits. In *Proceedings of the 27th Advances in Neural Information Processing Systems*, pp. 379–387. 2014.

Chen, W., Wang, Y., Yuan, Y., and Wang, Q. Combinatorial multi-armed bandit and its extension to probabilistically triggered arms. *Journal of Machine Learning Research*, 17(1):1746–1778, 2016.

Dubhashi, D. P. and Panconesi, A. *Concentration of measure for the analysis of randomized algorithms*. Cambridge University Press, 2009.

Gupta, A., Koren, T., and Talwar, K. Better algorithms for stochastic bandits with adversarial corruptions. In *Proceedings of the 32nd Conference on Learning Theory*, pp. 1562–1578, 2019.

Jun, K.-S., Jamieson, K. G., Nowak, R. D., and Zhu, X. Top arm identification in multi-armed bandits with batch arm pulls. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pp. 139–148, 2016.

Jun, K.-S., Li, L., Ma, Y., and Zhu, J. Adversarial attacks on stochastic bandits. In *Proceedings of the 31st Advances in Neural Information Processing Systems*, pp. 3640–3649, 2018.

Karnin, Z., Koren, T., and Somekh, O. Almost optimal exploration in multi-armed bandits. In *Proceedings of the 13th International Conference on Machine Learning*, pp. 1238–1246, 2013.

Krishnamurthy, A., Lykouris, T., and Podimata, C. Corrupted multidimensional binary search: Learning in the presence of irrational agents. *arXiv preprint arXiv:2002.11650*, 2020.

Lattimore, T. and Szepesvári, C. *Bandit algorithms*. Cambridge University Press, 2020.

Li, Y., Lou, E. Y., and Shan, L. Stochastic linear optimization with adversarial corruption. *arXiv preprint arXiv:1909.02109*, 2019.

Liu, F. and Shroff, N. Data poisoning attacks on stochastic bandits. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 4042–4050, 2019.

Liu, G. and Lai, L. Action-manipulation attacks on stochastic bandits. In *Proceedings of the 45th International Conference on Acoustics, Speech and Signal Processing*, pp. 3112–3116, 2020.

Lykouris, T., Mirrokni, V., and Leme, R. P. Stochastic bandits robust to adversarial corruptions. In *STOC 2018: Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pp. 114–122, 2018.

Lykouris, T., Simchowitz, M., Slivkins, A., and Sun, W. Corruption robust exploration in episodic reinforcement learning. *arXiv preprint arXiv:1911.08689*, 2020.

Mitzenmacher, M. and Upfal, E. *Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis*. Cambridge University Press, 2017.

Rejwan, I. and Mansour, Y. Top-k combinatorial bandits with full-bandit feedback. In *Proceedings of the 31st International Conference on Algorithmic Learning Theory*, pp. 752–776, 2020.

Shen, C. Universal best arm identification. *IEEE Transactions on Signal Processing*, 67(17):4464–4478, 2019.

Zhong, Z., Cheung, W. C., and Tan, V. Y. F. Best arm identification for cascading bandits in the fixed confidence setting. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Zimmert, J. and Seldin, Y. An optimal algorithm for stochastic and adversarial bandits. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*, pp. 467–475, 2019.

Zuo, S. Near optimal adversarial attack on UCB bandits. *arXiv preprint arXiv:2008.09312*, 2020.