UNIVERSITÄT MANNHEIM

Variance-aware Algorithms for Stochastic Bandit Problems

Development, Performance Analysis and Implementation

submitted by

Elise Marie Wolf

born 11th April 2003

Bachelor Thesis

School of Business Informatics and Mathematics
Chair of Probability Theory
University of Mannheim
Supervised by Prof. Dr. Leif Döring

Mannheim
23rd July 2024

Declaration of Authorship

I hereby certify that this thesis was written by me personally and that I have not made use of any unauthorized assistance from third parties. I also affirm that this thesis or parts thereof have not been submitted by myself as proof of examination elsewhere. Verbatim or analogous copies from other writings and publications in printed or electronic form are marked. All secondary literature and other sources are acknowledged and listed in the bibliography. The same applies to graphic representations and images as well as to all internet sources. I also agree that my work may be sent and stored anonymously in electronic form for the purpose of a plagiarism check. I am aware that the work may not be corrected if this declaration is not given.

Place, Date Signature

Acknowledgements

I would like to thank my supervisor Professor Dr. Leif Döring for his guidance and support throughout this project. His availability and quick responses to my questions were greatly appreciated.

I also want to acknowledge Benedikt Wille for his helpful advice on key issues. His input was valuable in addressing the challenges I encountered.

Contents

1 Introduction

For nearly a century, multi-armed bandits have been studied as fundamental components of decision-making problems in Reinforcement Learning. These problems involve balancing the exploration-exploitation trade-off to optimize decision-making in uncertain environments. The multi-armed bandit (MAB) problem consists of making a sequence of choices among different options, known as arms, to maximize the cumulated reward over time while balancing the exploration of new, potentially better arms, and the exploitation of known ones. MABs find various applications, including online advertising and clinical trials [5].

One of the most influential algorithms for addressing this problem is the Upper Confidence Bound (UCB) algorithm. The UCB algorithm balances exploration and exploitation by considering the uncertainty of rewards. In many practical situations, the variability in the rewards can significantly impact performance. This has led to the development of variance-aware algorithms, which aim to improve decision-making by incorporating variance estimates of the rewards. Despite extensive studies on MABs, there has been no comprehensive comparison of variance-aware algorithms and those that do not incorporate variance estimates. Therefore, this thesis is intended to close this gap by examining how variance-aware bandit algorithms compare to non-variance-aware bandit algorithms in terms of performance, and what conclusions can be drawn regarding their exploration-exploitation trade-offs.

To establish a basis for comparison, we first examine the theoretical foundations of MAB algorithms. In Chapter 2, we introduce the basics of stochastic bandit problems as well as the development of the UCB algorithm, upon which the variance-aware algorithms are based.

Chapter 3 derives various variance-aware algorithms based on the UCB framework. The main focus of the variance-aware algorithms we explore is UCB-V. Additionally, we discuss algorithms initially introduced alongside the UCB algorithm. Finally, we establish the groundwork for deriving another variance-aware algorithm, EUCBV, which introduces a fundamentally different elimination process for arms.

Transitioning from theoretical discussions to empirical analysis, Chapter 4 employs simulation techniques to investigate the performance of the introduced algorithms. We identify a gap in the current literature, noting the absence of a comprehensive empirical comparison between variance-aware and non-variance-aware algorithms. We describe the simulation techniques used to evaluate these algorithms and design an interactive tool, providing readers with the means to analyze the algorithms themselves. This chapter concludes with a performance comparison of the algorithms.

By integrating theoretical background with interactive simulations for empirical analysis, this thesis contributes to a comprehensive understanding of the exploration-exploitation dynamics and performance differences between variance-aware algorithms and those which do not consider variance estimates.

2 UCB Algorithm and its Basics

In this chapter, we introduce the theoretical framework for MAB problems. We begin by building an understanding of MABs and systematically trace the development of algorithms to optimize decision-making for these problems, moving from basic to more advanced forms. The chapter concludes with an examination of the UCB algorithm, which serves as the foundation for the variance-aware algorithms and subsequent performance evaluation discussed in the following chapters.

2.1 Introduction to Multi-Armed Bandit Problems

The first section of this chapter aims to motivate the question of the thesis, how the consideration of variance affects the trade-off between exploration and exploitation. In order to answer this question, we will study the underlying MAB model of those problems for a thorough analysis of algorithmic behaviour and outcome generation. The following introduction is primarily based on "The Mathematics of Reinforcement Learning" by L. Döring [5] and the book "Bandit Algorithms" by T. Lattimore et al. [11]. The notations have been slightly adapted.

The theoretical foundation of RL are bandit problems, a concept introduced by William R. Thompson in 1933. These problems establish criteria to determine the optimal choice of actions [16]. The simplest bandit problem is the two-armed bandit machine. In this scenario, the learner, or so-called agent, chooses one or the other arm of the machine per round, each resulting in a random payoff that is distributed uniquely for each arm and unknown to the learner beforehand. The machine was given this name in memory of the one-armed bandit, an old-fashioned name describing a slot machine that relies on lever-operation [11].

Formally defined, these problems are settings in which different decision-making options are presented as a set of arms, also called actions $\mathcal{A} = \{a_1, \ldots, a_K\}$, which in our case are finite unless stated differently. Each arm $a_k \in \mathcal{A}$ yields a certain payoff at time step t, also called reward $X_{a_k,t} \sim \mathbb{P}_{a_k}$. The reward is unknown to the learner beforehand. Instead, the expectation of the reward can be utilized. Our objective is to maximize the expected reward over all iterations of the decision-making process up to the end of the time horizon n by selecting the best arm more frequently. The concept of bandit problems is integrated into a stochastic bandit model ν and is commonly referred to as K-armed bandit, where K denotes the number of arms. The sequence of probability distributions, that the player chooses certain arms on $\mathcal{A}$ is denoted as a learning strategy $(\pi_t)_{t=1,\ldots,n}$, only depending on what as been played prior to time t [5].

Definition 2.1.1. Bandit problems in stochastic bandit models (Compare [5], Definition 1.2.1 and [11], Chapter 4.1)

Suppose $\mathcal{A} = \{a_1, \ldots, a_K\}$ and $\nu = \{\mathbb{P}_{a_k}\}_{a_k \in \mathcal{A}}$ is a family of real-valued distributions.

- The set ν is called a stochastic bandit model and K is the number of arms.

- The action value, also referred to as the Q-value, of an arm a_k is defined by the expectation of the reward:

  $$Q_{a_k} := \mathbb{E}[X_{a_k}] = \int x \, \mathbb{P}_{a_k}(dx).$$

  We simplify the notation of $X_{a_k,t}$ to X_{a_k}, assuming that $(X_{a_k,t})_{t=1,\dots,n}$ are drawn from the same distribution $\mathbb{P}_{a_k}$ across time steps. A best arm, usually denoted by a^*, is defined as the arm with the highest action value:

  $$a^* = \arg\max_{a_k \in \mathcal{A}} Q_{a_k}.$$

  Typically, one abbreviates Q^* for the largest action value Q_{a^*}, and if there are several optimal arms, the arg max chooses any of them.

- A learning strategy (or policy) for n rounds ($n = +\infty$ is allowed, but will not be the focus of this thesis) consists of

  – an initial distribution π_1 on $\mathcal{A}$,

  – a sequence $(\pi_t)_{t=2,\dots,n}$ of kernels on $\Omega_{t-1} \times \mathcal{A}$, where for time step t, π_t is a probability kernel from $\Omega_{t-1} \times \mathcal{A}$ to $(\mathcal{A}, \mathcal{P}(\mathcal{A}))$ such that

    (i) $y \mapsto \pi_t(y, A)$ is $(\mathcal{F}_{t-1}, \mathcal{B}(\overline{\mathbb{R}}))$-measurable for all $A \in \mathcal{A}$

    (ii) $A \mapsto \pi_t(y, A)$ is a probability measure for all $y \in \Omega_t$.

    Ω_t denotes all trajectories $(a_1, x_1, a_2, x_2, \dots, a_t, x_t) \in (\mathcal{A} \times \mathbb{R})^t$ and $\mathcal{F}_t = \mathcal{B}(\Omega_t)$. We will write the kernels in reverse ordering of the arguments

    $$\pi_t(\cdot; a_1, x_1, a_2, x_2, \dots, a_t, x_t)$$

    with the meaning that $\pi_t(\{a_k\}; a_1, x_1, a_2, x_2, \dots, a_t, x_t)$ is the probability arm a_k being chosen at time t if the past rounds resulted in actions and rewards $a_1, x_1, a_2, x_2, \dots, a_t, x_t$.

In the following, we discuss stochastic bandit processes that operate through a two-step mechanism. At each round $t \leq n$, a specified learning strategy π determines the action or arm A_t to be selected. Upon taking this action, we receive a reward X_t. Consequently, the process (A_t^π, X_t^π) is the sequence of actions and corresponding rewards throughout the rounds.

Definition 2.1.2. Stochastic bandit process (Compare [5], Definition 1.2.3 and [11], Definition 4.7)
Suppose ν is a stochastic bandit model and $(\pi_t)_{t=1,\dots,n}$ is a learning strategy for n rounds. Then a stochastic process $(A_t^\pi, X_t^\pi)_{t=1,\dots,n}$ on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$ is called a stochastic bandit process with learning strategy π if $A_1^\pi \sim \pi_1$ and:

(i) $\mathbb{P}(A_t^\pi = a_k \mid A_1^\pi, X_1^\pi, \dots, A_{t-1}^\pi, X_{t-1}^\pi) = \pi_t(\{a_k\}; A_1^\pi, X_1^\pi, \dots, A_{t-1}^\pi, X_{t-1}^\pi)$,

(ii) $\mathbb{P}(X_t^\pi \in \cdot \mid A_1^\pi, X_1^\pi, \ldots, A_t^\pi) \sim \mathbb{P}_{A_t^\pi}$,

where $\mathbb{P}_{A_t^\pi}$ is the (random) distribution $\sum_{a_k \in \mathcal{A}} \mathbb{P}_{a_k}(\cdot)\mathbb{1}_{A_t^\pi = a_k}$.We will call A_t^π the action, which is the chosen arm at time t and X_t^π the reward when playing arm A_t^π at time t. The superscript π will be dropped if the learning strategy is clear from the context.

Theorem 2.1.3. For every stochastic bandit model ν and every learning strategy $(\pi_t)_{t=1,\ldots n}$ there exists a corresponding stochastic bandit process $(A_t^\pi, X_t^\pi)_{t=1,\ldots n}$.

Proof. We refer to the book [11], Chapter 4.6. □

Example 2.1.4. Bernoulli Bandit (Compare [5], Example 1.2.5)
Consider a Bernoulli bandit with ten arms, where the probability of receiving a reward of 1 for each arm is given by:

$$p \in \{0.9, 0.85, 0.8, 0.5, 0.1, 0.88, 0.7, 0.3, 0.88, 0.75\}.$$

Each probability p_i represents the likelihood of receiving a reward of 1 for the corresponding arm. For instance, with $p = 0.9$ the first arm provides a reward of 1 and a reward of 0 with probability 0.1. For a 10-armed Bernoulli bandit with iid random variables the p_i are equal for all ten arms $X_{a_i}, i = 1, ..., 10$.

Remark 2.1.5. Defining a bandit problem comes with the significant challenge, that the learner has limited information about the optimal next action. As a result, the learner must choose from a set of different stochastic bandit models ν, that we denote as an environment class $\mathcal{E}$. The actual environment or stochastic bandit model selected is then $\nu \in \mathcal{E}$. Thus, the environment can be viewed as an element within the environment class $\mathcal{E}$, from which a single environment ν is drawn [10].

As mentioned before, our main objective is to maximize the cumulated reward over all time steps n by finding an algorithm that generates a best learning strategy $(\pi_t)_{t=1,\ldots,n}$ such that the expected cumulated reward $\mathbb{E}[\sum_{t=1}^n X_t] = \sum_{t=1}^n \mathbb{E}[X_t]$ is maximized. If the best arm a^* would be known in advance, the learner would always choose it to maximize the reward. In this case, the reward at time n would be nQ^*, which is also the upper bound for the reward until time n. To assess the effectiveness of the algorithm based on its chosen actions, we can calculate the regret, which is the difference between playing the optimal actions and the actions actually chosen at each time step. The optimization problem of maximizing the expected cumulated reward is equivalent to minimizing the regret, accordingly, the stochastic bandit problem consists of finding learning strategies that minimize the regret [5].

Definition 2.1.6. Regret (Compare [5], Definition 1.2.6)
Suppose ν is a bandit model and $(\pi_t)_{t=1,\ldots,n}$ a learning strategy. Then the (cumulated) regret is defined by

$$R_n(\pi) := nQ^* - \mathbb{E}\left[\sum_{t=1}^n X_t^\pi\right].$$

For simplicity, we drop the superscript π if the learning strategy is clear from context.

Definition 2.1.7. Reward gap (Compare [5], Definition 1.2.7)
The differences $\Delta_{a_k} := Q^* - Q_{a_k}$ are called reward gaps of arm a_k.

Remark 2.1.8. The regret bounds can be divided into model-based and model-independent bounds. Model-based bounds depend on the underlying bandit model ν, i.e. take into account the properties of the bandit model and provide regret bounds tailored to the specific problem. In comparison, model-independent bounds are not based on the bandit model ν. Instead, they focus on properties that apply across a class of bandit models and typically only depend on the time horizon n [15].

Achieving a regret of zero necessitates that the agent must know the best action a^* and decides to chose only that action. In contrast, when arms are chosen uniformly at random, the regret increases linearly with n and can be expressed as

$$R_n(\pi) := nQ^* - \mathbb{E}\left[\sum_{t=1}^{n} X_t\right] = n\left(Q^* - \frac{1}{K}\sum_{k=1}^{K} Q_{a_k}\right).$$

Here $Q^* - \frac{1}{K}\sum_{k=1}^{K} Q_{a_k}$ represents the average regret per round. Since the difference does not decrease for uniformly chosen arms, a strategy without learning will always have a linear regret. Consequently, strategies with sublinear regret are reasonable [5]. However, given that only $\nu \in \mathcal{E}$ is known, a more attainable objective is to identify a policy π with sublinear regret, meaning that

$$\lim_{n\to\infty} \frac{R_n(\pi,\nu)}{n} = 0, \quad \forall \nu \in \mathcal{E}.$$

For a policy with sublinear regret, the learner chooses the best action a^* almost always as the time horizon n tends to infinity [10].

Definition 2.1.9. Number of draws of an arm
Suppose ν is a bandit model and $(\pi_t)_{t=1,\ldots,n}$ a learning strategy. Define

$$T^{\pi}_{a_k}(n) := \sum_{t=1}^{n} \mathbb{1}_{A^{\pi}_t = a_k}$$

as the number of times, arm a_k has been played until time step n under the learning strategy $(\pi_t)_{t=1,\ldots n}$.

The regret decomposition lemma is a commonly used tool in calculating the regret of a given algorithm.

Lemma 2.1.10. Regret decomposition Lemma (Compare [5], Lemma 1.2.8)
Defining $T^{\pi}_{a_k}(n)$ as in Definition 2.1.9, the following decomposition holds:

$$R_n(\pi) = \sum_{a_k \in \mathcal{A}} \Delta_{a_k} \mathbb{E}[T^{\pi}_{a_k}(n)].$$

Proof. The proof involves using a clever 1 and the tower property. For each fixed $t \leq n$ holds $\sum_{a_k \in \mathcal{A}} \mathbb{P}(A_t = a_k) = 1$. Then the regret $R_n(\pi)$ results as

$$R_n(\pi) = nQ^* - \mathbb{E}\left[\sum_{t=1}^{n} X_t\right] = \sum_{t=1}^{n} \mathbb{E}[Q^* - X_t] = \sum_{t=1}^{n} \sum_{a_k \in \mathcal{A}} \mathbb{E}[(Q^* - X_t)\mathbb{1}_{A_t = a_k}].$$

Suppose a fixed round t. The expected return conditioned on A_t, is Q_{A_t}. Then for each action $A_t = a_k$ holds $Q_{A_t} = Q_{a_k}$, such that

$$\begin{aligned}
\mathbb{E}[(Q^* - X_t)\mathbb{1}_{A_t = a_k} \mid A_1, X_1, \ldots, A_t] &= \mathbb{1}_{A_t = a_k}\mathbb{E}[Q^* - X_t \mid A_1, X_1, \ldots, A_t] \\
&= \mathbb{1}_{A_t = a_k}(Q^* - Q_{A_t}) \\
&= \mathbb{1}_{A_t = a_k}(Q^* - Q_{a_k}) \\
&= \mathbb{1}_{A_t = a_k}\Delta_{a_k}.
\end{aligned}$$

By applying the tower rule and linearity of the expectation, the combination of both calculations results in

$$R_n(\pi) = \sum_{t=1}^{n} \sum_{a_k \in \mathcal{A}} \mathbb{E}[\mathbb{E}[(Q^* - X_t)\mathbb{1}_{A_t = a_k} \mid A_1, X_1, \ldots, A_t]] = \sum_{a_k \in \mathcal{A}} \Delta_{a_k} \mathbb{E}\left[\sum_{t=1}^{n} \mathbb{1}_{A_t = a_k}\right].$$

Using the definition of $T_{a_k}(n)$, we obtain the desired statement. □

For the evaluation of the algorithms, let us establish the additional notation of the estimated action value to facilitate our analysis.

Definition 2.1.11. Estimated action value (Compare [5], Definition 1.3.1)
If (A_t, X_t) is a stochastic bandit process for some learning strategy π, then we define

$$\hat{Q}_{a_k}(t) := \frac{\sum_{i=1}^{t} X_i \mathbb{1}_{A_i = a_k}}{T_{a_k}(t)}, \quad a_k \in \mathcal{A},$$

and call $\hat{Q}_{a_k}(t)$ the estimated action value, which is the average return from playing arm a_k up to time t.

Remark 2.1.12. When choosing $A_i = a_k$ at time i we can rewrite $X_i \mathbb{1}_{A_i = a_k} = X_{a_k, i}$, allowing us to write the estimated action value of Definition 2.1.11 as

$$\hat{Q}_{a_k}(t) = \frac{\sum_{i=1}^{t} X_{a_k, i}}{T_{a_k}(t)}, \quad a_k \in \mathcal{A},$$

We employ the estimated action value to determine which arm to play in the next round, as it allows us to estimate the true action value Q using $\hat{Q}_{a_k}$. By the Law of Large Numbers for independent and identically distributed (iid) samples, $\hat{Q}$ converges almost surely towards the actual value Q. This method is referred to as the sample-average method [15].

2.2 Development of the UCB Algorithm

To understand the components of the UCB algorithm, which forms the basis for variance-aware adaptations and is discussed in the following section, we first examine the most basic learning strategy, the Explore-Then-Commit (ETC) algorithm. Our examination on the ETC algorithm highlights the concepts of exploration and exploitation. However, ETC has limitations due to its lack of justification in both derivation and functionality. We then introduce the purely Greedy algorithm, which focuses on exploiting known optimal actions and primarily serves to illustrate the advanced principles behind the UCB algorithm.

The ETC algorithm is based on the simple idea that it is best to compare all available options before committing to a single one. First, ETC evaluates all possible actions to determine the most promising one and then assigns the remaining choices to the arm that is expected to generate the highest rewards based on the initial testing phase. Thus, the estimated action value in Definition 2.1.11 is derived from the rewards obtained by each arm during this testing phase. The basic ETC algorithm with m rounds of testing all arms K is stated in Algorithm 1. Its formulation is based on [5], Algorithm 1.

Algorithm 1 Explore-Then-Commit Algorithm

```
Data: Bandit model $\nu$, number of arms $K$, exploration rounds $m$, time horizon $n$
Result: Actions $A_1, \dots, A_n$ and rewards $X_1, \dots, X_n$
1: Initialize $\hat{Q}(0) = 0$, $t = 0$
2: while $t \leq n$ do
3:     Set $A_t = \begin{cases} a_{t \mod (K+1)} & \text{if } t \leq mK \\ \arg\max_{a_k} \hat{Q}_{a_k}(mK) & \text{if } t > mK \end{cases}$
4:     Obtain reward $X_t$ by playing arm $A_t$
5:     Update the estimated action value function $\hat{Q}_{a_k}(t)$
6:     $t \leftarrow t + 1$
```

As defined in Algorithm 1, ETC can be divided into two different parts. For time steps up to $t = mK$, the algorithm explores different arms. The concept of exploration, as described by A. Makone in [13], aims to improve the estimated action value $\hat{Q}$ by exploring different options a_k, $k \in \{1, ..., K\}$ and refining the estimated action values by the obtained reward X_{a_k} for each option a_k through repeated selection. Conversely, the ETC behaves greedily for $t > mK$. Only the arm a_k with the highest value of $\hat{Q}_{a_k}$ that appears to offer the best reward is selected. This approach is called exploitation because it prioritizes the generation of high rewards while potentially sacrificing the possibility of discovering even better rewards. Exploitation therefore leaves much of the environment unexplored [13].

Remark 2.2.1. We will compare the effectiveness of our algorithms through their upper regret bounds. An upper bound on the regret provides a limit on how large the regret can be, ensuring that the algorithm performs reasonably well in the worst-case scenario. Conversely, a lower bound on the regret indicates a minimum level of regret that no algorithm can surpass, thereby providing a baseline for evaluating the performance of any algorithm.

Before we specify the upper regret bound of the ETC algorithm, we introduce the definition of a σ-subgaussian random variable.

Definition 2.2.2. σ-subgaussian random variable (Compare [5], Definition 1.3.3)
A random variable X on a probability space $(\Omega, \mathcal{A}, \mathbb{P})$ is called σ-subgaussian for $\sigma > 0$, if $\mathcal{M}_X(\lambda) = \mathbb{E}[e^{\lambda(X-\mathbb{E}[X])}] \leq e^{\frac{\lambda^2\sigma^2}{2}}$ for all $\lambda \in \mathbb{R}$.

Remark 2.2.3. The term "subgaussian" comes from the property that for $X \sim \mathcal{N}(0, \sigma^2)$, $\mathbb{E}[e^{\lambda X}] = e^{\frac{\lambda^2\sigma^2}{2}}$, indicating that X is σ-subgaussian. It means that the tails of a σ-subgaussian random variable decay at least as fast as those of a Gaussian random variable $X \sim \mathcal{N}(0, \sigma^2)$, implying a similar tail behavior [5].

Remark 2.2.4. For scaled variables $c|X|$ it holds, that if X is σ-subgaussian, then the scaled variable is $|c|\sigma$-subgaussian. Additionally, any centered random variable bounded between a and b is $\frac{b-a}{2}$-subgaussian. For example, Bernoulli random variables are $\frac{1}{2}$-subgaussian [5].

In the following analysis, we will assume a σ-subgaussian bandit model, where all reward distributions $\mathbb{P}_{a_k}$ are σ-subgaussian. This assumption is advantageous as it often simplifies the regret analysis. By that, the following regret bound can be achieved for the ETC algorithm.

Theorem 2.2.5. Regret bound for simple ETC (Compare [5], Theorem 1.3.2)
Suppose ν is a σ-subgaussian bandit model, i.e., all $\mathbb{P}_{a_k}$ are σ-subgaussian, with K arms and $Km \leq n$ for some $n \in \mathbb{N}$, then

$$R_n(\pi) \leq \underbrace{m \sum_{a_k \in \mathcal{A}} \Delta_{a_k}}_{\text{exploration}} + \underbrace{(n - mK) \sum_{a_k \in \mathcal{A}} \Delta_{a_k} \exp\left(-\frac{m\Delta_{a_k}^2}{4\sigma^2}\right)}_{\text{exploitation}}$$

Proof. The main idea of the proof is to decompose the exploration and exploitation phase of the ETC algorithm. For details, refer to Theorem 1.3.2 in [5]. $\square$

Even though the ETC algorithm is an initial approach to finding a solution, it is far from optimal. The ETC algorithm could benefit from improvements in the derivation of its bound. For example, it is unclear, which m to select for an optimal number of exploration rounds. Additionally, if the number of exploration rounds mK exceeds the number of actions performed n, the algorithm cannot benefit from the exploitation of the arm that has been calculated as the optimal one up to this time step. Consequently, in such scenarios, ETC performs no better than random exploration [5].

Another method that utilizes the concept of greediness is illustrated in the purely Greedy algorithm. The key idea of this algorithm is to select the arm a_k that has proven to have the highest estimated action value $\hat{Q}_{a_k}(t-1)$ in the previous iterations up to $t-1$. It is obvious that the initialisation significantly influences the initial value of the estimated action value $\hat{Q}(0)$ as well as subsequent estimates, so it is unreasonable to expect to find the optimal solution with a suboptimal initialisation. Nevertheless, the purely Greedy Algorithm, which is stated in Algorithm 2 is a useful starting point for the discussion of further modifications. It was adopted from [5], Algorithm 2.

Algorithm 2 Purely Greedy Bandit Algorithm

```
Data: Bandit model $\nu$, time horizon $n$
Result: Actions $A_1, \dots, A_n$ and rewards $X_1, \dots, X_n$
1: Initialize $\hat{Q}(0)$, $t = 0$
2: while $t \leq n$ do
3:     Set $A_t = \arg\max_{a_k} \hat{Q}_{a_k}(t-1)$
4:     Obtain reward $X_t$ by playing arm $A_t$
5:     Update the estimated action value function $\hat{Q}_{a_k}(t)$
6:     $t \leftarrow t + 1$
```

Remark 2.2.6. Through the definition of the purely Greedy algorithm, it becomes evident that there is a high risk of focusing on suboptimal decisions to early due to insufficient exploration to discover actions with higher rewards. This behaviour is known as committal [5].

Remark 2.2.7. A more reasonable approach would be to incorporate random exploration alongside the purely Greedy algorithm. This additional exploration is integrated into the ε-Greedy algorithm through a random variable, which determines whether to choose a randomized action with probability ε or act greedily with the remaining probability of $(1 - \varepsilon)$. However, this approach has its drawbacks. Firstly, if ε is chosen too small, the forced exploration may be ineffective. On the other hand, if ε is chosen to large, then the exploration is forced on suboptimal arms, although the optimal arm may already be known. Secondly, it can be proven that the exploration phase only influences the lower bound and not the upper bound of regret.

Since our objective is to minimize the potential highest regret, using the ε-Greedy algorithm instead of the purely Greedy algorithm may not adequately address the main issue. Despite its drawbacks, we are still going to refer to the ε-Greedy algorithm in our further empirical analysis and simulation to make the potential flaws comprehensible.

2.3 The classic UCB Algorithm and its Optimality

In Section 2.2, we introduced the simplest and most intuitive algorithms for bandit models to optimize the reward. This section is devoted to the study of the UCB algorithm. The objective is to thoroughly understand its functionality in order to perform variance-aware adaptations on it.

In their paper "Finite-time Analysis of the Multiarmed Bandit Problem" [4], P. Auer et al. introduced the UCB algorithm under the name of the UCB1 algorithm, as well as further adaptations that become relevant in Section 3.1. We will remain referring to it under the simplified name "UCB algorithm".

Remark 2.3.1. The paper [10] by T.L. Lai and H. Robbins introduces the principle commonly known as "optimism in the face of uncertainty" [11]. This principle suggests that if an arm has been frequently chosen and consistently yielded rewards with a certain confidence, one can be optimistic that selecting it again will likely result in a similar reward, even if the outcome is technically uncertain due to sampling. In practice, this means utilizing all available data up to the latest observation to determine the next choice of action. The UCB algorithm implements this principle by assigning a UCB value to each arm, which typically overestimates the unknown true action value, resulting in sublinear regret. The overestimation for a specific arm a_k occurs only a limited number of times because choosing arm a_k provides additional data that gradually adjusts the estimated action value of arm a_k towards its true action value by the Law of Large Numbers for iid samples. Consequently, if a suboptimal arm a_k is overestimated and chosen frequently enough, its UCB will eventually converge below the UCB of the optimal arm [10].

For the determination of the next action the algorithm utilizes the following function, which we refer to as the UCB function, for $\delta > 0$ and σ-subgaussian bandit models.

$$UCB_{a_k}(t,\delta) := \begin{cases} \infty & : T_a(t) = 0, \\ \underbrace{\hat{Q}_{a_k}(t)}_{\text{greedy}} + \underbrace{\sqrt{\frac{2\sigma^2 \log(1/\delta)}{T_{a_k}(t)}}}_{\text{exploration bonus}} & : T_{a_k}(t) \neq 0, \end{cases} \tag{2.1}$$

Remark 2.3.2. In the case of $T_{a_k}(t) \neq 0$ the $UCB_{a_k}(t,\delta)$ is split into a part that acts greedy and a bonus for choosing to explore further states, making it an extension of a Greedy algorithm. The UCB function (2.1) ensures that each arm a_k is chosen at least once due to $UCB_{a_k}(t,\delta) = \infty$ for $T_{a_k}(t) = 0$.

Let us now state the UCB Algorithm 3, adapted from [5], Algorithm 4.

Algorithm 3 UCB Algorithm

```
Data: Bandit model $\nu$, $\delta \in (0,1)$, time horizon $n$
Result: Actions $A_1, \dots, A_n$ and rewards $X_1, \dots, X_n$
1: Initialize $\hat{Q}(0)$, $t = 0$
2: while $t \leq n$ do
3:     Set $A_t = \arg\max_{a_k} UCB_{a_k}(t-1,\delta)$
4:     Obtain reward $X_t$ by playing arm $A_t$
5:     Update the estimated action value function $\hat{Q}_{a_k}(t)$
6:     $t \leftarrow t + 1$
```

For the proof of the regret bound, we must first establish certain properties and inequalities, all drawn from [5].

Proposition 2.3.3. (Compare [5], Proposition 1.3.4)
If X is σ-subgaussian, then

$$\mathbb{P}(X \geq a) \leq e^{-\frac{a^2}{2\sigma^2}} \quad \text{and} \quad \mathbb{P}(|X| \geq a) \leq 2e^{-\frac{a^2}{2\sigma^2}}, \quad \forall a > 0.$$

Proof. The proof is based on a trick called the Cramér-Chernoff method. The trick uses the Markov inequality for a parametrized family of functions.

$$\mathbb{P}(X \geq a) = \mathbb{P}\left(e^{\lambda X} \geq e^{\lambda a}\right) \overset{\text{Markov Ineq.}}{\leq} \frac{\mathbb{E}[e^{\lambda X}]}{e^{\lambda a}} \leq e^{\frac{\lambda^2\sigma^2}{2}} e^{-\lambda a} = e^{\frac{\lambda^2\sigma^2}{2} - \lambda a}.$$

To find λ, that minimizes $e^{\frac{\lambda^2\sigma^2}{2} - \lambda a}$, we take the derivative with respect to λ and set it to zero.

$$\frac{d}{d\lambda}\left(\frac{\lambda^2\sigma^2}{2} - \lambda a\right) = \lambda\sigma^2 - a = 0.$$

Solving for λ gives $\lambda = \frac{a}{\sigma^2}$, which yields the smallest bound, and therefore the first claim of the proposition holds. Since the same holds for $-X$, we obtain the second claim by writing $\mathbb{P}(|X| \geq a) = \mathbb{P}(X \geq a \text{ or } X \leq -a) \leq \mathbb{P}(X \geq a) + \mathbb{P}(-X \geq a)$. □

Corollary 2.3.4. Hoeffding's inequality (Compare [5], Corollary 1.3.5)
Suppose $X_1, \ldots, X_n$ are iid random variables on a probability space $(\Omega, \mathcal{A}, \mathbb{P})$ with expectation $\mu = \mathbb{E}[X_1]$ such that X_1 is σ-subgaussian. Then

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n} X_i - \mu \geq a\right) \leq e^{-\frac{na^2}{2\sigma^2}} \quad \text{and} \quad \mathbb{P}\left(\left|\frac{1}{n}\sum_{i=1}^{n} X_i - \mu\right| \leq a\right) \leq 2e^{-\frac{na^2}{2\sigma^2}}, \quad \forall a > 0.$$

Proof. We know that each X_i is σ-subgaussian, i.e. $\mathbb{E}[e^{\lambda(X_i-\mu)}] \leq e^{\frac{\lambda^2\sigma^2}{2}}$ for all $\lambda \in \mathbb{R}$. Then Proposition 2.3.3 holds for $\lambda^* = \frac{1}{n}\lambda$ and we can rewrite

$$\begin{aligned}\mathbb{E}\left[e^{\lambda\left(\frac{1}{n}\sum_{i=1}^{n} X_i - \mu\right)}\right] &= \mathbb{E}\left[e^{\lambda\frac{1}{n}\sum_{i=1}^{n}(X_i-\mu)}\right] = \prod_{i=1}^{n}\mathbb{E}\left[e^{\lambda^*(X_i-\mu))}\right] \\ &\leq \left(e^{\frac{(\lambda^*)^2\sigma^2}{2}}\right)^n = e^{\frac{n\lambda^2\sigma^2}{2n^2}} = e^{\frac{\lambda^2(\frac{\sigma}{\sqrt{n}})^2}{2}}.\end{aligned}$$

Thus, $\frac{1}{n}\sum_{i=1}^{n} X_i - \mu$ is $\frac{\sigma}{\sqrt{n}}$-subgaussian. Applying Proposition 2.3.3 $\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n} X_i - \mu \geq a\right) = \mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n}(X_i - \mu) \geq a\right) \leq e^{-\frac{a^2}{2(\sigma^2/n)}} = e^{-\frac{na^2}{2\sigma^2}}$.
The second equation is obtained by applying the same idea of the second claim of Proposition 2.3.3. □

By applying Hoeffding's Inequality 2.3.4 as well as Proposition 2.3.3, we derive the following adaptation on Bernstein's inequality.

Corollary 2.3.5. Adaptation on Bernstein's inequality (Compare [5], Proof of Theorem 1.3.8)
For a sequence of independent 1-subgaussian random variables $(X_i)_{i=1,\dots,n}$ holds

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n} X_i \geq \mu + \sqrt{\frac{2\log(1/\delta)}{n}}\right) \leq \exp\Big(-\frac{n\left(\sqrt{\frac{2\log(1/\delta)}{n}}\right)^2}{2}\Big) = \delta \quad \text{for all} \quad \delta \in (0,1).$$

Remark 2.3.6. As derived in the adaptation of Bernstein's inequality (2.3.5), the choice of $\sqrt{\frac{2\log(1/\delta)}{T_{a_k}(t-1)}}$ in the UCB function ensures that, with probability larger than $1-\delta$, the actual action value Q is less than the estimated one. Therefore, we can interpret the exploration bonus of $\text{UCB}_{a_k}(t-1,\delta)$ as a confidence interval, ensuring that a 1-subgaussian random variable X takes values in $(-\sqrt{2\log(1/\delta)}, \sqrt{2\log(1/\delta)})$ [11].

Theorem 2.3.7. Regret bound of the UCB algorithm (Compare [5], Theorem 1.3.8)
Suppose ν is a bandit model with 1-subgaussian arms, $n \in \mathbb{N}$, and $\delta = \frac{1}{n^2}$. Then the UCB algorithm has the model-based upper regret bound

$$R_n(\pi) \leq 3\sum_{a_k \in \mathcal{A}} \Delta_{a_k} + 16\log(n) \sum_{a_k:\Delta_{a_k}>0} \frac{1}{\Delta_{a_k}}. \tag{2.2}$$

Proof. For the proof, we will simulate the bandit process (A_t, X_t) through constructing an array $(X_t^{(a_k)})_{t\leq n, a_k\in\mathcal{A}}$ of independent random variables with each $X_t^{(a_k)}$ being drawn according to the distribution $\mathbb{P}_{a_k}$. For a given arm $a_k \in \mathcal{A}$, $X_t^{(a_k)}$ represents the reward when arm a_k is pulled at time t. We can also write it equivalently as $X_{a_k,t}$. One way of expressing the bandit process X_t is then as $X_{T_{A_t}(t)}^{(A_t)}$. We define for each arm a_k at time s

$$\overline{Q}_s^{(a_k)} = \frac{1}{s}\sum_{i\leq s} X_i^{(a_k)}.$$

This is simply the average of the observed rewards for an arm $a_k \in \mathcal{A}$ after s pulls.
We will assume without loss of generality that $a^* = a_1 \in \mathcal{A}$ and estimate the expected times a suboptimal arm $a_k \in \{a_2, ..., a_K\} = \mathcal{A}\setminus\{a_1\}$ will be played. Combined with the regret decomposition lemma, we will prove the theorem by bounding $\mathbb{E}[T_{a_k}(n)]$ for each suboptimal arm a_k. To analyze the number of pulls for a suboptimal arm a_k, we study the event $G_{a_k} = G_1 \cap G_{a_k,2}$, where

$$G_1 = \left\{\omega : Q_{a_1} < \min_{t\leq n} UCB_{a_1}(t,\delta)(\omega)\right\} \quad \text{and}$$
$$G_{a_k,2} = \left\{\omega : \overline{Q}_{u_{a_k}}^{(a_k)}(\omega) + \sqrt{\frac{2\log(1/\delta)}{u_{a_k}}} < Q_{a_1}\right\}.$$

Allow $u_{a_k} \in \mathbb{N}$ to be specified later. Here, event G_1 states that the true action value of the optimal arm a_1, Q_{a_1}, is always less than the estimation of the lowest upper confidence bound of the optimal arm through the function $UCB_{a_1}(t, \delta)$ for all $t \leq n$. This ensures that the UCB function always covers the true action value of a_1. Event $G_{a_k,2}$ states that the estimated action value of the suboptimal arm a_k after u_{a_k} pulls, plus a confidence interval, is always less than the true action value of the optimal arm a_1. This means that even accounting for uncertainty, the estimated value of a_k is still below the true value of a_1. The idea behind these events is that if both events G_1 and $G_{a_k,2}$ occur, then the suboptimal arm a_k cannot be pulled more than u_{a_k} times. This is because the UCB value of the suboptimal arm a_k will never exceed the UCB value of the optimal arm a_1, thus a_k will not be selected.

We show that this holds by assuming the contrary. Suppose $\omega \in G_{a_k}$ (i.e., G_1 and $G_{a_k,2}$ occur), but $T_{a_k}(n)(\omega) > u_{a_k}$. This means there is some time step $t \leq n$ such that $T_{a_k}(t-1)(\omega) = u_{a_k}$ and $A_t(\omega) = a_k$. Then, if a_k is chosen, the UCB value of a_k must have been the highest, i.e. $UCB_{a_k}(t-1, \delta)(\omega) > UCB_{a_1}(t-1, \delta)(\omega)$.

However, applying the definitions of G_1, $G_{a_k,2}$, and UCB yields

$$\begin{aligned} UCB_{a_k}(t-1,\delta)(\omega) &\overset{\text{Def UCB}}{=} \hat{Q}_{a_k}(t-1)(\omega) + \sqrt{\frac{2\log(1/\delta)}{T_{a_k}(t-1)(\omega)}} \\ &\overset{\text{Def } X^{(A_t)}_{T_{a_k}(t-1)}}{=} \overline{Q}^{(a_k)}_{u_{a_k}}(\omega) + \sqrt{\frac{2\log(1/\delta)}{u_{a_k}}} \\ &\overset{G_{a_k,2}}{<} Q_{a_1} \overset{G_1}{<} UCB_{a_1}(t-1,\delta)(\omega), \end{aligned}$$

Combining these inequalities leads to a contradiction. Hence, $T_{a_k}(n)(\omega) \leq u_{a_k}$ if both events G_1 and $G_{a_k,2}$ occur and therefore the number of pulls is bounded by u_{a_k}.

Since $T_{a_k}(n)$ is trivially bounded by n, the following estimate holds due to linearity:

$$\mathbb{E}[T_{a_k}(n)] = \mathbb{E}\left[T_{a_k}(n)\mathbb{1}_{G_a}\right] + \mathbb{E}\left[T_{a_k}(n)\mathbb{1}_{G_a^c}\right] \leq u_{a_k} + n\left(\mathbb{P}(G_1^c) + \mathbb{P}(G_{a_k,2}^c)\right). \tag{2.3}$$

We now estimate the probability that the events G_1 and $G_{a_k,2}$ do not occur in order to bound the expected number of pulls of the suboptimal arm a_k. Essentially, we need to estimate both probabilities separately. First, consider the probability of G_1^c:

$$\begin{aligned} \mathbb{P}(G_1^c) &= \mathbb{P}\left(Q_{a_1} \geq UCB_{a_1}(t,\delta) \text{ for some } t \leq n\right) \\ &\leq \sum_{t\leq n} \mathbb{P}\left(Q_{a_1} - \sqrt{\frac{2\log(1/\delta)}{t}} \geq \hat{Q}_{a_1}(t)\right) \\ &\leq \sum_{t \leq n} \delta = n\delta. \end{aligned}$$

This follows because for each time t, the probability that the true action value Q_{a_1} exceeds the UCB is at most δ due to Hoeffding's Inequality (2.3.4). Next, we need to choose u_{a_k}, which so far has been unspecified. We will choose u_{a_k} large enough such that through rearrangement:

$$\Delta_{a_k} - \sqrt{\frac{2\log(1/\delta)}{u_{a_k}}} \geq \frac{1}{2}\Delta_{a_k} \quad \Leftrightarrow \quad \sqrt{\frac{2\log(1/\delta)}{u_{a_k}}} \leq \frac{1}{2}\Delta_{a_k}$$
$$\Leftrightarrow \quad u_{a_k} \geq \frac{2\log(1/\delta)}{\frac{1}{4}\Delta_{a_k}^2} = \frac{8\log(1/\delta)}{\Delta_{a_k}^2}.$$

To ensure u_{a_k} is an integer and sufficiently large, we set

$$u_{a_k} = \left\lceil \frac{8\log(1/\delta)}{\Delta_{a_k}^2} \right\rceil + 1 \overset{\delta=\frac{1}{n^2}}{=} 1 + \left\lceil \frac{8\log(n^2)}{\Delta_{a_k}^2} \right\rceil = 1 + \left\lceil \frac{16\log(n)}{\Delta_{a_k}^2} \right\rceil. \tag{2.4}$$

With this choice of u_{a_k}, the probability that the event $G_{a_k,2}$ does not occur can be made sufficiently small.

$$\begin{aligned}
\mathbb{P}(G_{a_k,2}^c) &= \mathbb{P}\left(\overline{Q}_{u_{a_k}}^{(a_k)} + \sqrt{\frac{2\log(1/\delta)}{u_{a_k}}} \geq Q_{a_1}\right) \\
&\overset{\text{Def } \Delta_{a_k}}{=} \mathbb{P}\left(\overline{Q}_{u_{a_k}}^{(a_k)} - Q_{a_k} \geq \Delta_{a_k} - \sqrt{\frac{2\log(1/\delta)}{u_{a_k}}}\right) \\
&\overset{\text{Eq. (2.4)}}{\leq} \mathbb{P}\left(\overline{Q}_{u_{a_k}}^{(a_k)} \geq Q_{a_k} + \frac{1}{2}\Delta_{a_k}\right) \\
&\overset{\text{Cor. 2.3.4}}{\leq} \exp\left(-\frac{u_{a_k}\Delta_{a_k}^2}{8}\right)
\end{aligned}$$

Combining the above yields

$$\mathbb{E}[T_{a_k}(n)] \leq u_{a_k} + n\left(n\delta + \exp\left(-\frac{u_{a_k}\Delta_{a_k}^2}{8}\right)\right).$$

With $n\delta = \frac{1}{n}$ and $\exp\left(-\frac{u_{a_k}\Delta_{a_k}^2}{8}\right) \leq \exp(-\log(1/\delta)) = \frac{1}{n^2}$ we can finally estimate

$$\mathbb{E}[T_{a_k}(n)] \leq \left\lceil \frac{16\log(n)}{\Delta_{a_k}^2} \right\rceil + 1 + \frac{1}{n} + \frac{1}{n^2} \leq \left\lceil \frac{16\log(n)}{\Delta_{a_k}^2} \right\rceil + 2 \leq \frac{16\log(n)}{\Delta_{a_k}^2} + 3.$$

Hence, the model-based regret bound is

$$\begin{aligned}
R_n(\pi) &= \sum_{a_k:\Delta_{a_k}\geq 0} \Delta_{a_k}\mathbb{E}[T_{a_k}(n)] \leq \sum_{a_k:\Delta_{a_k}\geq 0} \Delta_{a_k}\left(3 + \frac{16\log(n)}{\Delta_{a_k}^2}\right) \\
&= 3\sum_{a_k\in\mathcal{A}} \Delta_{a_k} + 16\log(n)\sum_{a_k:\Delta_{a_k}>0} \frac{1}{\Delta_{a_k}},
\end{aligned}$$

which proves the theorem. □

If the second best arm a_k is almost as good as the best arm, the reward gap Δ_{a_k} between them is quite small and therefore the inverse of it can be large, influencing the bound. On the other hand, a statement of the model-independent regret bound eliminates the dependence on the reward gap but the dependence on n becomes less favorable.

Theorem 2.3.8. Alternative regret bound (Compare [5], Theorem 1.3.9)
Suppose ν is a bandit model with 1-subgaussian arms, $n \in \mathbb{N}$, and $\delta = \frac{1}{n^2}$. Then the UCB algorithms has the alternative model-independent upper regret bound

$$R_n(\pi) \leq 8\sqrt{Kn\log(n)} + 3\sum_{a_k \in \mathcal{A}} \Delta_{a_k}.$$

Proof. First, we recall that in the previous proof it was shown that

$$\mathbb{E}[T_{a_k}(n)] \leq \frac{16\log(n)}{\Delta_{a_k}^2} + 2.$$

From the regret decomposition Lemma 2.1.10, we know that small reward gaps Δ_{a_k} do not have a major influence on the overall regret, as they only have a multiplicative effect. We therefore only estimate the regret for arms with larger reward gaps than Δ and then minimize over this threshold value. The total regret can be written as follows:

$$R_n(\pi) = \sum_{a_k \in \mathcal{A}} \Delta_{a_k}\mathbb{E}[T_a(n)] = \sum_{a_k \in \mathcal{A}:\Delta_{a_k} < \Delta} \Delta_{a_k}\mathbb{E}[T_{a_k}(n)] + \sum_{a_k \in \mathcal{A}:\Delta_{a_k} \geq \Delta} \Delta_{a_k}\mathbb{E}[T_{a_k}(n)].$$

For the arms with $\Delta_{a_k} < \Delta$ we set the trivial bound $\mathbb{E}[T_{a_k}(n)] \leq n$:

$$\sum_{a_k \in \mathcal{A}:\Delta_{a_k} < \Delta} \Delta_{a_k}\mathbb{E}[T_{a_k}(n)] \leq \sum_{a_k \in \mathcal{A}:\Delta_{a_k} < \Delta} \Delta_{a_k} n \leq n\Delta.$$

For the arms with $\Delta_a \geq \Delta$ we use the previously proven bound and sum both parts up:

$$R_n(\pi) \leq n\Delta + \sum_{a_k \in \mathcal{A}:\Delta_{a_k} \geq \Delta} \left(\frac{16\log(n)}{\Delta_{a_k}} + 3\Delta_{a_k}\right) \leq n\Delta + \frac{16K\log(n)}{\Delta} + 3\sum_{a_k \in \mathcal{A}} \Delta_{a_k}$$

Since Δ can be chosen arbitrarily, we minimize the expression on the right-hand side as a function of Δ. By differentiating and finding the minimum, we obtain $\Delta = \sqrt{\frac{16K\log(n)}{n}}$. Inserting Δ and simplifying results in the conclusion of the proof:

$$R_n(\pi) \leq 8\sqrt{Kn\log(n)} + 3\sum_{a \in \mathcal{A}} \Delta_{a_k}.$$

□

Remark 2.3.9. A large disadvantage of the UCB Algorithm that we have discussed is its dependence on n, particularly when choosing $\delta = 1/n^2$. In practice, the time horizon n is not always known in advance. Therefore, P. Auer et al. introduced in their paper [4] a variant using $\log(t)$, to address this issue. These modifications often have similar or even better regret bounds. For example, in their paper the leading constant in the model-based upper bound in Eq. (2.2) of the UCB algorithm is improved from 16 to 8, but only for bandit models where the reward distributions are bounded in [0, 1].

In conclusion, the regret bounds of the UCB algorithm are highly desirable as they provide a near-optimal reference for further adapations of the UCB algorithm.

3 Variance-aware Algorithms based on the UCB Algorithm

The basic UCB algorithm in Section 2.3 has been significantly refined, particularly since the 1980s. There are two main reasons motivating the refinement, first and most obviously the empirical performance, which can be improved, and secondly the computational complexity, which can be optimised. In this chapter, we will examine two of the most important extensions of the UCB algorithm. We begin by introducing the UCB-V algorithm in Section 3.1 and lead over to the EUCBV algorithm in Section 3.2, which can be seen as a further adaptation of the UCB-V algorithm. Both algorithms are presented in combination with the other algorithms mentioned in their corresponding publications.

3.1 Towards the UCB-V Algorithm

In this section, we present and analyze several adaptations of the UCB algorithm, as discussed by J.-Y. Audibert et al. in their paper "Exploration-exploitation tradeoff using variance estimates in multi-armed bandits" [2]. We focus specifically on the four adaptations UCB-Normal, UCB-Tuned, UCB-V, and PAC-UCB. These algorithms extend the classical UCB algorithm, which we introduced in Section 2.3, by incorporating variance estimates.

Although in their paper [4] P. Auer et al. originally introduced UCB-Normal, which they call UCB1-Normal, and UCB-Tuned, which they call UCB1-Tuned, we have chosen to present them here alongside UCB-V because they are all variance-aware adaptations of the original UCB algorithm. J.-Y. Audibert et al. also used UCB-Normal and UCB-Tuned as starting points in their work, making it logical to discuss them in this context [2]. The main contribution of their paper is the UCB-V algorithm, which incorporates variance estimates into the UCB calculations. Their numerical experiments show that UCB-V performs best when the variance is smaller than the reward range. Additionally, they introduce the PAC-UCB algorithm, further enhancing the exploration-exploitation trade-off. In this section, we will examine the underlying idea of these four algorithms and for the most interesting ones also the regret bounds. To prepare for a detailed explanation, we will provide a brief overview of their interconnections.

The UCB-Normal algorithm makes the stronger assumption that the reward of each arm follows a normal distribution with unknown mean and variance. By incorporating the unbiased empirical estimates of the variance $\bar{\sigma}^2_{a_k}$, UCB-Normal refines the exploration bonus term in the UCB function, resulting in an upper regret bound that scales with the actual variances of the arms $\sigma^2_{a_k}$, rather than depending on the choice of the parameter δ as in the original UCB Algorithm 3. Similar to UCB-Normal, UCB-Tuned uses empirical variance estimates, but does not assume a normal distribution of rewards. It is designed to work with any bounded payoff distribution. The empirical results show that UCB-Tuned outperforms other algorithms, including UCB and UCB-Normal [4]. The objective of J.-Y. Audibert et al. in [2] is to study such "variance-aware" algorithms. Therefore, they suppose a more generalized term for the exploration bonus, or also called bias sequence of the UCB function. In the specific case that the variance is much smaller than the length of the reward range, they suggest the UCB-V algorithm, extending the UCB framework by incorporating an explicit variance term into the exploration term in the UCB function as well as an option for tuning parameters through a refined exploration function. The regret bound of UCB-V as well shows improved scaling with the actual variances $\sigma_{a_k}^2$ of the arms, making it a robust choice for a wide range of applications [2].

The rest of this section is devoted to the study of the algorithms.

UCB-Normal

The UCB-Normal algorithm was initially introduced by P. Auer et al. in [4]. Consequently, we will utilize [4] as our primary source for the formulation of this algorithm and the proof of its regret bound.

UCB-Normal maintains a sublinear regret in finite time even though the mean and variance are unknown. The idea of this algorithm lies in its assumption of knowning that the rewards are normally distributed, which allows the use of the unbiased estimated variance $\bar{\sigma}_{a_k}^2$ as an estimate of the distribution's variance due to the Law of Large Numbers. The empirical estimate of the variance of an arm a_k at time step n depends on the actual payoff $X_{a_k,t}$ of this arm for all time steps t up until time step n and its empirical estimate of the reward.

Definition 3.1.1. Empirical estimates of the variance (Compare [2], Section 3)
For a given arm $a_k \in \mathcal{A}$, $X_{a_k,n}$ represents the reward when arm a_k is pulled at time n. The empirical estimate of the variance of the arm a_k including all available data up until time step n can be calculated as

$$\hat{\sigma}_{a_k}^2(n) = \frac{1}{n} \sum_{t=1}^{n} \left(X_{A_i,t} \mathbb{1}_{\{A_i = a_k\}} - \hat{Q}_{a_k}(n) \right)^2$$

The unbiased empirical estimate can be calculated as

$$\bar{\sigma}_{a_k}^2(n) = \frac{1}{n-1} \left(\sum_{i=1}^{n} X_{a_k,i}^2 - n\hat{Q}_{a_k}^2(s) \right).$$

The UCB-Normal algorithm is formally defined in Algorithm 4.

We proceed with the proof of the regret bound of the UCB-Normal algorithm. It requires the bounds for the tails of the Student's t-distribution in Conjecture 3.1.2 and bounds on the tail of the χ^2-distribution in Conjecture 3.1.3 that could only be verified numerically.

Conjecture 3.1.2. Let Y be a Student random variable with s degrees of freedom. Then, for all $0 \leq a \leq \sqrt{2(s+1)}$,

$$\mathbb{P}\{Y \geq a\} \leq e^{-a^2/4}.$$

Conjecture 3.1.3. Let Z be a χ^2 random variable with s degrees of freedom. Then

$$\mathbb{P}\{Z \geq 4s\} \leq e^{-(s+1)/2}.$$

Algorithm 4 UCB-Normal Algorithm

```
Data: Bandit model $\nu$, time horizon $n$
Result: Actions $A_1, \dots, A_n$ and rewards $X_1, \dots, X_n$
1: Initialize $\hat{Q}(0)$, $t = 0$
2: while $t \leq n$ do
3:     if there exists an arm $a_j$ that has been played less than $\lceil 8 \log n \rceil$ times then
4:         Set $A_t = a_j$
5:     else
6:         Set $A_t = \arg\max_{a_k} \left( \hat{Q}_{a_k} + \sqrt{16 \bar{\sigma}^2_{a_k}(t-1) \cdot \frac{\log(t-1)}{T_{a_k}(t-1)}} \right)$
7:     Obtain reward $X_t$ by playing arm $A_t$
8:     Update the estimated action value $\hat{Q}_{a_k}(t)$
9:     $t \leftarrow t + 1$
```

In the Theorem 3.1.4 we state and prove the regret bound for the UCB1-Normal algorithm.

Theorem 3.1.4. Regret bound of the UCB-Normal algorithm (Compare [4], Theorem 4)
For all $K > 1$, if UCB-Normal is run on K arms having normal reward distributions $\mathbb{P}_1, \dots, \mathbb{P}_K$, then its cumulated regret after any number n of plays is bounded by

$$R_n(\pi) \leq 256(\log n) \left(\sum_{a_k : \Delta_{a_k}} \frac{\sigma^2_{a_k}}{\Delta_{a_k}} \right) + \left(1 + \frac{\pi^2}{2} + 8 \log n \right) \left(\sum_{k=1}^{K} \Delta_{a_k} \right)$$

where $\mu_1, \dots, \mu_K$ and $\sigma^2_1, \dots, \sigma^2_K$ are the means and variances of the distributions $\mathbb{P}_1, \dots, \mathbb{P}_K$.

Proof. (Compare [4], Appendix B: Proof of Theorem 4)
We now proceed with the proof of Theorem 3.1.4. Fix an arm a_k and, for any t, define the confidence interval width $c_{t,s}$ as

$$c_{t,s}(a_k) = \sqrt{16 \frac{1}{s-1} \left(\sum_{i=1}^{s} X^2_{a_k,i} - s \hat{Q}^2_{a_k}(s) \right) \frac{\log t}{s}} = \sqrt{16 \bar{\sigma}^2_{a_k} \frac{\log t}{s}},$$

where $c_{t,s}$ is the confidence width. Let $c^*_{t,s} = c_{t,s}(a^*)$ be the corresponding quantity for the optimal arm. To upper bound $T_{a_k}(n)$, we define for each $t \geq 1$ a bounded indicator function of $\mathbb{1}_{\{A_t = a_k\}}$, then for any $\ell \in \mathbb{N}^+$, we can write

$$T_{a_k}(n) = 1 + \sum_{t=K+1}^{n} \mathbb{1}_{\{A_t = a_k\}} \overset{\text{Def } T_{a_k}}{\leq} \ell + \sum_{t=K+1}^{n} \mathbb{1}_{\{A_t = a_k, T_{a_k}(t-1) \geq \ell\}}$$

$$\leq \ell + \sum_{t=K+1}^{n} \tag{3.1}$$

$$\left\{\hat{Q}^*(T_{a^*}(t-1)) + c^*_{t-1,T_{a^*}(t-1)} \leq \hat{Q}_{a_k}(T_{a^*}(t-1)) + c_{t-1,T_{a_k}(t-1)}, T_{a_k}(t-1) \geq \ell\right\}$$

$$\leq \ell + \sum_{t=K+1}^{n} \left\{\min_{0<s<t} \hat{Q}^*(s) + c_{t-1,T_{a^*}(t-1)} \leq \max_{\ell \leq s_{a_k} < t} \hat{Q}_{a_k}(s_{a_k}) + c_{t-1,s_{a_k}}\right\}$$

$$\leq \ell + \sum_{t=1}^{\infty}\sum_{s=1}^{t-1}\sum_{s_{a_k}=\ell}^{t-1} \left\{\hat{Q}^*(s) + c_{t,s} \leq \hat{Q}_{a_k}(s_{a_k}) + c_{t,s_{a_k}}\right\}$$

$$\leq \ell + \sum_{t=1}^{\infty}\sum_{s=1}^{t-1}\sum_{s_{a_k}=\ell}^{t-1} \Big[\ \left\{\hat{Q}^*(s) \leq Q^* - c^*_{t,s}\right\} + \left\{\hat{Q}_{a_k}(s_{a_k}) \geq Q_{a_k} + c_{t,s_{a_k}}\right\} \tag{3.2}$$

$$+ \left\{Q^* < \mu_{a_k} + 2c_{t,s_{a_k}}\right\} \Big] \ .$$

Now consider the probability that $\hat{Q}_{a_k}(s_{a_k}) \geq Q_{a_k} + c_{t,s_{a_k}}$. The normalized random variable

$$\frac{\hat{Q}_{a_k}(s_{a_k}) - Q_{a_k}}{\sqrt{\bar{\sigma}^2_{a_k}(s_{a_k})/s_{a_k}}}$$

follows a Student's t-distribution with $s_{a_k} - 1$ degrees of freedom as proven in S. Wilks et al. [17]. Using the bound for the Student's t-distribution of conjecture 3.1.2) with parameters $s = s_{a_k} - 1$ and $a = 4\sqrt{\log t}$, we get for $s_{a_k} \geq 8 \log t$

$$\mathbb{P}\left\{\hat{Q}_{a_k}(s_{a_k}) \geq Q_{a_k} + c_{t,s_{a_k}}\right\} = \mathbb{P}\left\{\frac{\hat{Q}_{a_k}(s_{a_k}) - Q_{a_k}}{\sqrt{\bar{\sigma}^2_{a_k}(s_{a_k})/s_{a_k}}} \geq 4\sqrt{\log t}\right\} \leq t^{-4}.$$

Similarly, we can bound the probability that $\hat{Q}^*(s) \leq Q^* - c^*_{t,s}$. Furthermore, the quantity

$$\frac{\sum_{i=1}^{s_{a_k}} X^2_{a_k,i} - s_{a_k}\hat{Q}^2_{a_k,s_{a_k}}}{\sigma^2_{a_k}}$$

follows a χ^2-distribution with $s_{a_k} - 1$ degrees of freedom as S. Wilks has proven in [17]. Using the inequality of conjecture 3.1.3) with $s = s_{a_k} - 1$ and $a = 4s$, we get

$$\begin{aligned}\mathbb{P}\left\{Q^* < Q_{a_k} + 2c_{t,s_{a_k}}\right\} &= \mathbb{P}\left\{\frac{\sum_{i=1}^{s_{a_k}} X^2_{a_k,i} - s_{a_k}\hat{Q}^2_{a_k,s_{a_k}}}{\sigma^2_{a_k}} > (s_{a_k} - 1)\frac{\Delta^2_{a_k}}{\sigma^2_{a_k}}\frac{s_{a_k}}{64\log t}\right\} \\ &\leq \mathbb{P}\left\{\frac{\sum_{i=1}^{s_{a_k}} X^2_{a_k,i} - s_{a_k}\hat{Q}^2_{a_k,s_{a_k}}}{\sigma^2_{a_k}} > 4(s_{a_k} - 1)\right\} \\ &\leq e^{-s_{a_k}/2} \leq t^{-4},\end{aligned}$$

for $s_{a_k} \geq \max\left\{\frac{256\sigma^2_{a_k}}{\Delta^2_{a_k}}, 8\right\}\log t$.

Finally, setting

$$\ell = \left\lceil \max\left\{\frac{256\sigma_{a_k}^2}{\Delta_{a_k}^2}, 8\right\}\log t\right\rceil$$

completes the proof of the theorem.

$\square$

UCB-Tuned

In their work, J.-Y. Audibert et al. [2] took reference on the performance of the UCB-Tuned algorithm, which was primarily introduced in the experimental section of P. Auer et al. [4]. This fine-tuned version of the UCB algorithm incorporates an upper confidence bound of the empirical estimate for the variance of an arm into the UCB function of the algorithm, to decide, which arm to play. In the experimental section of their paper [4], they show a empirical superiority of the UCB-Tuned algorithm, although these regret bounds were not formally proven.

Remark 3.1.5. The UCB-Tuned algorithm modifies the upper confidence bound used in the UCB function. Instead of the classical bound $\sqrt{2\log(1/\delta)/T_{a_k}(t)}$ for a 1-subgaussian random variable X, UCB-Tuned employs the following upper confidence bound with the upper bound 1/4 for the variance of a Bernoulli random variable for an arm a_k.

$$\sqrt{\frac{\log(1/\delta)}{T_{a_k}(t)}\min\left\{\frac{1}{4}, V_{a_k}(t)\right\}}.$$

Here, an upper confidence bound of the variance $\sigma_{a_k}^2$ of arm a_k after being played $T_{a_k}(t)$ times within the first t plays is at most the empirical estimate for the variance plus an additional bias term,

$$V_{a_k}(t) = \hat{\sigma}_{a_k}^2(t) + \sqrt{\frac{2\log t}{T_{a_k}(t)}}.$$

Remark 3.1.6. Due to the similarity of the upper confidence bound as described in Remark 3.1.5 of the UCB-Tuned algorithm with the UCB algorithm stated in Algorithm 3, and since the core idea of both algorithms involves adding an exploration term to the empirical mean of the rewards, we have decided to not to include a separate pseudocode for UCB-Tuned. The adaptation from UCB to UCB-Tuned involves only this specific change in the exploration term, maintaining the overall structure and approach of the original UCB algorithm.

Despite the absence of a formal proof in the paper of P. Auer et al. [4], empirical evidence shows that UCB-Tuned often outperforms the basic UCB algorithm, as simulated in their experimental section under various conditions.

UCB-V

The primary contribution of J.-Y. Audibert et al.'s work [2] is the introduction of the UCB-V algorithm. This algorithm integrates the empirical estimation of variance into its exploration term in the upper confidence bound, enhancing the performance of the upper confidence bound method. Let $s = T_{a_k}(t-1)$ the number of pulls of arm a_k until time step $t-1$. Assuming all rewards are bounded within $[0, b]$ with $b > 0$ known to the decision maker, the upper confidence bound function for the UCB-V algorithm is defined as

$$B_{a_k,s,t} = \hat{Q}_{a_k}(t) + \sqrt{\frac{2\hat{\sigma}^2_{a_k}(t) \cdot \varepsilon_{s,t}}{s}} + c\frac{3b\varepsilon_{s,t}}{s},$$

Although the UCB-V algorithm is not explicitly presented in [2], we define it in Algorithm 5. In the definition of the algorithm we choose a function $\varepsilon_{s,t}$ that we call exploration function. The function $t \mapsto \varepsilon_{s,t}$ is nondecreasing.

Algorithm 5 UCB-V Algorithm

Data: Bandit model ν, time horizon n, parameters b, c and exploration function ε
Result: Actions $A_1, \ldots, A_n$ and rewards $X_1, \ldots, X_n$

```
1: while $t \leq n$ do
2:     Set $A_t = \arg\max_{a_k} B_{a_k,s,t}$
3:     Obtain reward $X_t$ by playing arm $A_t$
4:     Update $\hat{Q}_{a_k}(t)$, $s_{a_k} = s_{a_k} + 1$
5:     $t \leftarrow t + 1$
```

The general functionality of UCB-V mirrors that of UCB. Additionally, $\varepsilon_{s,t}$ is often set to $\theta \log(t)$. This exploration function ensures that the characteristics of the UCB algorithm are retained. It increases over time for a specific arm a_k if that arm has not been selected for a while, eventually dominating the remaining component $\hat{Q}_{a_k}(t)$ of $B_{a_k,s,t}$. Furthermore, the exploration term in the upper confidence bound function $B_{a_k,s,t}$,

$$\sqrt{\frac{2\hat{\sigma}^2_{a_k}(t) \cdot \varepsilon_{s,t}}{s}} + c\frac{3b\varepsilon_{s,t}}{s},$$

serves as a bound of the confidence width of the estimated action value, making $B_{a_k,s,t}$ the largest for this specific arm a_k among all.

The optimization of the UCB-V algorithm can be viewed in two ways. Firstly, if $\varepsilon_{s,t}$ depends solely on t, the number of times, suboptimal arms are selected, is restricted, scaling sublinear with t. Alternatively, $\varepsilon_{s,t}$ can depend only on s. We will explore this in more detail in the further developed version of the algorithm afterwards in Section 3.1.1. For the first option, assuming that $\varepsilon = (\varepsilon_t)_{t \geq 0}$ does not depend on s, UCB-V achieves the regret bound of Theorem 3.1.7.

Theorem 3.1.7. Regret bound of the UCB-V algorithm (Compare [2], Theorem 3)

We have

$$R_n \leq \sum_{a_k:\Delta_{a_k}>0} \left\{ 1 + 8\,(c \vee 1) \left(\frac{\sigma^2_{a_k}}{\Delta_{a_k}} + \frac{2b}{\Delta_{a_k}} \right) \varepsilon_n + n e^{-\varepsilon_n} \left(\frac{24\sigma^2_{a_k}}{\Delta_{a_k}} + \frac{4b}{\Delta_{a_k}} \right) \right.$$
$$\left. + \sum_{t=16\varepsilon_n}^{n} \beta((c \wedge 1)\varepsilon_t, t) \right\} \Delta_{a_k},$$

where we introduce $\beta(x,t) = 3\inf_{1\leq\alpha\leq 3} \left(\frac{\log t}{\log \alpha} \wedge t\right) \exp(-x/a)$. Therefore, $\beta\left((c \wedge 1)\,\varepsilon_t, t\right)$ is essentially of order $e^{-(c\wedge 1)\varepsilon_t}$.

To prove the regret bound of the UCB-V algorithm, we apply different key results of Theorem 3.1.8, Lemma 3.1.9 and Theorem 3.1.10. Since we will refer to these statements in the proof of the regret bound, let us introduce them first.

Theorem 3.1.8. (Compare [2], Theorem 2) The followings hold:

(i) After K plays, each arm has been pulled once.

(ii) Pick an arm a_k and a time $n \in \mathbb{N}^+$. For any $\tau \in \mathbb{R}$ and any integer $u > 1$, it holds that

$$T_{a_k}(n) \leq u + \sum_{t=u+K-1}^{n} \left(\mathbb{1}_{\left\{\exists s\in\{u,\ldots,t-1\}:B_{a_k,s,t}>\tau\right\}} + \mathbb{1}_{\left\{\exists s^*\in\{1,\ldots,t-1\}:\tau\geq B_{a^*,s^*,t}\right\}} \right). \tag{3.3}$$

Hence, also

$$\mathbb{E}[T_{a_k}(n)] \leq u + \sum_{t=u+K-1}^{n} \sum_{s=u}^{t-1} \mathbb{P}\left(B_{a_k,s,t} > \tau\right)$$
$$+ \sum_{t=u+K-1}^{n} \mathbb{P}\left(\exists s \in \{1,\ldots,t-1\} : B_{a^*,s,t} \leq \tau\right). \tag{3.4}$$

Both inequalities hold independently of the form of the quantity $B_{a_k,s,t}$. Further, it holds that

$$\mathbb{P}(T_{a_k}(n) > u) \leq \sum_{t=u+1}^{n} \mathbb{P}(B_{a_k,u,t} > \tau) + \mathbb{P}\left(\exists s \in \{1,\ldots,n-u\} : B_{a^*,s,u+s} \leq \tau\right). \tag{3.5}$$

Note that even though the above statements hold for any arm, the bounds are trivial for the optimal arms.

Proof. (Compare [2], Proof of Theorem 2)
Part (i) is trivial since at the beginning each arm has an infinite UCB value, which becomes finite as soon as the arm has been played once. Let us thus turn to the proof of part (ii). To obtain Eq. (3.3), we note that

$$T_{a_k}(n) - u \leq \sum_{t=u+K-1}^{n} \mathbb{1}_{\left\{A_t=a_k; T_{a_k}(t)>u\right\}} \tag{3.6}$$

$$\leq \mathbb{1}_{\left\{A_t=a_k; u\leq T_{a_k}(t-1); B_{a_k,T_{a_k}(t-1),t}\leq B_{a*,T_{a^*}(t-1),t}\right\}} \tag{3.7}$$

$$\leq \mathbb{1}_{\left\{\exists s\in\{u,\ldots,t-1\}: B_{a_k,s,t}>\tau\right\}} + \mathbb{1}_{\left\{\exists s^*\in\{1,\ldots,t-1\}: \tau\geq B_{a*,s^*,t}\right\}} \tag{3.8}$$

For the inequality (3.6) the term $\mathbb{1}_{A_t=a_k}$ indicates that arm a_k was pulled at time t, and $T_{a_k}(t) > u$ ensures that a_k has been pulled more than u times by time t. Inequality (3.7) holds because the condition $T_{a_k}(t) > u$ is ensured if $T_{a_k}(t-1) \geq u$, and the arm a_k was selected at time t due to its UCB value being less than that of the optimal arm a_k^*. Using the definition of $B_{a_k,s,t}$, we can further decompose to inequality (3.8).
Summing Eq. (3.8) from $t = u + K - 1$ to n yields:

$$T_{a_k}(n) - u \leq \sum_{t=u+K-1}^{n} \left(\mathbb{1}_{\left\{\exists s\in\{u,\ldots,t-1\}: B_{a_k,s,t}>\tau\right\}} + \mathbb{1}_{\left\{\exists s^*\in\{1,\ldots,t-1\}: \tau\geq B_{a^*,s^*,t}\right\}}\right) \tag{3.9}$$

Adding u to both sides gives Eq. (3.3):

$$T_{a_k}(n) \leq u + \sum_{t=u+K-1}^{n} \left(\mathbb{1}_{\left\{\exists s\in\{u,\ldots,t-1\}: B_{a_k,s,t}>\tau\right\}} + \mathbb{1}_{\left\{\exists s^*\in\{1,\ldots,t-1\}: \tau\geq B_{a^*,s^*,t}\right\}}\right) \tag{3.10}$$

Taking expectations of both sides, we prove Eq. (3.4).:

$$\begin{aligned}\mathbb{E}[T_{a_k}(n)] &\leq u + \sum_{t=u+K-1}^{n} \mathbb{E}\left[\mathbb{1}_{\left\{\exists s\in\{u,\ldots,t-1\}: B_{a_k,s,t}>\tau\right\}}\right] + \mathbb{E}\left[\mathbb{1}_{\left\{\exists s^*\in\{1,\ldots,t-1\}: \tau\geq B_{a^*,s^*,t}\right\}}\right] \\ &\leq u + \sum_{t=u+K-1}^{n} \sum_{s=u}^{t-1} \mathbb{P}\left(B_{a_k,s,t} > \tau\right) + \sum_{t=u+K-1}^{n} \mathbb{P}\left(\exists s \in \{1,\ldots,t-1\} : B_{a^*,s,t} \leq \tau\right)\end{aligned} \tag{3.11}$$

To prove Eq. (3.5), consider the event $\{T_{a_k}(n) > u\}$. This implies that a_k was pulled more than u times by time n. If $T_{a_k}(n) > u$, there must exist a time $t \in \{u+1,\ldots,n\}$ such that:

$$B_{a_k,u,t} > \tau \quad \text{or} \quad \exists s \in \{1,\ldots,n-u\} : B_{a^*,s,u+s} \leq \tau \tag{3.12}$$

If neither of these conditions held, then a_k would not have been pulled more than u times, as its UCB value would be too low compared to the optimal arm a_k^*.

Thus, we have:

$$\{T_{a_k}(n) > u\} \subset \{\exists t \in \{u+1, \ldots, n\} : B_{a_k,u,t} > \tau \text{ or } \exists s \in \{1, \ldots, n-u\} : B_{a^*,s,u+s} \leq \tau\} \tag{3.13}$$

Taking the probability of both sides and using the union bound, we prove Eq. (3.5):

$$\mathbb{P}(T_{a_k}(n) > u) \leq \sum_{t=u+1}^{n} \mathbb{P}(B_{a_k,u,t} > \tau) + \mathbb{P}\left(\exists s \in \{1, \ldots, n-u\} : B_{a^*,s,u+s} \leq \tau\right) \tag{3.14}$$

□

Lemma 3.1.9. (Compare [2], Lemma 1)
Let $u = \left\lceil 8\,(c \vee 1) \left(\frac{\sigma^2_{a_k}}{\Delta_{a_k}} + \frac{2b}{\Delta_{a_k}}\right) \varepsilon_n \right\rceil$. Then for any s, t such that $u \leq s \leq t \leq n$, $t \geq 2$, it holds that

$$\mathbb{P}\left(B_{a_k,s,t} > Q^*\right) \leq 2e^{-\frac{s\Delta^2_{a_k}}{(8\sigma^2_{a_k} + 4b\Delta_{a_k})/3}}.$$

Note that for any suboptimal arm a_k the probability decays exponentially in s for s large enough, independently of the value of t and n. Intuitively, this makes sense as the main term in $B_{a_k,s,t}$ is $\hat{Q}_{a_k}(s)$, which estimates $Q_{a_k} < Q^*$.

Proof. (Compare [2], Proof of Lemma 1)
From the definition of $B_{a_k,s,t}$, we obtain

$$\mathbb{P}\left(B_{a_k,s,t} > Q^*\right) \leq \mathbb{P}\left(\hat{Q}_{a_k}(s) + \sqrt{\frac{2\hat{\sigma}^2_{a_k}(s)\varepsilon_t}{s}} + \frac{3bc\varepsilon_t}{s} > Q_{a_k} + \Delta_{a_k}\right).$$

Utilizing the definition of the empirical variance $\hat{\sigma}^2_{a_k}(s)$, we can bound $\hat{\sigma}^2_{a_k}(s) \leq \sigma^2_{a_k} + (b\Delta_{a_k})/2$. Substituting this bound, we get:

$$\mathbb{P}\left(B_{a_k,s,t} > Q^*\right) \leq \mathbb{P}\left(\hat{Q}_{a_k}(s) + \sqrt{\frac{2(\sigma^2_{a_k} + \frac{b\Delta_{a_k}}{2})\varepsilon_t}{s}} + \frac{3bc\varepsilon_t}{s} > Q_{a_k} + \Delta_{a_k}\right)$$
$$+ \mathbb{P}\left(\hat{\sigma}^2_{a_k}(s) \geq \sigma^2_{a_k} + \frac{b\Delta_{a_k}}{2}\right).$$

To bound the second term, we need to consider the empirical estimate of the variance $\hat{\sigma}^2_{a_k}(s)$. By rewriting it as

$$\hat{\sigma}^2_{a_k}(s) = \frac{1}{s}\sum_{i=1}^{s}(X_{a_k,i} - Q_{a_k})^2 - (Q_{a_k} - \hat{Q}_{a_k}(s))^2.$$

Thus,

$$\mathbb{P}\left(\hat{\sigma}^2_{a_k}(s) \geq \sigma^2_{a_k} + \frac{b\Delta_{a_k}}{2}\right) \leq \mathbb{P}\left(\frac{1}{s}\sum_{i=1}^{s}(X_{a_k,i} - Q_{a_k})^2 - \sigma^2_{a_k} \geq \frac{b\Delta_{a_k}}{2}\right).$$

To bound the first term, we introduce $\varepsilon'_n = (c \vee 1)\varepsilon_n$. Given the choice of u, which ensures that $u \leq s, t \leq n$, we have:

$$\sqrt{\frac{2(\sigma^2_{a_k} + \frac{b\Delta_{a_k}}{2})\varepsilon_t}{s}} + \frac{3bc\varepsilon_t}{s} \leq \sqrt{\frac{2(\sigma^2_{a_k} + b\Delta_{a_k})\varepsilon'_n}{u}} + \frac{3b\varepsilon'_n}{u} \tag{3.15}$$

$$\overset{\text{Def } u}{\leq} \sqrt{\frac{(2\sigma^2_{a_k} + b\Delta_{a_k})\Delta^2_{a_k}}{8(\sigma^2_{a_k} + 2b\Delta_{a_k})}} + \frac{3b\Delta^2_{a_k}}{8(\sigma^2_{a_k} + 2b\Delta_{a_k})} \tag{3.16}$$

Next, simplify the terms inside the square root through factoring out $\Delta_{a_k}/2$ and the fraction combine the terms yields

$$\sqrt{\frac{(2\sigma^2_{a_k} + b\Delta_{a_k})\Delta^2_{a_k}}{8(\sigma^2_{a_k} + 2b\Delta_{a_k})}} + \frac{3b\Delta^2_{a_k}}{8(\sigma^2_{a_k} + 2b\Delta_{a_k})} = \frac{\Delta_{a_k}}{2}\left(\sqrt{\frac{2\sigma^2_{a_k} + b\Delta_{a_k}}{2\sigma^2_{a_k} + 4b\Delta_{a_k}}} + \frac{3b\Delta_{a_k}}{4\sigma^2_{a_k} + 8b\Delta_{a_k}}\right).$$

To show that this is less than or equal to $\frac{\Delta_{a_k}}{2}$, it remains to be proven that:

$$\sqrt{\frac{2\sigma^2_{a_k} + b\Delta_{a_k}}{2\sigma^2_{a_k} + 4b\Delta_{a_k}}} + \frac{3b\Delta_{a_k}}{4\sigma^2_{a_k} + 8b\Delta_{a_k}} \leq 1.$$

Setting $x = \sqrt{\frac{2\sigma^2_{a_k} + b\Delta_{a_k}}{2\sigma^2_{a_k} + 4b\Delta_{a_k}}}$, it follows that $x \leq 1$, thus:

$$x + \frac{3b\Delta_{a_k}}{4\sigma^2_{a_k} + 8b\Delta_{a_k}} \leq 1.$$

Putting it all together, we can now bound:

$$\begin{aligned}\mathbb{P}(B_{a_k,s,t} > Q^*) &\leq \mathbb{P}\left(\hat{Q}_{a_k}(s) - Q_{a_k} > \frac{\Delta_{a_k}}{2}\right) + \mathbb{P}\left(\frac{1}{s}\sum_{i=1}^{s}(X_{a_k,i} - Q_{a_k})^2 - \sigma^2_{a_k} \geq \frac{b\Delta_{a_k}}{2}\right)\\ &\leq 2e^{-\frac{s\Delta^2_{a_k}}{(8\sigma^2_{a_k} + 4b\Delta_{a_k})/3}},\end{aligned}$$

where in the last step we used an adaptation of Bernstein's inequality to obtain the exponential bound. □

Theorem 3.1.10. (Compare [2], Theorem 1)
Let $X_1, \dots, X_t$ be iid random variables taking their values in $[0, b]$. Let $Q := Q_{a_1}$ be their common action value. Consider the estimated action value $\hat{Q}(t)$ as in Definition 2.1.11 and empirical estimate of the variance $\hat{\sigma}^2(t)$ as in Definition 3.1.1 for iid random variables.

Then, for any $t \in \mathbb{N}$ and $x > 0$, with probability at least $1 - 3e^{-x}$,

$$|\hat{Q}(t) - Q| \leq \sqrt{\frac{2\hat{\sigma}^2(t)x}{t}} + \frac{3bx}{t}.$$

Furthermore, defining

$$\beta(x,t) = 3 \inf_{1 \leq \beta \leq 3} \left(\frac{\log t}{\log \beta} \wedge t \right) e^{-x/\beta},$$

as in Theorem 3.1.7 where $u \wedge v$ denotes the minimum of u and v, we have for any $t \in \mathbb{N}$ and $x > 0$, with probability at least $1 - \beta(x,t)$,

$$|\hat{Q}(s) - Q| \leq \sqrt{\frac{2\hat{\sigma}^2(s)x}{s}} + \frac{3bx}{s}$$

holds simultaneously for $s \in \{1, 2, \dots, t\}$.

We provide a proof sketch instead of the full proof to offer a concise overview of the steps involved. The full proof involves detailed calculations and applications of concentration inequalities, which are beyond the scope of this section. For further details please refer to the Proof of Theorem 1 in J.-Y. Audibert et al. [2].

Proof Sketch. (Compare [2], Proof of Theorem 1)
The proof is based on applying Bernstein's inequality 2.3.5 to the sum of the random variables X_i. First, recall that for iid random variables $X_1, \dots, X_t$ taking values in $[0, b]$, Bernstein's inequality gives us a bound on the deviation of the empirical action value from the actual value.
To apply Bernstein's inequality, we need to bound the variance term. Let $\sigma^2 = \mathrm{Var}(X_1)$. Then, the empirical variance $\hat{\sigma}^2(t)$ is an unbiased estimator of σ^2. Using this, we can derive that for any $x > 0$ and $t \in \mathbb{N}$, with high probability,

$$|\hat{Q}(t) - Q| \leq \sqrt{\frac{2\sigma^2 x}{t}} + \frac{3bx}{t}.$$

Next, we account for the empirical variance $\hat{\sigma}^2(t)$ instead of the true variance σ^2. This yields the result

$$|\hat{Q}(t) - Q| \leq \sqrt{\frac{2\hat{\sigma}^2(t)x}{t}} + \frac{3bx}{t}.$$

Further details on this inequality with the formal name "Benett's inequality" can be found in the Appendix 3 of [2]. The second part of the theorem introduces a more refined bound that holds uniformly over time. By minimizing the bound over a range of β values, we can ensure that the bound holds with high probability for all $s \in \{1, 2, \dots, t\}$ simultaneously. The infimum over β provides the best possible confidence bound, leading to the definition of $\beta(x,t)$. □

The proof of the regret bound of the UCB-V algorithm, stated in Theorem 3.1.7, is structured into three main steps: First, we decompose the regret using the regret decomposition Lemma 2.1.10. Next, we bound $\mathbb{E}[T_{a_k}(n)]$ by breaking it down into three parts, each of which is constrained by the bounds established in Theorem 3.1.8 and Lemma 3.1.9. Finally, we demonstrate the uniformity in time using Theorem 3.1.10.

Proof of the Regret bound of the UCB-V algorithm of 3.1.7. (Compare [2], Proof of Theorem 3)
Because of the regret decomposition Lemma 2.1.10, it holds that $R_n = \sum_{a_k} \Delta_{a_k}\mathbb{E}[T_{a_k}(n)]$. Consequently, it suffices to bound $\mathbb{E}\left[T_{a_k}(n)\right]$, where a_k is the index of a suboptimal arm. Thus, pick such an index a_k. We use Eq. (3.4) to bound $\mathbb{E}\left[T_{a_k}(n)\right]$ with $\tau = Q^*$ and $u = \left\lceil 8\left(\frac{\sigma^2_{a_k}}{\Delta^2_{a_k}} + \frac{2b}{\Delta_{a_k}}\right)\varepsilon'_n\right\rceil$ with $\varepsilon'_n = (c \vee 1)\varepsilon_n$, as in Lemma 3.1.9:

$$\mathbb{E}\left[T_{a_k}(n)\right] \leq u + \sum_{t=u+1}^{n}\sum_{s=u}^{t-1}\mathbb{P}\left(B_{a_k,s,t} > Q^*\right) + \sum_{t=u+1}^{n}\sum_{s=1}^{t-1}\mathbb{P}\left(B_{a^*,s,t} \leq Q^*\right). \tag{3.17}$$

Now, we need to bound each term on the right-hand side of this inequality. Starting with the first double sum, we apply Lemma 3.1.9 to get:

$$\sum_{s=u}^{t-1}\mathbb{P}\left(B_{a_k,s,t} > Q^*\right) \leq 2\sum_{s=u}^{\infty} e^{-\frac{s\Delta^2_{a_k}}{(8\sigma^2_{a_k}+4b\Delta_{a_k}/3)}}$$

To simplify this expression, we note that the series $\sum_{s=u}^{\infty} e^{-\frac{s\Delta^2_{a_k}}{(8\sigma^2_{a_k}+4b\Delta_{a_k}/3)}}$ is a geometric series. Using the formula for the sum of a geometric series and the fact that $1-e^{-x} \geq 2x/3$ for $0 \leq x \leq 3/4$, we obtain:

$$2\sum_{s=u}^{\infty} e^{-\frac{s\Delta^2_{a_k}}{(8\sigma^2_{a_k}+4b\Delta_{a_k}/3)}} = 2\frac{e^{-\frac{u\Delta^2_{a_k}}{(8\sigma^2_{a_k}+4b\Delta_{a_k}/3)}}}{1-e^{-\frac{\Delta^2_{a_k}}{(8\sigma^2_{a_k}+4b\Delta_{a_k}/3)}}} \leq 2\frac{e^{-\frac{u\Delta^2_{a_k}}{(8\sigma^2_{a_k}+4b\Delta_{a_k}/3)}}}{\frac{2\Delta^2_{a_k}}{3(8\sigma^2_{a_k}+4b\Delta_{a_k}/3)}}$$

$$= \left(\frac{24\sigma^2_{a_k}}{\Delta^2_{a_k}} + \frac{4b}{\Delta_{a_k}}\right) e^{-\frac{u\Delta^2_{a_k}}{(8\sigma^2_{a_k}+4b\Delta_{a_k}/3)}}.$$

Substituting u back into the expression, we have:

$$\sum_{s=u}^{\infty} e^{-\frac{s\Delta^2_{a_k}}{(8\sigma^2_{a_k}+4b\Delta_{a_k}/3)}} \leq \left(\frac{24\sigma^2_{a_k}}{\Delta_{a_k}} + \frac{4b}{\Delta_{a_k}}\right) e^{-\varepsilon'_n}.$$

For the second double sum in Eq. (3.17), we use the uniform deviation bound based on the empirical estimate of the variance as provided in Theorem 3.1.10. Combining these bounds, we get:

$$\mathbb{E}\left[T_{a_k}(n)\right] \leq 1 + 8\varepsilon_n' \left(\frac{\sigma_{a_k}^2}{\Delta_{a_k}^2} + \frac{2b}{\Delta_{a_k}}\right) + ne^{-\varepsilon_n'} \left(\frac{24\sigma_{a_k}^2}{\Delta_{a_k}^2} + \frac{4b}{\Delta_{a_k}}\right) + \sum_{t=u+1}^{n} \beta\left((c \wedge 1)\,\varepsilon_t, t\right).$$

This gives the announced result since by assumption $u \geq 16\varepsilon_n$. □

PAC-UCB

We now examine the scenario where the exploration function $\varepsilon_{s,t}$ depends solely on s, the number of times arm a_k has been played. This consideration leads us to the Probably Approximately Correct Upper Confidence Bound (PAC-UCB) algorithm. Its name is derived from the fact that the number of times a suboptimal arm is played with high probability is low.

Given that the dependency on t is eliminated, we denote $B_{a_k,s,t}$ as $B_{a_k,s}$ for this algorithm. Additionally, for simplicity we set $c = 1$ in the upper confidence bound $B_{a_k,s}$.

Remark 3.1.11. As the primary distinction between the PAC-UCB algorithm and the UCB algorithm lies in the modification of the exploration function, we have not to included a separate pseudocode for PAC-UCB. The algorithm's core idea remains consistent with UCB.

The following Theorem 3.1.12 prepares the proof of the regret bound through calculating the probability of playing a suboptimal arm more frequently than u_{a_k}.

Theorem 3.1.12. *Let $\beta \in (0,1)$. Consider a sequence $(\varepsilon_s)_{s\geq 0}$ that takes values in $\mathbb{R} \cup +\infty$ and satisfies $\varepsilon_s \geq 2$ and*

$$4K \sum_{s \geq 7} \exp^{-\varepsilon_s} \leq \beta. \tag{3.18}$$

Let a_k be a suboptimal arm and let u_{a_k} be the smallest integer satisfying

$$\frac{u_{a_k}}{\varepsilon_{u_{a_k}}} > \frac{8\sigma_{a_k}^2}{\Delta_{a_k}^2} + \frac{26b}{3\Delta_{a_k}}. \tag{3.19}$$

If no arm a_k satisfies the inequality then $u_{a_k} = +\infty$ and with probability at least $1-\beta$ it holds that no suboptimal arm a_k is played more than u_{a_k} times by PAC-UCB.

Remark 3.1.13. We typically choose the exploration function for the PAC-UCB algorithm as $\varepsilon_s = \log(Ks^q\beta^{-1}) \vee 2$, such that Eq. (3.18) holds.

Remark 3.1.14. By assuming that Eq. (3.19) holds for the chosen ε_s and suitable transformation, Theorem 3.1.12 states, that for any suboptimal arm a_k and a constant $\lambda > 0$, with probability at least $1-\beta$, where $\Delta_{a_k} > 0$, the number of plays is bounded by an equation, that does not depend on the total number of plays,

$$\mathcal{T}_{a_k,\beta} \leq \lambda \left(\frac{\sigma_{a_k}^2}{\Delta_{a_k}^2} + \frac{b}{\Delta_{a_k}} \right) \log \left(K \left(\frac{\sigma_{a_k}^2}{\Delta_{a_k}^2} + \frac{b}{\Delta_{a_k}} \right) \beta^{-1} \right). \tag{3.20}$$

Now we can derive the upper bound of the regret. As defined in Remark 3.1.14, $T_{a_k}(n)$ is bounded by $\mathcal{T}_{a_k,\beta}$. There are situations where the probability that the regret is of order n is at least β, in these cases, we split our upper bound of the expected regret by adding the upper bound n with probability β. Hence, it is not possible to improve the following bound.

$$R_n \leq \sum_{k=1}^{K} \mathbb{E}[T_{a_k}(n)]\Delta_{a_k} \leq (1-\beta) \sum_{a_k:\Delta_{a_k}>0} \mathcal{T}_{k,\beta}\Delta_{a_k} + \beta n. \tag{3.21}$$

3.2 Adaptations on the UCB-V algorithm

This section aims to further adapt the UCB-V algorithm introduced in Section 2.3. We will look more closely at the EUCBV algorithm, proposed by S. Mukherjee et al. in [14]. It combines elements of both the UCB-V and the UCB-Improved algorithm. The UCB-Improved algorithm developed by P. Auer and R. Ortner in [3] relies on a fundamentally different strategy to decide on the optimal arm, radically deleting all suboptimal arms from the set of options. While in [3] P. Auer et al. already suggested that including an estimate of the arms' variance could enhance the performance of the UCB-Improved algorithm, S. Mukherjee et al. implemented this idea in [14] by expanding the UCB-Improved algorithm through variance-aware adaptaions of the UCB-V algorithm.

UCB-Improved

Since the EUCBV algorithm is refined upon the strategy of the UCB-Improved algorithm, we will begin with introducing it as it was originally proposed by P. Auer et al. in [3].

The UCB-Improved algorithm differentiates itself from the UCB algorithm in Section 2.3 by adjusting the range of confidence bounds through progressively eliminating low-performing arms based on their reward gaps Δ_{a_k}. Especially for arms with rewards that are close to the optimum, corresponding to arms with small reward gaps, the regret bound of UCB-Improved provides a better performance than the regret bound of the UCB algorithm in Theorem 2.3.7. However, Δ_{a_k} is unknown to the learner. To address this, the algorithm starts with an estimated reward gap $\tilde{\Delta}$ set to 1, which is halved whenever the confidence intervals fall below $\tilde{\Delta}$.

The primary innovation in UCB-Improved lies in its aggressive arm elimination strategy. By discarding low-performing arms early in the process, the algorithm optimizes exploration by concentrating on the more promising arms. The criterion for the arm elimination works as follows. Suppose we have an arm a_k, such that for the current $\tilde{\Delta}$ it holds that

$$\tilde{\Delta} < \frac{\Delta_{a_k}}{2}$$

This means that if the estimated reward gap $\tilde{\Delta}$ is smaller than half of the reward gap Δ_{a_k} of an arm, the arm a_k has with high probability a larger reward gap than any other arm $a_j, j \neq k$, for which it holds, that $\tilde{\Delta} > \frac{\Delta_{a_l}}{2}$, $l \neq k$. Then arm a_k is likely suboptimal and can be excluded. After the exclusion of all suboptimal arms, the confidence intervals fall below $\tilde{\Delta}$. Therefore, through refinement of the rewards of the remaining arms through playing those, we can repeat the process by halving $\tilde{\Delta}$ again. This mechanism results in tighter confidence intervals compared to the original UCB.

We can consider two cases, either the time horizon n is known to the learner or not. We will restrict our attention to the case, where the time horizon is known in advance, since this has been the objective of the algorithms discussed before. If the reader is interested in the case of the unknown time horizon, we refer the explanation of this case in Section 4 of [3].

The UCB-Improved algorithm is formulated in Algorithm 6.

Remark 3.2.1. We have adopted the original version of the algorithm from P. Auer et al. in Figure 1 in [3] to retain the logical structure and the idea of the arm elimination process of it, although it would complicate the difficulty of comparing this algorithm with others we have already presented, as the running variable here must be reintroduced in the arm selection part due to the multiple selection of arms in one time step if a comparison between the algorithms is to be possible.

In Theorem 3.2.2 we state of the UCB-Improved algorithm. The proof involves two main cases: one where a suboptimal arm is not eliminated in a specific round, and another where the optimal arm is eliminated by a suboptimal arm. By bounding the regret from each case, we can establish the overall regret bound.

Theorem 3.2.2. Regret bound of the UCB-Improved algorithm (Compare [3], Theorem 3.1)

The upper bound of the regret of the UCB-Improved algorithm up to time step n is given by

$$R_n(\pi) \leq \sum_{a_k \in \mathcal{A}: \Delta_{a_k} > \lambda} \left(\Delta_{a_k} + \frac{32 \log(n \Delta_{a_k}^2)}{\Delta_{a_k}} + \frac{96}{\Delta_{a_k}} \right) + \max_{a_k \in \mathcal{A}: \Delta_{a_k} \leq \lambda} \Delta_{a_k} n$$

for all $\lambda \geq \sqrt{e/n}$.

Remark 3.2.3. The logarithmic term is suitable for λ due to the boundedness of

$$\max_{a_k \in \mathcal{A}: \Delta_{a_k} \leq \lambda} \Delta_{a_k} n$$

by $\sqrt{en}$ for a chosen $\lambda := \sqrt{e/n}$.

Algorithm 6 UCB-Improved Algorithm

```
Data: Bandit model $\nu$, time horizon $n$
Result: Actions $A_1, \dots, A_n$ and rewards $X_1, \dots, X_n$
 1: Initialization: Initialize $\tilde{\Delta}_0 := 1$, and $B_0 := \{a_1, \dots, a_K\}$ a set of arms, $\hat{Q}(0)$, $t = 0$
 2: for $t \leq \left\lfloor \frac{1}{2} \log_2 \left( \frac{n}{e} \right) \right\rfloor$ do
 3:     if $|B_t| > 1$ then Arm selection:
 4:         Choose each arm $a_k$ in $B_t$ until $T_{a_k}(t-1) := \left\lceil \frac{2 \log(n \tilde{\Delta}_t^2)}{\tilde{\Delta}_t^2} \right\rceil$
 5:     else
 6:         Choose the single arm in $B_t$ until time step $n$ is reached
 7:     for each arm $a_k$ in $B_t$ do Arm elimination:
 8:         if $\hat{Q}_{a_k}(t) + \sqrt{\frac{\log(n\tilde{\Delta}_t^2)}{2T_{a_k}(t-1)}} < \max_{a_j \in B_t} \left( \hat{Q}_{a_j}(t) - \sqrt{\frac{\log(n\tilde{\Delta}_{a_j}^2)}{2T_{a_j}(t-1)}} \right)$ then
 9:             Delete arm $a_k$ from $B_t$
10:     Set $B_{t+1}$ to the remaining arms in $B_t$, $\tilde{\Delta}_{t+1} := \tilde{\Delta}_t / 2$
11:     Obtain reward $X_t$ by playing arm $A_t$
12:     Update the estimated action value function $\hat{Q}_{a_k}(t)$
13: for $t \geq \left\lfloor \frac{1}{2} \log_2 \left( \frac{n}{e} \right) \right\rfloor$ do
14:     Obtain reward $X_t$ by playing arm $A_t = \arg\max_{a_k} \hat{Q}_{a_k}(t-1)$
15:     Update the estimated action value function $\hat{Q}_{a_k}(t)$
```

The proof of the algorithms regret bound evaluates two primary cases. In Case (a), it is shown that if a suboptimal arm a_k is not eliminated by the round m_{a_k}, its estimated value $\hat{Q}_{a_k}$ must be significantly lower than the optimal arm's value, with the probability of failure to eliminate being exponentially small. In Case (b), two sub-cases are considered: (b1) where all suboptimal arms are correctly eliminated or (b2) where the optimal arm a^* might be incorrectly eliminated by some suboptimal arm. The proof demonstrates that both scenarios lead to a bounded regret, summing up to an overall bound that accounts for all potential errors and suboptimal arms, thus confirming the desired regret bound.

Proof of the regret bound of the UCB-Improved algorithm. (Compare [3], Proof of Theorem 3.1)

In the following, we use a^* to indicate an arbitrary but fixed optimal arm. Further, for each suboptimal arm a_k, let

$$m_{a_k} := \min\{m \mid \tilde{\Delta}_m < \Delta_{a_k}/2\}$$

be the first round in which $\tilde{\Delta}_m < \Delta_{a_k}/2$.

Note that by the definition of $\tilde{\Delta}_m$ and m_{a_k}, we have the inequality:

$$2^{m_{a_k}} = \frac{1}{\tilde{\Delta}_{m_{a_k}}} \leq \frac{4}{\Delta_{a_k}} < \frac{1}{\tilde{\Delta}_{m_{a_k}+1}} = 2^{m_{a_k}+1}. \tag{3.22}$$

This follows from the definition of m_{a_k} and the fact that $\tilde{\Delta}_m = 2^{-m}$.
We consider suboptimal arms in $\mathcal{A}' := \{a_k \in \mathcal{A} \mid \Delta_{a_k} > \lambda\}$ for some fixed $\lambda \geq \sqrt{e/n}$, and analyze the regret in the following cases:
Case (a): Some suboptimal arm a_k is not eliminated in round m_{a_k} (or before) with $a^* \in B_{m_{a_k}}$.
Let us consider an arbitrary suboptimal arm a_k. First note that if

$$\hat{Q}_{a_k}(t) \leq X_{a_k} + \sqrt{\frac{\log(n\tilde{\Delta}^2_{m_{a_k}})}{2T_{m_{a_k}}(t-1)}} \tag{3.23}$$

and

$$\hat{Q}^* \geq X^* - \sqrt{\frac{\log(n\tilde{\Delta}^2_{m_{a_k}})}{2T_{m_{a_k}}(t-1)}} \tag{3.24}$$

hold, then under the assumption that a^* and a_k are both in $B_{m_{a_k}}$, arm a_k will be eliminated in round m_{a_k}.
To see why, consider the elimination phase of round m_{a_k}. By Eq. (3.22), we have:

$$\sqrt{\frac{\log(n\tilde{\Delta}^2_{m_{a_k}})}{2T_{m_{a_k}}(t-1)}} \leq \frac{\tilde{\Delta}_{m_{a_k}}}{2} = \tilde{\Delta}_{m_{a_k}+1} < \frac{\Delta_{a_k}}{4}.$$

Now, using Eq. (3.23) and Eq. (3.24), we get:

$$\begin{aligned}\hat{Q}_{a_k} + \sqrt{\frac{\log(n\tilde{\Delta}^2_{m_{a_k}})}{2T_{m_{a_k}}(t-1)}} \leq X_{a_k} + 2\sqrt{\frac{\log(n\tilde{\Delta}^2_{m_{a_k}})}{2T_{m_{a_k}}(t-1)}} &< X_{a_k} + \Delta_{a_k} - 2\sqrt{\frac{\log(n\tilde{\Delta}^2_{m_{a_k}})}{2T_{m_{a_k}}(t-1)}} \\ &= X^* - 2\sqrt{\frac{\log(n\tilde{\Delta}^2_{m_{a_k}})}{2T_{m_{a_k}}(t-1)}} \leq \hat{Q}^* - \sqrt{\frac{\log(n\tilde{\Delta}^2_{m_{a_k}})}{2T_{m_{a_k}}(t-1)}},\end{aligned}$$

which means that $\hat{Q}_{a_k}$ is strictly less than $\hat{Q}^*$ in round m_{a_k}, leading to the elimination of arm a_k as claimed.
Now, by Hoeffding's inequality (2.3.4), for each $m = 0, 1, 2, \ldots$, we have:

$$\mathbb{P}\left(\hat{Q}_{a_k} > X_{a_k} + \sqrt{\frac{\log(n\tilde{\Delta}^2_m)}{2T_m(t-1)}}\right) \leq \frac{1}{n\tilde{\Delta}^2_m}, \tag{3.25}$$

and

$$\mathbb{P}\left(\hat{Q}^* > X^* - \sqrt{\frac{\log(n\tilde{\Delta}^2_m)}{2T_m(t-1)}}\right) \leq \frac{1}{n\tilde{\Delta}^2_m}. \tag{3.26}$$

These inequalities state that the probability of the estimated action value $\hat{Q}_{a_k}$ being much greater than the actual reward X_{a_k} or the estimated action value $\hat{Q}^*$ being much smaller than the actual reward X^* is exponentially small in $n\tilde{\Delta}_m^2$. Thus, the probability that a suboptimal arm a_k is not eliminated in round m_{a_k} (or before) is bounded by $\frac{2}{n\tilde{\Delta}^2_{m_{a_k}}}$.

Summing up over all arms in $\mathcal{A}'$ and bounding the regret for each arm a_k trivially by $n\Delta_{a_k}$, we obtain by Eq. (3.22) a contribution of:

$$\sum_{a_k \in \mathcal{A}'} \frac{2\Delta_{a_k}}{\tilde{\Delta}^2_{m_{a_k}}} \leq \sum_{a_k \in \mathcal{A}'} \frac{8}{\tilde{\Delta}_{m_{a_k}}} \leq \sum_{a_k \in \mathcal{A}'} \frac{32}{\Delta_{a_k}}$$

to the expected regret. Here, the first inequality comes from the fact that Δ_{a_k} can be upper bounded by a constant factor of $\tilde{\Delta}_{m_{a_k}}$, and the second inequality comes from the definition of $\tilde{\Delta}_{m_{a_k}}$ as $2^{-m_{a_k}}$.

Case (b): For each suboptimal arm a_k: either a_k is eliminated in round m_{a_k} (or before) or $a^* \notin B_{m_{a_k}}$.

Case (b1): If $a^* \in B_{m_{a_k}}$ for all arms a_k in $\mathcal{A}'$, then each arm a_k in $\mathcal{A}'$ is eliminated in round m_{a_k} or before and consequently played not more often than

$$T_{m_{a_k}}(t-1) = \left\lceil \frac{2\log(n\tilde{\Delta}^2_{m_{a_k}})}{\tilde{\Delta}^2_{m_{a_k}}} \right\rceil \leq \left\lceil \frac{32\log(n\Delta^2_{a_k}/4)}{\Delta^2_{a_k}} \right\rceil \tag{3.27}$$

Here, the inequality follows from the definition $\tilde{\Delta}_{m_{a_k}} < \Delta_{a_k}/2$, and hence $1/\tilde{\Delta}^2_{m_{a_k}} \leq 4/\Delta^2_{a_k}$. The contribution to the expected regret is then

$$\sum_{a_k \in \mathcal{A}'} \Delta_{a_k} \left\lceil \frac{32\log(n\Delta^2_{a_k}/4)}{\Delta^2_{a_k}} \right\rceil \overset{\text{linear approx.}}{<} \sum_{a_k \in \mathcal{A}'} \left(\Delta_{a_k} + \frac{32\log(n\Delta^2_{a_k})}{\Delta_{a_k}} \right).$$

Case (b2): Now let us consider the case that arm a^* is eliminated by some arm a_k in some round m^*. First note that if Eq. (3.23) and Eq. (3.24) hold in round $m = m^*$, then the optimal arm will not be eliminated by arm a_k in this round. Indeed, this would only happen if

$$\hat{Q}_{a_k} - \sqrt{\frac{\log(n\tilde{\Delta}^2_{m^*})}{2T_{m^*}(t-1)}} > \hat{Q}^* + \sqrt{\frac{\log(n\tilde{\Delta}^2_{m^*})}{2T_{m^*}(t-1)}},$$

which however leads by Eq. (3.23) and Eq. (3.24) to the contradiction $X_{a_k} > X^*$. Consequently, by Eq. (3.25) and Eq. (3.26) the probability that a^* is eliminated by a fixed suboptimal arm a_k in round m^* is upper bounded by $\frac{2}{n\tilde{\Delta}^2_{m^*}}$.

Now if a^* is eliminated by arm a_k in round m^*, then $a^* \in B_{m_l}$ for all l with $m_l < m^*$. Hence by assumption of Case (b), all arms l with $m_l < m^*$ were eliminated in round m_l or even before. Consequently, a^* can only be eliminated in round m^* by an arm a_k with $m_{a_k} \geq m^*$. Further, the maximal regret per step after eliminating a^* is the maximal Δ_l among the remaining arms l with $m_l \geq m^*$. Hence, taking into account the error probability for elimination of a^* by some suboptimal arm, the contribution to the expected regret in the considered case is upper bounded by

$$\sum_{m^*=0}^{\max_{a_l \in \mathcal{A}'} m_{a_l}} \quad \sum_{a_k \in \mathcal{A}' : m_{a_k} \geq m^*} \frac{2}{n\tilde{\Delta}^2_{m^*}} \cdot n \max_{a_l \in \mathcal{A}' : m_{a_l} \geq m^*} \Delta_{a_l}$$

$$\leq \sum_{m^*=0}^{\max_{a_l \in \mathcal{A}'} m_{a_l}} \quad \sum_{a_k \in \mathcal{A}' : m_{a_k} \geq m^*} \frac{2}{\tilde{\Delta}}^{2}_{m^*} \cdot 4\tilde{\Delta}_{m^*}$$

$$= \sum_{a_k \in \mathcal{A}'} \sum_{m^*=0}^{m_{a_k}} \frac{8}{\tilde{\Delta}_{m^*}} = \sum a_k \in \mathcal{A}' \sum_{m^*=0}^{m_{a_k}} \frac{8}{2^{-m^*}}$$

$$< \sum_{a_k \in \mathcal{A}'} 8 \cdot 2^{m_{a_k}+1} \leq \sum_{k \in \mathcal{A}'} 8 \cdot \frac{8}{\Delta_{a_k}} = \sum_{a_k \in \mathcal{A}'} \frac{64}{\Delta_{a_k}}$$

Here, the steps involve bounding $\max_{l \in \mathcal{A}' : m_l \geq m^*} \Delta_l$ by $4\tilde{\Delta}_{m^*}$ due to the relationship $\Delta_l \leq 2\tilde{\Delta}_{m^*}$ and summing geometric series for simplification.

Finally, summing up the individual contributions to the expected regret of the considered cases, and taking into account suboptimal arms not in $\mathcal{A}'$ gives the claimed bound. □

EUCBV

The EUCBV algorithm combines the strategies from the UCB-Improved algorithm, using their arm elimination strategy proposed by P. Auer et al. in [3] and UCB-V algorithm, incorporating empirical variance estimates to compute their confidence bounds as described by J.-Y. Audibert et al. in [2]. A further, but minor, adaptation of the EUCBV algorithm in comparison to the UCB-Improved algorithm relies on the basic idea of dynamically adapting the exploration term based on the number of plays. This idea was presented in the paper by J.-Y. Liu and Y. Tsuruoka [12], who proposed the CCB algorithm. Due to the simplicity of this idea, it is not our purpose to study the CCB algorithm in detail. Instead, we focus on its incorporation into the EUCBV algorithm, as S. Mukherjee et al. utilized this concept in their approach on the EUCBV algorithm in [14].

Let us now state the algorithm in an adapted version of Algorithm 1 in [14].

Remark 3.2.4. The algorithm incorporates two exploration parameters. The arm elimination parameter ρ regulates the aggressiveness of the elimination of suboptimal arms. A higher value of ρ leads to more aggressive elimination, which can support the convergence to the optimal arm, but also increases the risk of potentially optimal arms being

Algorithm 7 EUCBV Algorithm

```
Data: Bandit model $\nu$, time horizon $n$, exploration parameters $\rho$, $\psi$
Result: Actions $A_1, \ldots, A_n$ and rewards $X_1, \ldots, X_n$
 1: Initialization: Initialize $\Delta_0 := 1$, and $B_0 := \{a_1, \ldots, a_K\}$ a set of arms, $t := 0$,
    $m := 0$, $M = \lfloor \frac{1}{2} \log_2 \frac{T}{e} \rfloor$, $n_0 = \lceil \frac{\log(\psi n \Delta_0^2)}{2\Delta_0} \rceil$ and $N_0 = K n_0$
 2: for $t \le K$ do
 3:     Pull each arm once
 4: for $t = K + 1$ to $n$ do
 5:     Chose arm $a_k \in \arg\max_{a_j \in B_m} \left( \hat{Q}_{a_j} + \sqrt{\frac{\rho(\hat{\sigma}^2_{a_j} + 2)\log(\psi n \Delta_m)}{4T_{a_j}(t-1)}} \right)$
 6:     for each arm $a_k \in B_m$ do Arm elimination:
 7:         if $\hat{Q}_{a_k} + \sqrt{\frac{\rho(\hat{\sigma}^2_{a_k}(T_{a_k}) + 2)\log(\psi n \Delta_m)}{4T_{a_k}(t-1)}} < \max_{l \in B_m} \left( \hat{Q}_{a_l} - \sqrt{\frac{\rho(\hat{\sigma}^2_{a_l} + 2)\log(\psi n \Delta_m)}{4T_{a_l}(t-1)}} \right)$ then
 8:             Delete arm $a_l$ from $B_m$
 9:     if $t \ge N_m$ and $m \le M$ then Reset Parameters
10:         $\Delta_{m+1} := \frac{\Delta_m}{2}$
11:         $B_{m+1} := B_m$
12:         $n_{m+1} := \lceil \frac{\log(\psi n \Delta^2_{m+1})}{2\Delta_{m+1}} \rceil$
13:         $m \leftarrow m + 1$
14:         $t \leftarrow t + 1$
15: Stop if $|B_m| = 1$ and chose $a_k \in B_m$ until $n$ is reached
```

discarded prematurely. The exploration regulatory factor ψ controls the trade-off between exploration and exploitation through adjusting the degree of uncertainty considered in the arm selection process. By affecting the weighting of the time horizon n in the logarithmic component of the upper confidence bound, it can improve the balance between exploration and exploitation by adjusting the elimination of suboptimal arms in dependence on the time horizon.

Remark 3.2.5. We will restrict our attention on model-dependent bound relying on the reward gap. S. Mukherjee et al. provided a model-independet bound in [14], but since the model-dependent bound already gives the main ideas in its proof, we will not state it here. A complete proof of the model-independent bound can be found in Appendix 7.7. of their paper [14].

The remaining section is devoted to an introducting into the study of the proof ida of the regret bound of the EUCBV algorithm. However, since the proof combined elements of the Proof of the regret bound of the UCB-V Algorithm 3.1.7 as well as the Proof of the regret bound of the UCB-Improved Algorithm 3.2.2, it is not worth pointing out all the details, but rather the overall structure in a proof sketch.

Theorem 3.2.6. Regret bound of the EUCBV algorithm (Compare [14], Theorem 1)
For $n \geq K^{2.4}$, $\rho = \frac{1}{2}$ and $\psi = \frac{n}{K^2}$, the regret R_n for EUCBV satisfies

$$R_n \leq \sum_{a_k \in \mathcal{A}: \Delta_{a_k} > \lambda} \left(\frac{C_0 K^4}{n^{\frac{1}{4}}} + \left(\Delta_{a_k} + \frac{320 \sigma_{a_k}^2 \log \left(\frac{n \Delta_{a_k}^2}{K} \right)}{\Delta_{a_k}} \right) \right) + \sum_{a_k \in \mathcal{A}: 0 < \Delta_{a_k} \leq \lambda} C_2 K^4 n^{\frac{1}{4}} + \max_{a_k \in \mathcal{A}: 0 < \Delta_{a_k} \leq \lambda} \Delta_{a_k} n$$

for all $\lambda \geq \sqrt{\frac{e}{n}}$ and C_0, C_2 are integer constants.

Outline of the Proof of the regret bound of the EUCBV algorithm. (Compare [14], Outline of the Proof of Theorem 1)
The proof follows the technique of the proof of the regret bound of the UCB-Improved algorithm by P. Auer et al. in their paper [3] and consists of three modules. In the first module, we prove the necessary conditions for arm elimination within a specified number of rounds. Additional technical results are required to bound the length of the confidence intervals. The EUCBV algorithm combines the variance-estimate based approach by J.-Y. Audibert et al. in [2] with the arm-elimination technique of P. Auer et al. in [3]. We use Bernstein inequality (2.3.5) instead of Hoeffding's inequality (2.3.4) as applied in the Proof of the regret bound of the UCB-Improved Algorithm 3.2.2 to derive the regret bound. In the proof, we bound the probability of non-uniform arm selection before elimination. In the second module, we bound the number of pulls required if an arm is eliminated on or

before a particular number of rounds. The number of pulls allocated in a round m for each arm is $n_m := \left\lceil \frac{\log(\psi n \Delta_m^2)}{2\Delta_m} \right\rceil$, which is lower than the one of UCB-Improved. Lemma 3.2.7 introduces the variance term in the most significant term of the bound. The third module deals with bounding the regret when a suboptimal arm eliminates the optimal arm. □

Lemma 3.2.7. (Compare [14], Lemma 6)
For two integer constants c_1 and c_2, if $20c_1 \leq c_2$, then,

$$c_1 \frac{4\sigma_{a_k}^2 + 4}{\Delta_{a_k}} \log\left(\frac{n\Delta_{a_k}^2}{K}\right) \leq c_2 \frac{\sigma_{a_k}^2}{\Delta_{a_k}} \log\left(\frac{n\Delta_{a_k}^2}{K}\right).$$

Proof. (Compare [14], Proof 7) We again prove this by contradiction. Suppose,

$$c_1 \frac{4\sigma_{a_k}^2 + 4}{\Delta_{a_k}} \log\left(\frac{n\Delta_{a_k}^2}{K}\right) > c_2 \frac{\sigma_{a_k}^2}{\Delta_{a_k}} \log\left(\frac{n\Delta_{a_k}^2}{K}\right).$$

Canceling out the common terms $\frac{1}{\Delta_{a_k}} \log\left(\frac{n\Delta_{a_k}^2}{K}\right)$ on both sides and further reducing the above two terms we can show that,

$$\begin{aligned}
&4c_1\sigma_{a_k}^2 + 4c_1 > c_2\sigma_{a_k}^2 \\
&\Rightarrow 4c_1 . \frac{1}{4} + 4c_1 \overset{0 \leq \sigma_{a_k}^2 \leq \frac{1}{4}, \in \mathcal{A}}{>} \frac{c_2}{4} \\
&\Rightarrow 20c_1 > c_2.
\end{aligned}$$

But, we already know that $20c_1 \leq c_2$. Hence,

$$c_1 \frac{4\sigma_{a_k}^2 + 4}{\Delta_{a_k}} \log\left(\frac{n\Delta_{a_k}^2}{K}\right) \leq c_2 \frac{\sigma_{a_k}^2}{\Delta_{a_k}} \log\left(\frac{n\Delta_{a_k}^2}{K}\right).$$

□

In this section, we have extensively discussed the UCB-Improved algorithm and provided a detailed proof of its regret bound. We also introduced the EUCBV algorithm, which represents an extension and combination of the UCB-Improved and UCB-V algorithms. The improvement in EUCBV is the incorporation of variance estimates, as derived from UCB-V. The proof of the regret bound for EUCBV follows a similar structure to that of UCB-Improved, with the added complexity of incorporating variance estimates into the analysis.

4 Simulation and Empirical Analysis

In most of the papers that introduced the bandit algorithms presented in Section 2 and 3, there is an experimental section in which these algorithms are simulated. Typically, the algorithm presented in the paper and its underlying algorithm on which its idea is based are implemented in such a simulation. In [2] for example, J.-Y Audibert et al. aimed to illustrate the obtained bounds achieved by the UCB-V algorithm by comparing it with its predecessor, the UCB algorithm, addressing multiple aspects of performance analysis. Another example is [14], where S. Mukherjee et al. compared the UCB-V, EUCBV, UCB-Improved and six other algorithms on 20 Bernoulli distributed arms in Section 5. Although this comparison is quite extensive, it is already one of the most comprehensive ones available.

Due to the gap of an available comparison of all of our algorithms presented in Section 2 and 3, our goal for this chapter is to provide a simulation of all these algorithms and allow the reader to recreate the simulation. We want to answer our research question of how variance-aware bandit algorithms and non-variance-aware algorithms compare in an empirical analysis in terms of their performance.

First, we will present related work and search for comparable studies that have already addressed this research gap. We will then provide detailed information on our simulation, including the requirements for a comprehensive implementation of the algorithms, as well as the necessary diagrams, variables and options for analysing algorithm performance in edge cases. Then, the implementation of the algorithms, providing an overview of the chosen parameters in each algorithm and their integration into the dashboard of the simulations, allowing a thorough evaluation of their performance. This simulation technique uses Bernoulli random variables with payoffs that are close together to reveal potential weaknesses in algorithmics' performance. The dashboards of our implemented algorithms and data are available on GitHub. Finally, we will present the results of the implementation and a comparison between these algorithms to make a statement about the performance of variance-aware adaptations of bandit algorithms and address threats to validity.

4.1 Research Question and Research Gap

In this section, we will pose the research question of our thesis in the context of the gap in current research, attempting to close it in the further sections of this chapter. Despite numerous studies introducing various stochastic bandit algorithms, there is currently no comprehensive comparison of all of the algorithms considered in this thesis. This section will review the literature to identify where these gaps exist and set the stage for the subsequent analysis. The objective of Section 4 is to answer the question of this thesis, how variance-aware bandit algorithms compare with non-variance-aware algorithms in terms of performance and their exploration-exploitation trade-off. Through an empirical analysis in this part of our thesis we want to compare both types of stochastic bandit algorithms, either incorporating variance estimates or not.

Throughout our literature search, we searched for related papers that address our research question or simulate variance-aware algorithms in comparison to non-variance-aware algorithms. Despite an extensive search in databases such as SCOPUS, ScienceDirect, Google Scholar, and Arxiv, as well as the digital library IEEE Explore, we could not find any work dealing with the simulation, empirical analysis or even a general evaluation of variance-aware bandit algorithms in comparison. Due to the lack of existing literature, we instead compared the experimental sections of [4], [3], [1] and [14] in terms of their performance evaluation of the implemented algorithms.

P. Auer et al. proposed in Section 4 of [4] a comparison of the empirical behavior of the UCB-Tuned, UCB2, and ε-Greedy algorithm. UCB2 is an algorithm that we did not consider in our theoretical analysis due to our focus on variance-aware algorithms, which UCB2 is not. In their experiment, they implement various bandit problems with two to 10-armed bandits with different Bernoulli distributed rewards. They start with a relatively simple reward distribution of the two-armed bandit with reward expectations of $[0.9, 0.8]$ for the first and second arm and later increased the difficulty of the ten-armed bandit up to the point where only one arm had a reward expectation of 0.55 and the other nine arms had a reward expectation of 0.45. They recorded their results over 100 rounds of 100 000 plays, comparing the proportion of the best machine played and the regret of the different algorithms. The tuning parameter ε of the ε-Greedy algorithm was set to $0.05, 0.10$, and 0.15 for different runs. Although P. Auer et al. [4] already proposed an empirical analysis of various parameters and choices of the bandit problem, this was the first paper to propose the UCB algorithm and therefore does not compare currently available MAB algorithms. Furthermore, they do not provide any information about the reproducibility of their simulation.

The following paper which took [4] as a reference and developed the UCB-V algorithm based on it, is by J.-Y. Audibert et al. [2]. In Section 6 of their paper, they conducted experiments to demonstrate the obtained bounds of the tails. They used a two-armed bandit with Bernoulli distributed arms with expected payoffs of 0.5 and 0.48 and a version with 0.5 and 0.495 as payoffs to prove the effectiveness of the UCB-V, especially in cases where the expected payoffs of the two arms are close to each other. They compared the UCB algorithm with the UCB-V algorithm through calculating and simulating the Value at Risk (VaR) function for the first $T = 2^{20} \approx 1,000,000$ steps. Although they proposed the PAC-UCB algorithm, they did not include it in their simulation.

A different approach to improve the UCB algorithm of [4] was proposed by P. Auer et al. in [3]. Their UCB-Improved algorithm follows the approach of eliminating arms for which the confidence intervals are not small enough. In their paper, they prove the regret bound but do not provide a simulation of the UCB-Improved algorithm. Only in [14], which proposed the EUCBV algorithm as a mix of the UCB-V and the UCB-Improved algorithm, the UCB-Improved algorithm was empirically evaluated together with EUCBV, UCB-V, UCB and six other algorithms. This is by far the most extensive simulation in terms of the number of algorithms. It was carried out in four different experiments. In the first, 20 Bernoulli distributed arms were used, while the other three were based on different Gaussian distributed arms to show that the EUCBV performs best with a large time horizon, a large number of arms and a large variance of the optimal arm in a long time horizon. Despite testing several scenarios, no adjustments were made within each of the four cases. Furthermore, no information on replication could be found.

After analyzing the related work, to our knowledge, there is no simulation or empirical analysis of MAB algorithms that allows a comprehensive comparison between variance-aware and non-variance-aware algorithms in terms of their performance in different scenarios. Due to this research gap, we decided to create such a simulation.

4.2 Simulation Techniques

In this section, we will describe the design of our the MAB simulation, the variables involved, and the necessary diagrams for a comprehensive comparison. We will discuss the simulation environment and the metrics used to evaluate performance.

Bandit model

In Section 4.1, the most common choice for a MAB model is the two-armed bandit model with Bernoulli distributed arms. This is obviously the simplest choice, but with the right design, it can be quite insightful and allows for variation, enabling us to observe differences in algorithm performance. We, therefore, restricted our focus to this specific bandit problem for our implementation.

As mentioned before, we will simulate all algorithms from Section 2 and 3. For eight of these nine algorithms, Bernoulli random variables are the most obvious choice for arm rewards. For the UCB-Normal algorithm, however, we take a different approach, as it is designed for normally distributed payoffs. For this algorithm, we designed a two-armed bandit model with Gaussian distributed rewards having the same means as in the Bernoulli setting and a variance comparable to those of the Bernoulli settings.

The bandit model for the Bernoulli random variables employed in the simulations consisted of two arms with different scenarios of expected rewards, generating rewards of either 0 or 1. In the first scenario, the optimal arm has an expected reward of 0.9, and the suboptimal arm has an expected reward of 0.8. The UCB-Normal algorithm is simulated with two normally distributed arms with means 0.9 and 0.8 and corresponding variances calculated as follows: for the optimal arm, the variance is $p(1-p) = 0.9 \times 0.1 = 0.09$, and for the suboptimal arm, it is $p(1-p) = 0.8 \times 0.2 = 0.16$. This scenario represents a relatively easy case.

To create more challenging conditions for the algorithms, we propose two additional cases. The second case focuses on narrowing the expected payoffs of the arms by adjusting the mean rewards to 0.9 for the optimal and 0.895 for the suboptimal arm. It is well-documented that UCB algorithms often struggle in scenarios where suboptimal arms are sampled frequently, as the UCB function for suboptimal arms does not significantly differ from that of the optimal arm, thus highlighting the exploration-exploitation trade-off. For the UCB-Normal algorithm, we chose means of 0.9 and 0.895 with variances of 0.09 for the optimal and 0.093975 for the suboptimal arm.

The third and final case was specifically designed to make the overall decision and identification of the optimal arm even harder by reducing the total reward. Therefore, we used mean rewards of 0.5 for the optimal and 0.495 for the suboptimal arm of the Bernoulli-distributed arms. Consequently, UCB-Normal was simulated with normally distributed rewards with means 0.5 and 0.495 and variances of 0.25 for the optimal and 0.2475 for the suboptimal arm.

In a two-armed bandit setting, some algorithms might quickly identify the optimal arm after the first two pulls. For example, the ETC algorithm will draw both arms in the first two time steps and repeat the selection of this procedure a chosen number of times m. If the rewards are different after choosing each arm ones, e.g., 1 for the optimal and 0 for the suboptimal one, the algorithm will focus on the optimal arm from the next time step onwards, giving it an unfair advantage over other algorithms that might not test each arm a certain number of times by default due to the fact, that it is a superior strategy only in the case, where there is time left to commit to the optimal arm after the exploration phase.

On the other hand due to the design of the implemented argmax function on the array of the means of the arm, corresponding to the probability of choosing them in a Bernoulli bandit model, $[a_{optimal}, a_{suboptimal}]$, which is designed to automatically draw the optimal arm, a bias could occur, if we do not give the reader an option to decide on the position of the optimal arm in the array. Then, the function always draws the first index of the array, which can lead to a bias if the optimal arm is not in the first position. Furthermore, for the ε-Greedy algorithm, the order of the arms matters as well. If the algorithm is in the exploitation phase during the first decision, it will exploit the arm with the highest value. Initially, both arms have a value of 0, and the argmax function chooses the first entry by default, which may bias our analysis. To solve this problem, for all three cases of Bernoulli variables, we have included the option to choose the arm that is in the first of the two positions of the corresponding array.

Comparison of the algorithms

In our analysis, we utilize two versions of the simulated data for each algorithm. The first version includes data for 100 rounds, collected over 100,000 iterations, and the second version is the average of these data points. For the 100 rounds, we record the following outputs.

During each time step, the algorithm selects the arm as the algorithms definition instructs. The reward for the selected arm is then generated based on a Bernoulli distribution with the corresponding expected value. This selection and reward generation process involves several calculations. For instance, the arm with the highest estimated mean reward is identified using the argmax function on the array of the arm means. The reward for each arm is then sampled from a Bernoulli distribution, designed trough a Binomial distribution with $n = 1$ for all arms, and the count of pulls for the selected arm is incremented. The total reward, described as “Total Reward”, is updated by adding the sampled reward.

The regret is calculated as the difference between the expected reward of the optimal arm and the expected reward of the selected arm. This value is added to the “Total Regret” in each round to make it comparable throughout the algorithms. In the end, we round the “Total Regret” to two decimal places after the comma, since otherwise due to numerical issues, the regret would have irrelevant deviations of the exact total regret. If a suboptimal arm is selected, the “Suboptimal Arms Count” is incremented. We regard this variable as needed, since for arms that have rewards close together, it is interesting to know not only the overall performance of the algorithm through the total reward, but also the influence of the suboptimal arm on the results. Depending on whether the reward is zero or one, the “Zeros Count” or “Ones Count” is updated accordingly.

For the average calculations, we compute the mean values across all 100 iterations. The “Average Total Reward” is obtained by averaging the “Total Reward” over all iterations for each time step. Similarly, “Average Suboptimal Arms” is the mean of “Suboptimal Arms”, “Average Regret” is the mean of “Total Regret”, “Average Zeros Count” is the mean of “Zeros Count”, and “Average Ones Count” is the mean of “Ones Count” across all iterations at each time step.

We decided to employ the following six diagrams in our dashboard for a comparison of all of the nine algorithms. Figure 1 to 3 and Figure 5 as well as 6 in the dashboard rely on the averaged data of the algorithms over 100 iteration. Figure 4 presents the data of the 100 iterations as a whole for one time step.

The first figure in the dashboard displays the average total reward over time for all nine algorithms. This figure provides insight into how effectively each algorithm maximizes rewards over time.

The second figure illustrates the average regret over time, to understand how well each algorithm minimizes regret over time, providing a complementary view to the first figure.

The third figure is a histogram showing the distribution of zero and one rewards for each algorithm. On the x-axis, positions 0 and 1 represent the counts of zero and one rewards, respectively. Each algorithm has two bars: one indicating the “Average Zeros Count” and one indicating the “Average Ones Count”. The height of these bars reflects the average counts. This figure helps analyze the reward distribution, highlighting the frequency of low and high rewards for each algorithm, which is especially interesting in the last case of the simulation where the mean rewards are close together.

The fourth figure focuses on the distribution of total regret at time step 100,000 for one specific algorithm defined through a chosen case in a selection box beforehand. For the selected algorithm, the total regret values at time step 100,000 across 100 iterations are plotted as a histogram. The histogram displays the frequency distribution of these values, ideally approximating a normal distribution due to the Law of Large Numbers. This figure offers an understanding of the variability and consistency of each algorithm’s performance.

The fifth figure presents the VaR function for α-values 0.01, 0.05, and 0.1. A selection box allows users to choose the desired α-value, influencing the underlying calculation of the VaR function. The VaR function indicates the maximum potential loss at a given confidence level. This figure provides a quantifiable measure of risk associated with each algorithm.

The sixth figure and therefore last figure of the dashboard depicts the proportion of suboptimal arms pulled up to each time step in comparison to all arms pulled. This proportion is calculated by dividing the “Average Suboptimal Arms” by the “Timestep”. This figure highlights the exploration-exploitation trade-off, showing how frequently suboptimal choices are made throughout the simulation.

4.3 Implementation of the Simulation

Here, we describe the nine algorithms implemented in our code, including any special considerations and extending them by an average over the results of the variance-aware algorithms as well as the non-variance-aware algorithms. This section also provides an overview of the dashboard we built, explaining how to navigate it and select different parameters. We will document the implementation process, the tuning parameters chosen, and how the simulation results are visualized in the dashboard.

To conduct a comprehensive evaluation of the implemented bandit algorithms, simulations were performed utilizing Python, a widely-used programming language known for its simplicity and utility in Machine Learning. Object-oriented programming principles were employed to ensure modularity and scalability across different algorithm implementations. The simulations and their corresponding visualizations were made available in a dedicated *GitHub Repository* (https://github.com/eelisee/varianceinucbalgorithms), allowing the reader to replicate the results.

As described in Section 4.2, we created data for three different cases of bandit models, each with two versions that change the order of the arms in the array in which they were defined. Furthermore, average calculations were made for each algorithm and case in a separate dataset. This results in twelve datasets for each of the nine algorithms. The algorithms are prefixed with a number indicating their introduction order in this thesis. The CSV files of the results are named “results” for the results of all 100 iterations and ‘average_results‘” for the results of the average calculation over all 100 iterations. Furthermore, the order of the optimal and suboptimal arms is indicated in the file name by “opt” or “subopt” for each, denoting the optimal arm or the suboptimal arm in the first position of the array. The case of the Bernoulli model is indicated with “ver$X$”, where X represents the version number. We define the cases as follows:

ver1: Bernoulli distributed rewards with an arm mean of 0.9 for the optimal and 0.8 for the suboptimal arm.

ver2: Bernoulli distributed rewards with an arm mean of 0.9 for the optimal and 0.895 for the suboptimal arm.

ver3: Bernoulli distributed rewards with an arm mean of 0.5 for the optimal and 0.495 for the suboptimal arm.

For the UCB-Normal algorithm we sample from a Gaussian distribution according to the values of the mean and variance described above.

Calculating the average results of the variance-aware algorithms, UCB-Tuned, UCB-V and EUCBV, as well as the average results of the non-variance aware algorithms, ETC, ε-Greedy, UCB, PAC-UCB and UCB-Improved, gives us additional datasets of the same format as the individual algorithms. UCB-Normal has been excluded due to its uniqueness by sampling from a Gaussian distribution to prevent biases. This averaged data on the criteria of the variance awareness facilitates the subsequent comparison of the empirical analysis to answer our research question.

Implementation of the Algorithms

The ETC algorithm (1_ETC) is defined as in the pseudocode, being split into a first part of exploration, that is exactly as long as the number of arms times a chosen parameter m, which we set to 100, thus drawing each arm 100 times before committing to the best arm up until that point in the second part. In the case where both arms draw the same reward in the exploration phase, the algorithm will afterwards choose the arm whose mean is stated in the first position of the array of the arm means.

The ε-Greedy algorithm (2_Greedy) is also defined as in the pseudocode. We set the exploration parameter ε to 0.05, since we have a long time horizon on which the ε can be selected and we do not want it to have an unfair advantage over other algorithms by making it too large to explore comparably much in the beginning.

The UCB algorithm (3_UCB) is designed by two different functions. The UCB_simulation function initializes the counts and implements the select_arm function, which defines our UCB function to calculate the UCB values for each arm and decides which arm to choose.

The UCB-Normal algorithm (4_UCB-Normal) draws rewards from a different distribution than the other algorithms, as mentioned in Section 4.2. It draws the reward from a normal distribution with the means of the arms of the Bernoulli distribution and the corresponding variances calculated in Section 4.2 and initialized in the array arm_variances.

We implemented UCB-Tuned (5_UCB-Tuned) as our UCB algorithm but adapted the exploration bonus of the UCB function as mentioned in Section 3.1.

The UCB-V algorithm (6_UCB-V) is implemented as in Algorithm 5. We need to choose three parameters: θ, c, and b. For our choice of $\theta = 1$, $c = 1$, and $b = 1$, we refer to the paper by J.-Y. Audibert et al. [2]. As mentioned in Section 3.1, a typical choice of $\varepsilon_{s,t}$ is $\varepsilon_{s,t} = \theta \log(t)$. In Theorem 5 of their paper, J.-Y. Audibert et al. prove that for any choice of c, if $\theta < 1$, the algorithm suffers polynomial regret, leaving us with the choice for $\theta \geq 1$. We, therefore, decided on choosing $\theta = 1$, as a worst-case scenario. Furthermore, they conclude from Theorem 6 of their paper that for a choice of $\theta = 1$, the minimal value of c to ensure logarithmic regret is $c = 1$. Since we draw rewards from a bandit model with Bernoulli distributed rewards of 0 and 1, we choose $b = 1$ as the upper bound of our reward range.

PAC-UCB (7_PAC-UCB) is the UCB-V adaptation for which the exploration function $\varepsilon_{s,t} = \varepsilon_s$ does not depend on t. As mentioned in Section 3.1, a typical choice for an exploration function is $\varepsilon_s = \log(Ks^q\beta^{-1}) \vee 2$, which we, therefore, implemented instead of the chosen exploration function of the UCB-V algorithm. We again set the parameters b and c to 1 as in the UCB-V algorithm, leaving us with the choice of how to set q and β. Since β can be interpreted as the value α commonly set to 0.05 to decide whether results are statistically significant in statistics, we set $\beta = 0.05$. To ease the calculation, we chose $q = 1.3$ to have an influence of s, the number of times an arm has been chosen, but keeping it in a moderate range.

The UCB-Improved algorithm (8_UCB-Improved) is the precursor of the EUCBV algorithm. In this algorithm, we do not have to set tuning parameters since the parameter $\tilde{\Delta}$ is adjusted by halving it after deleting the potentially suboptimal arms. We modified the UCB-Improved algorithm by adjusting the number of pulls per time step. In all of our other algorithms, only one arm gets chosen per time step. In the UCB-Improved algorithm, as originally proposed by P. Auer et al. in [3], there can be as many arm draws as the set B_t includes arms. This would be a different interpretation of the time steps since more decisions per time step can be made in comparison to all of our other algorithms. Therefore, we implemented another count that would still hold the proposed bounds of the first and second phases but restricts the choice of arms to one per time step.

The last algorithm is the EUCBV algorithm (9_EUCBV). While being a combination of the ideas of the UCB-V algorithm and the UCB-Improved algorithm, it has two tuning parameters. The arm elimination parameter ρ is set to $\frac{1}{2}$ and the exploration regulatory factor ψ is chosen as $\frac{n}{K^2}$ in the definition of the reward gap in Theorem 3.2.6. This algorithm is designed to only choose one arm at a time step, therefore implementing it as in Algorithm 7.

Implementation of the Dashboard

We developed an interactive dashboard named "Simulation of Variance-aware Algorithms for Stochastic Bandit Problems" for the visualization of the results of the nine algorithms described above. In Section 4.2, we already stated the necessary options and requirements to create a comprehensive evaluation and comparison between variance-aware and non-variance-aware algorithms. After creating our data, we decided to build a dashboard that gives the user easy access to the relevant plots and allows them to adjust the visualization as well as the conditions for the variables so that the reader is able to analyze the performance of the algorithms based on various key features of performance evaluation. In the following, the technical details of the dashboard's implementation, its interactive features, and how it aligns with the simulation techniques discussed earlier are described. The objective of this section is to give the reader all information needed to replicate and even extend the simulation.

Since a commonly used tool to visualize such data is Plotly, we decided to implement the simulation through the Dash library that Plotly provides. The code of the dashboard can be found in the "dashboard.py"-file, and the implementation can be executed by running the "dashboard.py"-file locally, which will start a server accessible via 127.0.0.1:8050. For additional details that are not mentioned in this section on the simulation setup and code implementation, please refer to the GitHub repository.

The dashboard interface is divided into two sections: a left sidebar for user input and configuration, and a right section for displaying the resulting figures in the dashboard. This design separates the settings from the actual visualizations, maintaining the possibility for the user to easily modify parameters and immediately visualize the effects on the algorithms' performance. For a graphical representation of the standard setting please refer to Figure 1 in Appendix A.

The left sidebar, titled Settings, contains several dropdown menus that allow users to configure the simulation parameters. The first dropdown menu, "Distribution of Arms", allows the user to select different reward distributions for the arms, offering three different cases to choose from. The second menu, "Order of Arms", specifies the order in which the arm means are listed in the corresponding array, with options for either starting with the optimal arm or the suboptimal arm. This allows the user to visualize the effects of the position of the optimal arm in the array in case the algorithms depend on the order of drawing one arm before another. Hereby, (arm_a, arm_b) refers to the position of the arm means in the corresponding array, where arm_a is in the first place and arm_b is in the second place. The third dropdown, "α", sets the significance level for the VaR calculations. Finally, the "Algorithm for Fig. 4" menu allows the user to select which algorithm's data to display in Figure 4 of the dashboard, with options including all nine algorithms. Below these configuration options, a legend section lists the algorithms and their corresponding colors used in the figures, providing clarity and coherence in data representation. As indicated in this legend, there are two further graphs added, one for the average results of the variance-aware and one for the average results of the non-variance-aware algorithms. When running the code for the first time, the standard parameters are set to a "Distribution of Arms" with arm means 0.9 for the optimal arm and 0.8 for the suboptimal arm. Secondly, in the initial setting, the "Order of Arms" is (optimal arm, suboptimal arm), meaning that the optimal arm is in the first place of the array. Then, the α-value is automatically set to 0.05 at the beginning. And the last dropdown menu, which sets the underlying algorithm for the fourth figure, is set to the UCB algorithm, serving as a baseline for later comparisons of advanced variance-aware algorithms.

The right section of the dashboard contains six figures, each providing a different perspective on the performance of the selected algorithms. These are implemented with respect to the plots described in Section 4.2.

The implementation of the dashboard begins with the initialization of the dashboard in its two parts. The app's layout is defined using HTML and Dash core components, including the sidebar for user inputs and the main section for displaying each plot in a figure. The primary data loading function "load_data" reads the CSV files containing the simulation results for each algorithm. This function loads two types of files for each algorithm as well as the files for the average results over the variance-aware and non-variance-aware algorithms and choice of case as well as order of arms: the detailed results of the 100 iterations and the average results, which are then structured for efficient handling within the dashboard. The core functionality of the dashboard is driven by a callback function that updates the plots based on user input. This function reloads the data for all algorithms whenever a configuration parameter of the dropdown menu is changed, so that the displayed plots always reflect the current settings. The callback function processes the selected options, retrieves the corresponding data, and updates each plot with the latest data. This updating mechanism is the basis for creating an interactive and responsive dashboard, allowing users to explore the impact of different parameters on the performance of the variance-aware algorithms.

4.4 Exploration-Exploitation Trade-Off

In this section, we present and analyze the results from the simulation of various algorithms. Due to the extensive range of variants and combinations for each of the six figure, we focus on the most interesting combinations per figure. This selection particularly emphasizes the comparison between variance-aware and non-variance-aware algorithms to highlight differences and similarities in performance.

Analysis of the results

We offer a brief description and detailed analysis of each figure. For graphical representations, please refer to the appendix A, where the respective figures utilized in this evaluation of the results are provided.

Figure 1 in the dashboard displays the average total reward over time steps for all nine algorithms. The most notable differences occur in the scenario where the optimal arm has a mean reward of 0.9 and the suboptimal arm has a mean reward of 0.8, with the suboptimal arm listed first in the array. In this scenario, the UCB-Improved algorithm performs significantly worse than the others. While most algorithms achieve a similar performance, fluctuating around 89,990 for UCB-Tuned to 89,720 for ε-Greedy at time step 100,000, UCB-Improved only reaches about 86,850. This discrepancy of nearly 3,000 in reward is substantial. The primary reason for the poor performance of UCB-Improved algorithm, as already predicted in Remark 3.2.1, is its implementation, which deviates from the algorithm's original definition in the paper of P. Auer et al. [3] and our implementation in the Python code. Specifically, a running variable was required to count time steps, as UCB-Improved often performed more steps per time step in the arm selection phase. This adjustment was necessary for a fair comparison with other algorithms but resulted in a distorted performance. Consequently, the average result of all non-variance-aware algorithms is lowered, given that UCB-Improved is part of this group and influences the calculation of the average. It is particularly notable that variance-aware algorithms outperform non-variance-aware ones. The average result of variance-aware algorithms exceeds even the best-performing non-variance-aware algorithm, ETC.

Figure 2 in the dashboard serves as a complement to Figure 1 by illustrating the average total regret over time steps. The regret is calculated as the cumulated difference between the mean reward of the optimal arm and the suboptimal arm. This figure highlights how effectively each algorithm minimizes regret over time. In the initial time steps, the performance of all algorithms is quite similar, especially in the scenario with mean rewards of 0.9 and 0.8, where the optimal arm is in the first place. However, as time progresses, differences become more pronounced. Again, UCB-Improved exhibits the highest regret, consistent with its low total reward observed in Figure 1. This confirms the implementation issues identified earlier. The average total regret for variance-aware algorithms is significantly lower, with values around 89 at time step 100,000, excluding UCB-Normal, which operates differently due to its Gaussian reward distribution. The average regret for non-variance-aware algorithms is noticeably higher, largely due to the outlier performance of UCB-Improved. When examining other reward distributions, such as mean rewards of 0.9 and 0.895, the algorithms face greater challenges, with regrets being nearly four times higher for UCB-Tuned. In the case of mean rewards of 0.5 and 0.495, the regrets for all algorithms, except UCB-Improved, are more closely clustered, ranging from about 100 to 210. This reduced spread compared to the scenario with 0.9 and 0.8 rewards, which had a range from 25 to 250, indicates a general difficulty in distinguishing between arms when the mean rewards are very close.

Figure 3 illustrates the average counts of zeros and ones for each algorithm. This figure highlights how each algorithm performs in terms of selecting arms that yield either zero or one rewards. For the cases with reward distributions of (0.9, 0.8) and (0.9, 0.895), the UCB-Normal algorithm stands out. Here, the UCB-Normal algorithm exhibits a split in roughly two halves in both bars, deviating from the other algorithms' behavior. This deviation occurs because the implementation of the UCB-Normal algorithm uses normal distributions for rewards, which aligns with the Law of Large Numbers, converging towards the expected value over time, thus showing a balanced count around the mean of each arm. Note that "Zeros" and "Ones" do not refer to the number of zeros and ones drawn from the distribution as in the other algorithms, but instead in the UCB-Normal algorithm refer to the fact that the rewards drawn are greater than or equal to the mean reward of the corresponding arm, which corresponds to the number of "Ones", or less than the mean reward of the corresponding arm, which corresponds to the "Zeros". In contrast, for the average counts of zeros and ones, variance-aware algorithms generally draw fewer zeros compared to non-variance-aware algorithms. For example, in the case of the (0.9, 0.8) reward distribution, variance-aware algorithms have an average of 10.04 zeros and 89.96 ones, while non-variance-aware algorithms average 10.70 zeros and 89.30 ones. This trend suggests that variance-aware algorithms are slightly better at selecting the more rewarding arms. As the reward means become closer, such as in the (0.9, 0.895) distribution, the averages of zeros and ones for variance-aware and non-variance-aware algorithms converge. When the reward means are very close, as in the (0.5, 0.495) distribution, the counts of ones and zeros for both sets of algorithms nearly equalize, reflecting the diminishing advantage of variance estimates when arm rewards are almost identical.

Figure 4 displays the distribution of total regret at time step 100,000 for each algorithm. This figure helps understand the variability and consistency in each algorithm's performance over multiple iterations. ETC and ε-Greedy show a bimodal distribution for arm means 0.9 for the optimal arm and 0.895 for the suboptimal arm, indicating that they either perform very well or very poorly, with less middle-ground performance. This behavior is logical given that, once these algorithms commit to an arm, the regret heavily depends on the early exploration phase. Algorithms based on UCB algorithm exhibit a more normally distributed total regret, which aligns with their mechanism of confidence interval-based decision-making, providing more consistent performance. Non-variance-aware and variance-aware averages are excluded from this comparison as they are not calculated over the 100 iterations.

Figure 5 illustrates the VaR function for various α-values across all algorithms. VaR measures the maximum potential loss at a given confidence level, offering insight into the risk associated with each algorithm. The UCB-Improved algorithm consistently exhibits the highest VaR due to its definition. For an α-value of 0.01 and arm means of 0.9 for the optimal arm in the first place and 0.8 for the suboptimal arm, UCB-Normal has the highest VaR at 296.74 for 100,000 timesteps if UCB-Improved is excluded from this analysis, whereas UCB-Tuned has the lowest at 50.18. This spread indicates significant differences in risk profiles among the non-variance-aware algorithms. The remaining algorithms fall into two groups: those with VaR values below 138 and those above 247, without a clear separation between variance-aware and non-variance-aware algorithms.

Figure 6 shows the proportion of suboptimal arms pulled relative to the total number of pulls over time. This figure emphasizes the exploration-exploitation trade-off of each algorithm. For the reward distribution (0.9, 0.8), the proportion of suboptimal arms pulled decreases quickly for all algorithms, with UCB-Improved being an exception. By time step 10,000, most algorithms have reduced this proportion to below 0.12, with variance-aware algorithms generally achieving lower proportions earlier. By time step 100,000, the variance-aware and non-variance-aware algorithms converge, with all proportions below 0.25. Variance-aware algorithms, except for UCB-Normal, achieve proportions as low as 0.008. In the (0.9, 0.895) reward distribution, differences between algorithms are less pronounced. ETC and ε-Greedy perform particularly well in this scenario, highlighting their adaptability in closer reward scenarios. In the challenging case of (0.5, 0.495), variance-aware algorithms significantly outperform non-variance-aware ones. Here, UCB-Tuned achieves a proportion of 0.25, whereas the best non-variance-aware algorithm, ε-Greedy, only manages 0.32. This result underscores the superior ability of variance-aware algorithms to identify suboptimal arms in closely matched reward scenarios.

Performance comparison with variance-aware algorithms

Our empirical analysis aimed to understand how variance-aware bandit algorithms compare with non-variance-aware algorithms in terms of performance and the exploration-exploitation trade-off. The variance-aware algorithms consistently outperformed the non-variance-aware algorithms across different performance measures, demonstrating their superior ability to handle the exploration-exploitation trade-off. These measures included average reward collection, regret minimization, risk assessment, and the efficiency of exploration versus exploitation.

Variance-aware algorithms showed a consistent ability to collect higher average rewards by distinguishing between arms more effectively, resulting in more optimal pulls. This was evident across all tested reward distributions, indicating that these algorithms are better at maximizing returns. The analysis of total regret revealed that variance-aware algorithms maintained lower and more consistent regret over time, suggesting their reliability in minimizing cumulative opportunity loss. This aspect is crucial for optimizing long-term performance. In terms of risk management, variance-aware algorithms exhibited a lower risk profile through the assessment of the VaR function, indicating their proficiency in avoiding large losses. Furthermore, variance-aware algorithms demonstrated a more efficient exploration-exploitation balance, adapting quickly to the reward landscape and pulling fewer suboptimal arms. This efficiency is vital in dynamic environments, where the ability to identify and exploit the best options rapidly leads to better overall performance. In contrast, non-variance-aware algorithms showed relatively lower performance across these metrics. They struggled more with identifying optimal arms, leading to higher regret and a less efficient balance between exploration and exploitation. Their VaR calculations were also less favorable, with higher potential losses, reflecting their inability to manage variability effectively.

Based on the comprehensive empirical evaluation, it is evident that variance-aware bandit algorithms offer significant advantages over non-variance-aware algorithms. These advantages manifest in higher average rewards, lower and more consistent regret, better risk management, and a more efficient exploration-exploitation trade-off. The incorporation of variance estimates allows these algorithms to handle uncertainty and variability more effectively, leading to superior overall performance. Consequently, variance-aware bandit algorithms should be the preferred choice in applications where optimizing long-term rewards and managing risks are critical. In conclusion, our empirical analysis strongly supports the theoretical superiority of variance-aware bandit algorithms, establishing them as the more efficient and effective approach to handling the exploration-exploitation trade-off in various settings.

Threats to Validity

In our simulation and empirical analysis, several threats to validity emerged that need to be addressed. One significant threat arises from the design of the implemented argmax function on the array $[a_{optimal}, a_{suboptimal}]$. As we discussed in Section 4.2, this function is designed to automatically draw the optimal arm, which can introduce bias if the reader is not given the option to decide the position of the optimal arm in the array. The function defaults to choosing the first index, potentially skewing results if the optimal arm is not in that position. This issue is particularly relevant for the ε-Greedy algorithm, where the order of the arms affects the outcome. During the exploitation phase, if both arms initially have the same value, the argmax function will default to the first entry, creating a bias. To ensure a fairer comparison, we provided the option to choose the arm's position in all three cases of Bernoulli variables.

Another threat concerns the adaptation of the UCB-Improved algorithm. In our study, we modified the algorithm to include a running variable in the arm selection process, as detailed in Remark 3.2.1. This adaptation ensures that only one arm is drawn per time step, facilitating comparison with other algorithms. However, this modification may compromise the original efficiency of UCB-Improved as intended by P. Auer et al. in [3], affecting the results of this simulation.

Additionally, non-optimization of algorithm parameters could pose a threat to the validity of our results. As noted in Section 4.2, we fixed the parameters for all algorithms, but allowing for free parameter selection might yield different performance outcomes. Furthermore, J.-Y. Audibert et al. suggested in [2] running simulations for a time horizon n up to 1,000,000 time steps, whereas our study was limited to 100,000 time steps. Extending the time horizon could provide a clearer distinction between the exploration and exploitation phases, especially in long term performances.

Moreover, our study focused on a two-armed bandit model with Bernoulli distributed rewards, except for UCB-Normal. Although we explored three versions of arm means, a more comprehensive evaluation would include additional k-armed bandit models to assess the algorithms' optimality in selecting the best arm as the number of arms increases.

Despite these threats, our work provides a thorough empirical analysis on variance-aware and non-variance-aware algorithms. While the potential optimizations and extended simulations would provide an even more extensive evaluation on the exploration-exploitation trade-off, they exceed the scope of our thesis. Our simulations and codes lay a comprehensive foundation, allowing future researchers with more computational resources and time to easily implement these adaptations. Thus, while acknowledging these threats, we believe that our study comprehensively compares the performance of variance-aware and non-variance-aware bandit algorithms in terms of the exploration-exploitation trade-off.

5 Conclusion

In this thesis, we aimed to understand how variance-aware bandit algorithms compare with non-variance-aware algorithms in terms of performance and their exploration-exploitation trade-off. Our primary focus was on analyzing the theoretical and empirical aspects of these algorithms to draw a conclusion on this question.

We began by laying a theoretical foundation, discussing the basics of stochastic bandit problems and the development of the Upper Confidence Bound (UCB) algorithm. This provided the necessary background to examine variance-aware algorithms like UCB-V and EUCBV, which we derived and analyzed in depth. These algorithms aim to improve decision-making by incorporating variance estimates of rewards, thereby addressing one of the key challenges in the MAB problem, which is to balance exploration and exploitation.

Our empirical analysis, presented through comprehensive simulations, compared the performance of variance-aware and non-variance-aware algorithms. The results showed the superior performance of variance-aware algorithms in different scenarios. They were more efficient in exploration, pulling suboptimal arms less frequently and achieving lower total regret over time compared to their non-variance-aware counterparts. This advantage became particularly clear when the differences between the reward distributions of the arms were subtle.

In summary, our thesis demonstrated that variance-aware bandit algorithms, by incorporating variance estimates, offer significant advantages in optimizing decision-making in uncertain environments. These algorithms effectively balance the exploration-exploitation trade-off, leading to improved performance. While further research with extended simulations and varying parameters could yield additional insights, our work provides a strong foundation for understanding the benefits of variance-aware approaches. This thesis contributes to the field by offering both theoretical insights and empirical evidence, highlighting the importance and effectiveness of variance-aware algorithms in MAB problems.

6 Literature

References

[1] J.-Y. Audibert and S. Bubeck. Minimax policies for adversarial and stochastic bandits. 1985. URL https://www.di.ens.fr/willow/pdfscurrent/COLT09a.pdf.

[2] J.-Y. Audibert, R. Munos, and C. Szepesvári. Exploration-exploitation tradeoff using variance estimates in multi-armed bandits. *Theoretical Computer Science*, 410:1876–1902, 2009. doi: 10.1016/j.tcs.2009.01.016. URL https://www.sciencedirect.com/science/article/pii/S030439750900067X.

[3] P. Auer and R. Ortner. Ucb revisited: Improved regret bounds for the stochastic multi-armed bandit problem. *Period Math Hung*, (61):55–65, 2010. doi: https://doi.org/10.1007/s10998-010-3055-6.

[4] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2–3):235–256, 2002. URL https://homes.di.unimi.it/~cesabian/Pubblicazioni/ml-02.pdf.

[5] L. Döring. *The Mathematics of Reinforcement Learning.* 2023. URL https://www.wim.uni-mannheim.de/media/Lehrstuehle/wim/doering/RL/RL_test_Leif.pdf/flipbook.

[6] M. Enzenberger. The ucb algorithm, 2007. URL https://webdocs.cs.ualberta.ca/~games/go/seminar/notes/2007/slides_ucb.pdf.

[7] A. Garivier and O. Cappé. The kl-ucb algorithm for bounded stochastic bandits and beyond. 2011. doi: 10.48550/arXiv.1102.2490. URL https://arxiv.org/pdf/1102.2490.

[8] A. Garivier and O. Cappé. Kullback-leibler upper confidence bounds for optimal sequential allocation. *The Annals of Statistics*, 41:1516–1541, 2013. doi: 10.1214/13-AOS1119. URL https://arxiv.org/pdf/1210.1136.

[9] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58:13–30, 1963. doi: 10.2307/2282952. URL https://www.jstor.org/stable/2282952.

[10] T.L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6:4–22, 1985. doi: 10.1016/0196-8858(85)90002-8. URL https://www.sciencedirect.com/science/article/pii/0196885885900028.

[11] T. Lattimore and C. Szepesvári. *Bandit Algorithms.* Cambridge University Press, 1st edition, 2020. URL https://tor-lattimore.com/downloads/book/book.pdf.

[12] Y.-C. Liu and Y. Tsuruoka. Modification of improved upper confidence bounds for regulating exploration in monte-carlo tree search. *Theoretical Computer Science*, 644:92–105, 2016. ISSN 0304-3975. doi: https://doi.org/10.1016/j.tcs.2016.06.034. URL `https://www.sciencedirect.com/science/article/pii/S0304397516302791`. Recent Advances in Computer Games.

[13] A. Makone. Reinforcement learning 6: Exploration vs exploitation, 2021. URL `https://ashutoshmakone.medium.com/reinforcement-learning-5-exploration-vs-exploitation-c1bae5a2ea42`.

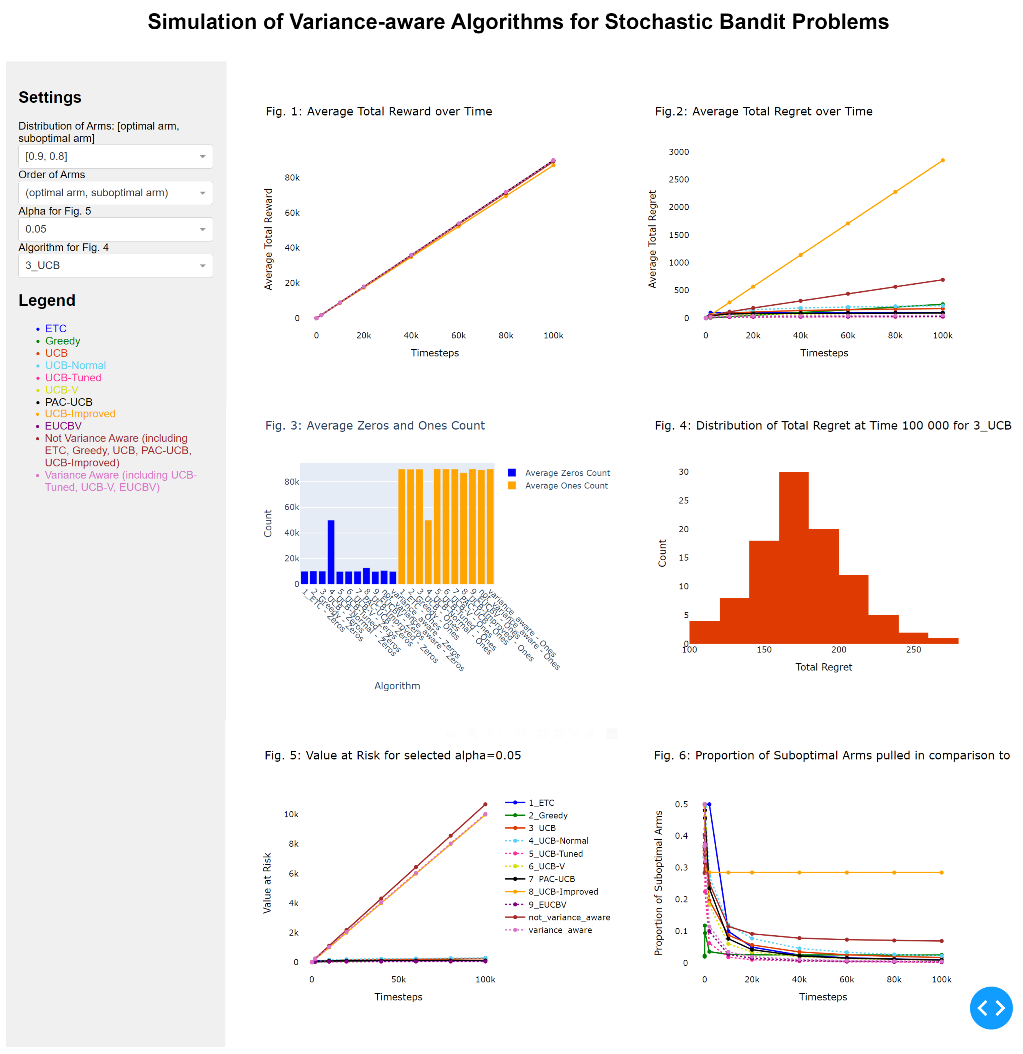
[14] S. Mukherjee, K. P. Naveen, N. Sudarsanam, and B. Ravindran. Efficient-ucbv: An almost optimal algorithm using variance estimates. *CoRR*, abs/1711.03591, 2017. URL `http://arxiv.org/abs/1711.03591`.

[15] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 2nd edition, 2018. URL `https://web.stanford.edu/class/psych209/Readings/SuttonBartoIPRLBook2ndEd.pdf`.

[16] W. R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 1933. doi: 10.2307/2332286. URL `https://www.jstor.org/stable/2332286`.

[17] S. Wilks. *Mathematical statistics*. John Wiley and Sons, 1962.

A Appendix


Simulation of Variance-aware Algorithms for Stochastic Bandit Problems

Settings

Distribution of Arms: [optimal arm, suboptimal arm]
[0.9, 0.8]

Order of Arms
(optimal arm, suboptimal arm)

Alpha for Fig. 5
0.05

Algorithm for Fig. 4
3_UCB

Legend

- ETC
- Greedy
- UCB
- UCB-Normal
- UCB-Tuned
- UCB-V
- PAC-UCB
- UCB-Improved
- EUCBV
- Not Variance Aware (including ETC, Greedy, UCB, PAC-UCB, UCB-Improved)
- Variance Aware (including UCB-Tuned, UCB-V, EUCBV)

Fig. 1: Average Total Reward over Time
Average Total Reward
Timesteps
Fig.2: Average Total Regret over Time
Average Total Regret
Timesteps
Fig. 3: Average Zeros and Ones Count
Count
Algorithm
Average Zeros Count
Average Ones Count
Fig. 4: Distribution of Total Regret at Time 100 000 for 3_UCB
Count
Total Regret
Fig. 5: Value at Risk for selected alpha=0.05
Value at Risk
Timesteps
1_ETC
2_Greedy
3_UCB
4_UCB-Normal
5_UCB-Tuned
6_UCB-V
7_PAC-UCB
8_UCB-Improved
9_EUCBV
not_variance_aware
variance_aware
Fig. 6: Proportion of Suboptimal Arms pulled in comparison to
Proportion of Suboptimal Arms
Timesteps

Figure 1: Dashboard Interface

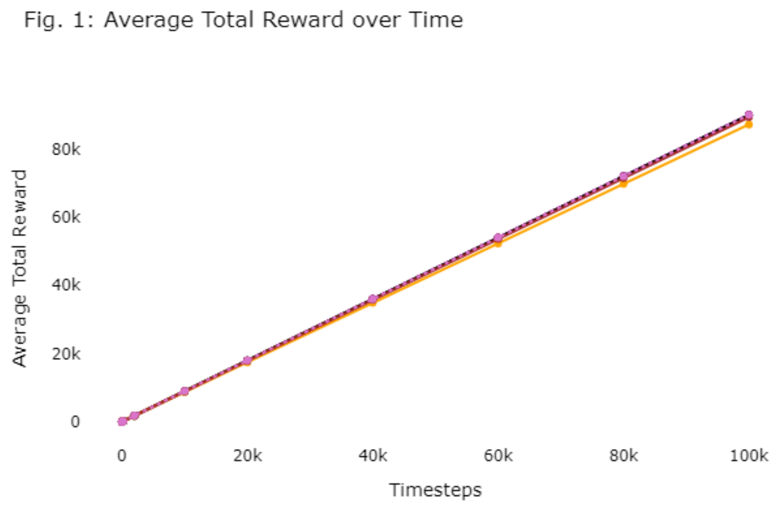


Figure 2: Average Total Reward over Time (Fig.1 of the dashboard) with the distribution and order of arms: [0.9, 0.8]

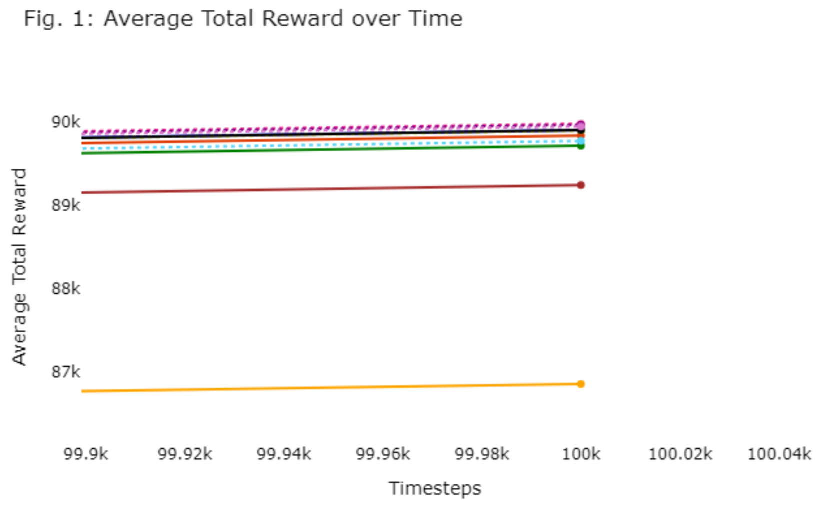


Figure 3: Average Total Reward over Time (Fig.1 of the dashboard) zoomed in on time step 100 000 with the distribution and order of arms: [0.8, 0.9]

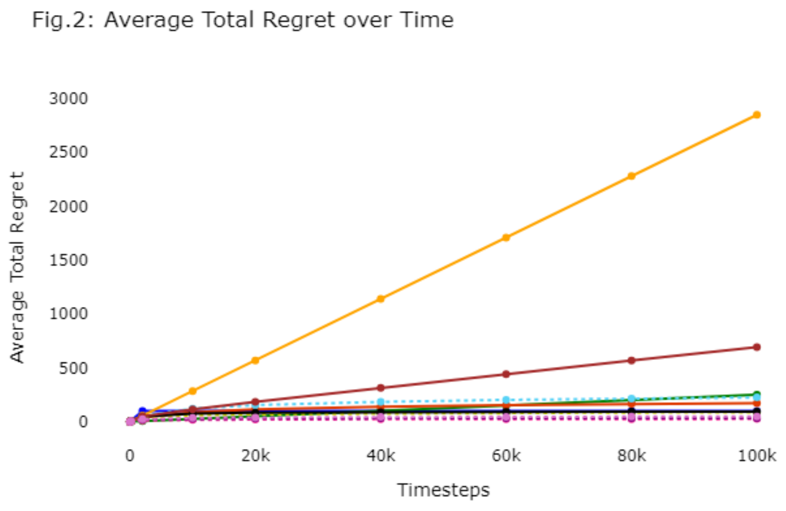


Figure 4: Average Total Regret over Time (Fig.2 of the dashboard) with the distribution and order of arms: [0.9, 0.8]

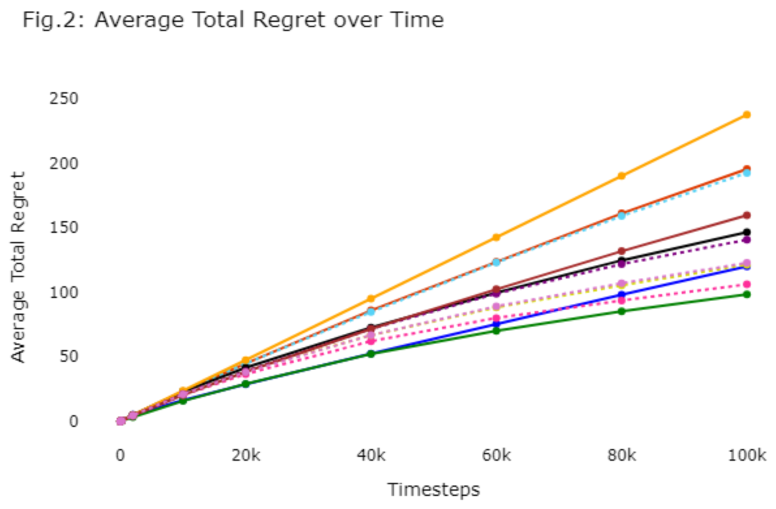


Figure 5: Average Total Regret over Time (Fig.2 of the dashboard) with the distribution and order of arms: [0.9, 0.895]

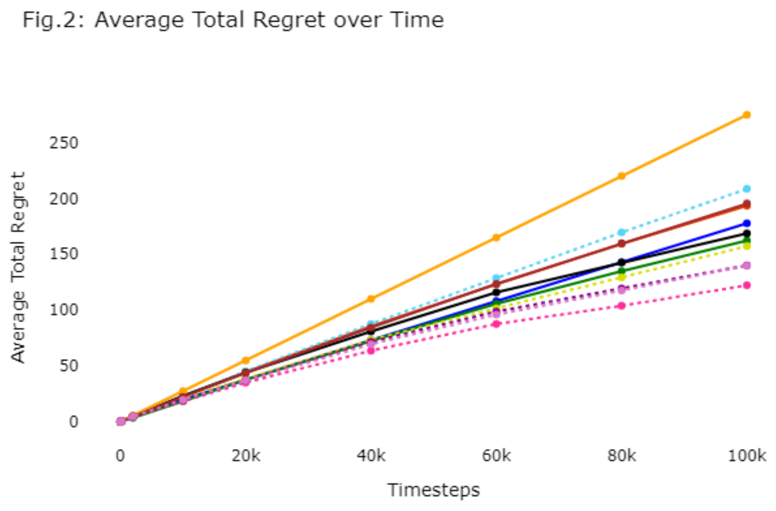


Figure 6: Average Total Regret over Time (Fig.2 of the dashboard) with the distribution and order of arms: [0.5, 0.495]

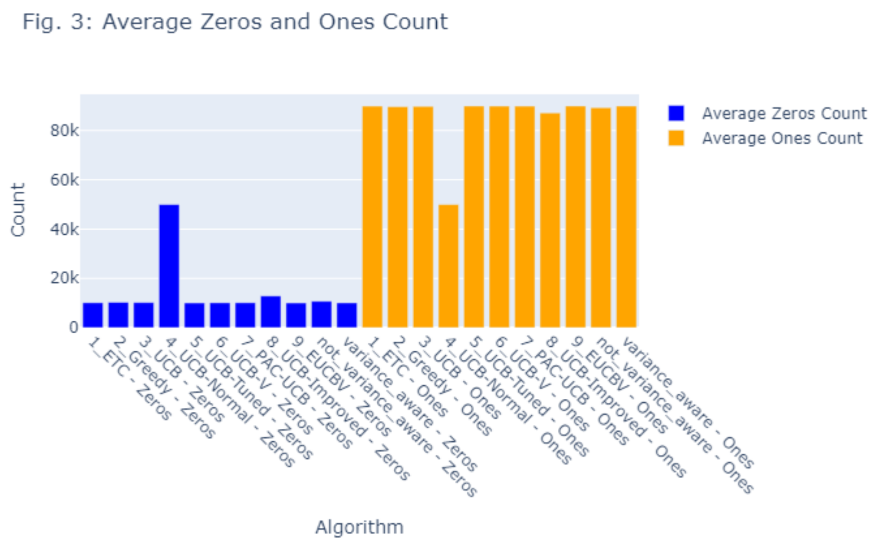


Figure 7: Average Zeros and Ones Count (Fig.3 of the dashboard) with the distribution and order of arms: [0.9, 0.8]

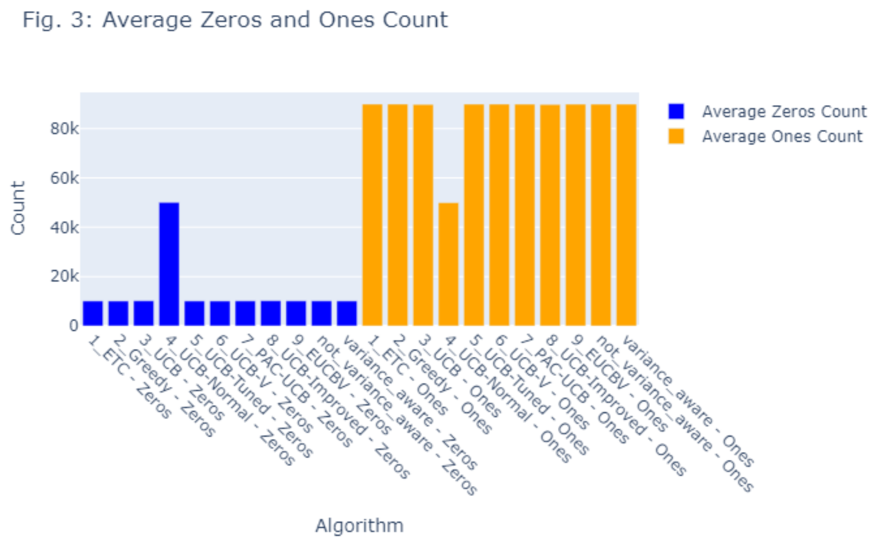


Figure 8: Average Zeros and Ones Count (Fig.3 of the dashboard) with the distribution and order of arms: [0.9, 0.8]

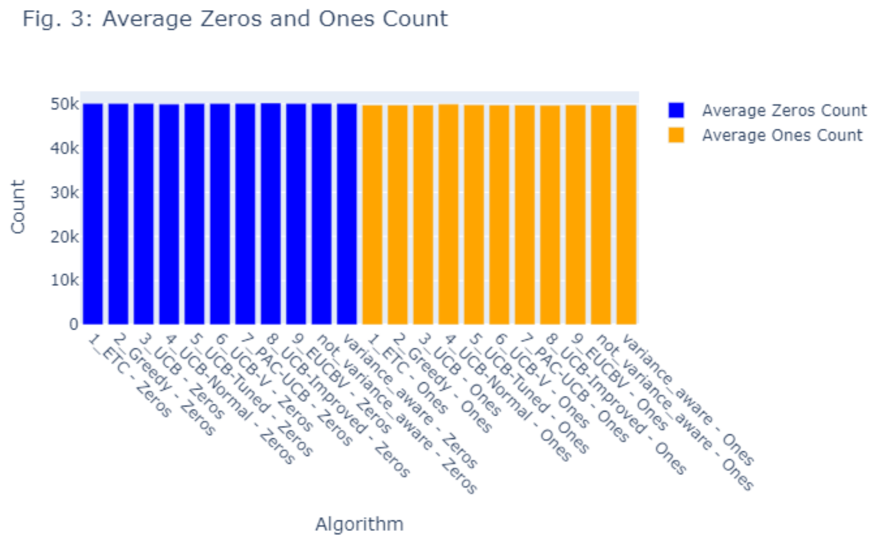


Figure 9: Average Zeros and Ones Count (Fig.3 of the dashboard) with the distribution and order of arms: [0.9, 0.8]

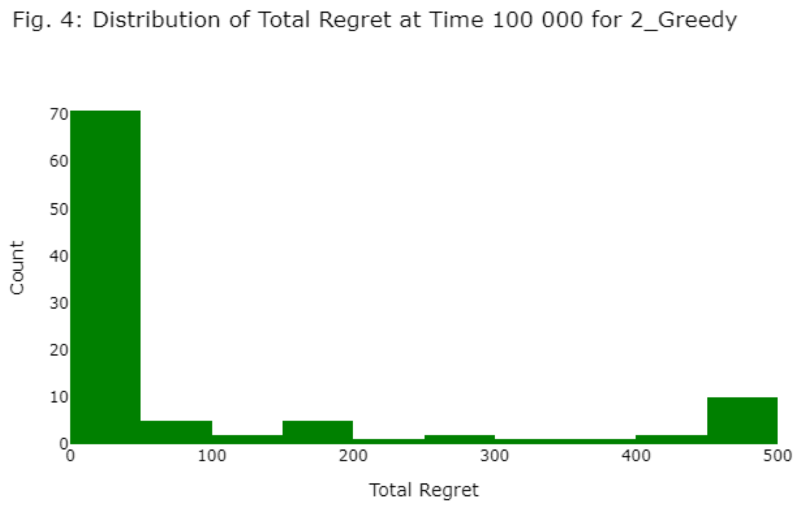


Figure 10: Distribution of Total Regret at Time 100 000 (Fig.4 of the dashboard) for the ε-Greedy algorithm with the distribution and order of arms: [0.9, 0.8]

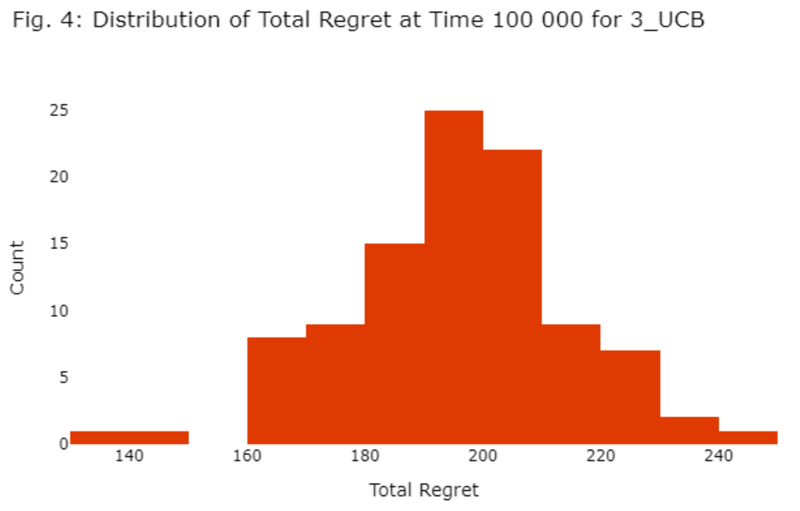


Figure 11: Distribution of Total Regret at Time 100 000 (Fig.4 of the dashboard) for the UCB algorithm with the distribution and order of arms: [0.9, 0.8]

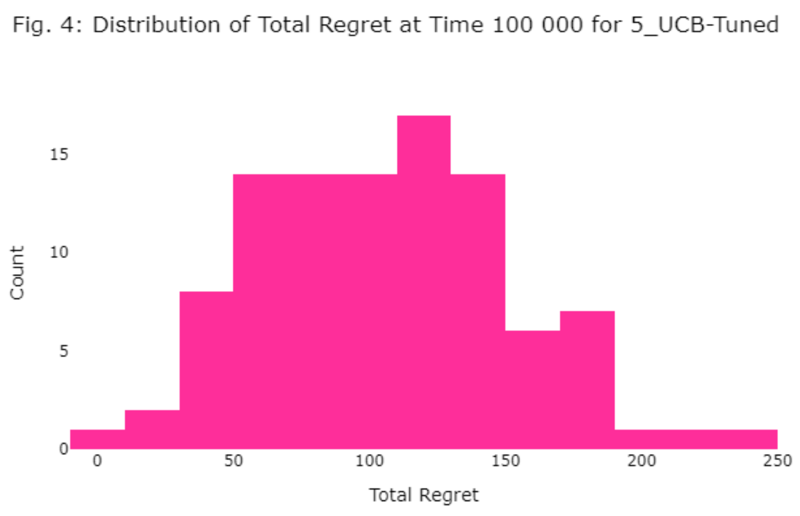


Figure 12: Distribution of Total Regret at Time 100 000 (Fig.4 of the dashboard) for the UCB-Tuned algorithm with the distribution and order of arms: [0.9, 0.8]

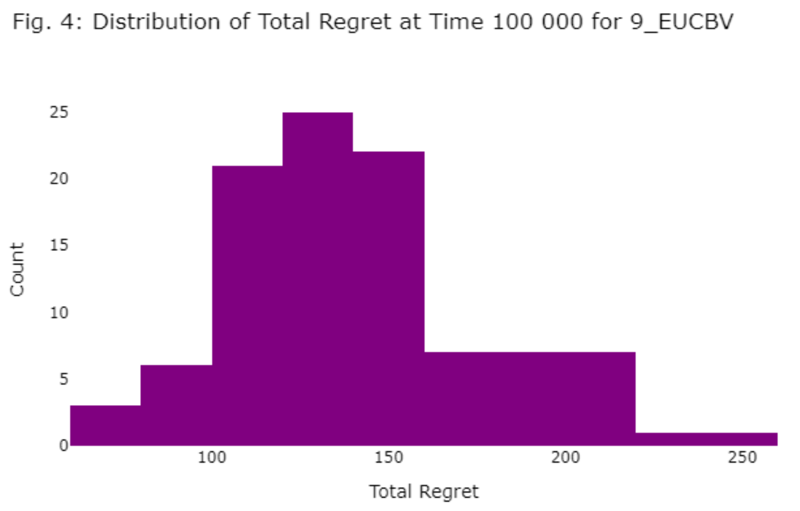


Figure 13: Distribution of Total Regret at Time 100 000 (Fig.4 of the dashboard) for the EUCBV algorithm with the distribution and order of arms: [0.9, 0.8]

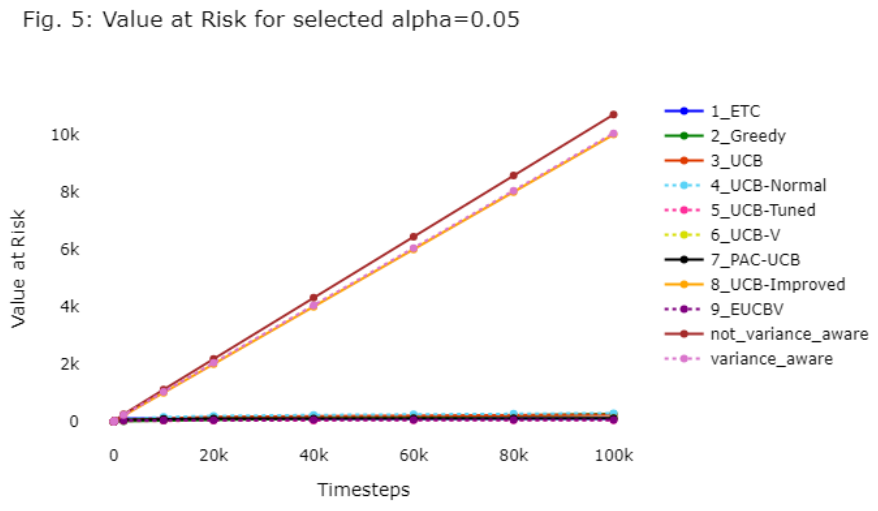


Figure 14: Value at Risk for selected $\alpha = 0.05$ *(Fig.5 of the dashboard) with the distribution and order of arms: [0.9, 0.8]*

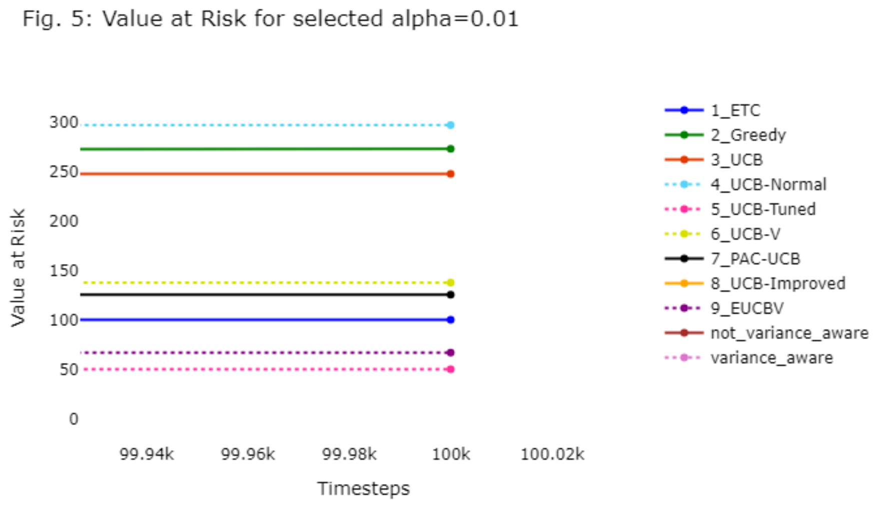


Figure 15: Value at Risk for selected $\alpha = 0.05$ *(Fig.5 of the dashboard) zoomed in on time step 100 000 with the distribution and order of arms: [0.9, 0.8]*

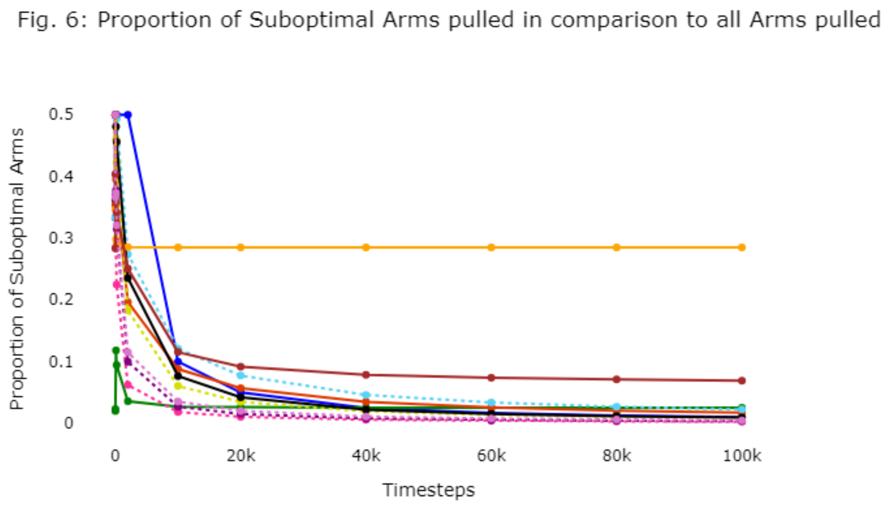


Figure 16: Average Total Regret over Time (Fig.2 of the dashboard) with the distribution and order of arms: [0.9, 0.8]

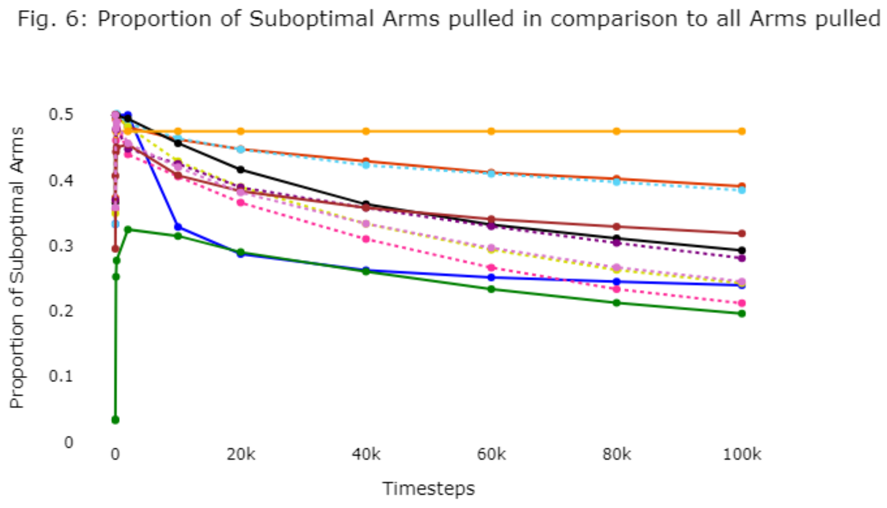


Figure 17: Average Total Regret over Time (Fig.2 of the dashboard) with the distribution and order of arms: [0.9, 0.8]

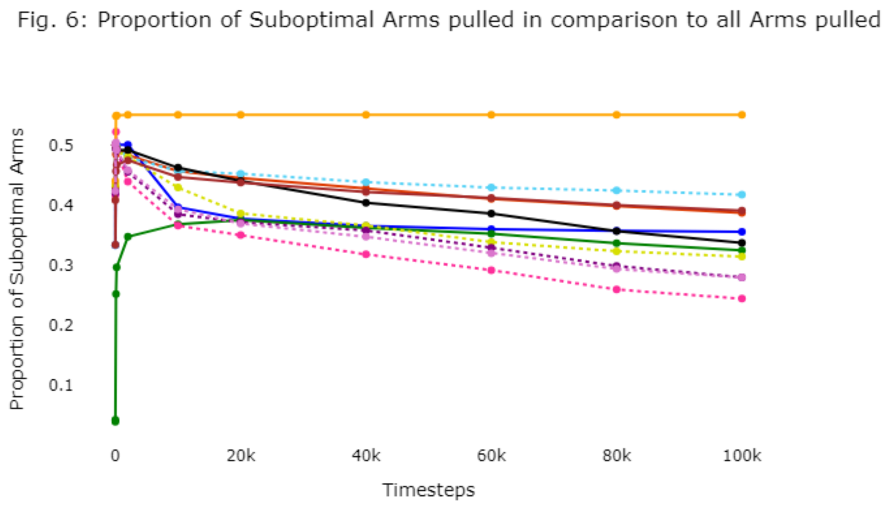


Figure 18: Average Total Regret over Time (Fig.2 of the dashboard) with the distribution and order of arms: [0.9, 0.8]