FRA503 Deep Reinforcement Learning for Robotics

Homework 1: Multi-Armed Bandit

Name	Student ID
Chantouch Orungrote	6340500011
Sasish Kaewsing	6340500076

1 Abstract

Textbook Perspective. In the k-armed bandit chapter of Reinforcement Learning: An Introduction by Sutton and Barto [1], bandit algorithms are analyzed using continuous-valued stochastic rewards to study learning under uncertainty and to highlight the exploration–exploitation trade-off. The underlying formulation and update mechanisms of these algorithms depend only on expected rewards and are independent of the specific reward distribution.

Experimental Design. In this homework, we build on the same theoretical formulation and consider an alternative experimental setting based on binary rewards. This success–failure reward structure allows the same bandit algorithms to be examined under a different stochastic outcome model while preserving the original learning objective.

2 Multi-Armed Bandit Problem

The Multi-Armed Bandit Problem is a single state problem formulation with sequential structure and a finite action set $A = \{1, \dots, K\}$. At each time step t, an agent selects an action $A_t \in A$ and receives a stochastic reward R_t with an unknown reward distribution for the selected action $p(r|a)$. The objective is to maximize return over the lifetime T.

2.1 Mathematical Formulation

Concept	Form	Definition
Stochastic Reward	$R_t \in \{0, 1\}$	The reward is observed after selecting action A_t. A value of 1 indicates a successful outcome, while 0 indicates no reward.
Action Value	$q^*(a) = \mathbb{E}[R_t \mid A_t = a]$	The true value of action is the expected reward. Since the bandit problem has a single state, the value depends only on the immediate reward.
Optimal Action	$a^* = \arg\max_{a \in A} q^*(a)$	The action with the highest expected reward. Selecting this action yields the best long-term performance.
Optimal Value	$q^*(a^*) = \max_{a \in A} q^*(a)$	The maximum achievable expected reward per timestep, used as a performance benchmark.
Value Estimation	$Q_{t+1}(a) = Q_t(a) + \frac{1}{N_t(a)}\big(R_t - Q_t(a)\big)$	Sample-average update rule used to estimate action values from observed rewards, where $N_t(a)$ is the number of times the action a has been selected.
Cumulative Regret	$L(T) = \sum_{t=1}^{T} \big(q^*(a^*) - q^*(A_t)\big)$	Measures the total performance loss compared to always selecting the optimal action. Lower regret indicates faster and more effective learning.

Table 1: Mathematical formulation of the multi-armed bandit problem.

2.2 Understanding the Algorithms

The Epsilon-Greedy alternates between exploration and exploitation by randomizing action selection.
Which exploits the current value estimates $Q_t(a)$ with probability $1-\epsilon$, explores by selecting a random action with ϵ.

$$A_t = \begin{cases} \text{a random action}, & \text{with probability } \epsilon, \\ \arg\max_a Q_t(a), & \text{with probability } 1-\epsilon, \end{cases} \tag{1}$$

The Upper Confidence Bound selects actions by augmenting the value estimate with an exploration bonus.
Select actions by combining $Q_t(a)$ with an uncertainty bonus term $c\sqrt{\frac{\log t}{N_t(a)}}$, which is larger for less-sampled actions $N_t(a)$ and decreases over time, resulting in directed exploration.

$$A_t = \arg\max_a \left(Q_t(a) + c\sqrt{\frac{\log t}{N_t(a)}} \right). \tag{2}$$

3 Experiment Design

3.1 Bernoulli Reward Setting

A Bernoulli reward model is used when each action yields a binary outcome, representing success or failure. This setting provides a simple and interpretable stochastic reward structure while preserving the essential exploration–exploitation challenge of the multi-armed bandit problem.

Fixed (p_a) the multi-armed bandit, where each arm produces a reward $\{0, 1\}$ with an arm-specific success probability. The reward distribution of each arm is fixed throughout all experiments and does not change across runs.

Arm (a)	1	2	3	**4**	5	6	7	8	9	10
Success probability (p_a)	0.10	0.50	0.60	**0.80**	0.10	0.25	0.60	0.45	0.75	0.65

Table 2: Fixed Bernoulli reward probabilities for each arm. The optimal arm is $a^* = 4$ with $p^* = 0.80$.

This configuration includes near-optimal arms (e.g., $p = 0.75$ versus $p^* = 0.80$), making effective exploration necessary to reliably identify the optimal action.

Note: In Bernoulli setting, each arm's action value is simply the probability that it will give a reward of 1 (i.e., $q^*(a) = p_a$)

3.2 Configuration

Paramater	Value
Number of arms (k)	10
Timesteps (T)	500
Independent runs (n_{runs})	10,000
Reward type	Bernoulli(p_a)
ϵ values	$\{0, 0.01, 0.05, 0.1, 0.5\}$
c values	$\{0.5, 1, 2, 3, 5\}$
Seed	42

Table 3: Experiment parameter setup.

3.3 Evaluation Metrics

The performance is evaluated using the following metrics, averaged over 10,000 independent runs.

Metric	Description
Average Reward ($\bar{r}_t$)	The mean reward obtained at timestep t, averaged across runs. This metric reflects how quickly an algorithm improves its immediate performance over time. Its evolution across t indicates long-run learning behavior.
Optimal Action %	The proportion of runs in which the optimal arm a^* is selected at timestep t. This metric measures how reliably the algorithm identifies and exploits the optimal action.
Cumulative Regret	The accumulated difference between the optimal expected reward and the reward obtained by the selected actions up to time t. Lower regret indicates more efficient learning and better long-term decision-making.
Convergence Step	The earliest timestep at which a 50-step moving average of ($\bar{r}_t$) exceeds $0.95\,p^*$. This metric captures how quickly an algorithm stabilizes near-optimal performance.

Table 4: Evaluation metrics.

4 Analysis

Following the methodology in subsection 3.2, this section presents a comparative analysis of the performance of the ϵ-greedy and UCB agents. Based on graphical representations that illustrate the reward, optimal action selection, and regret over each timestep.

4.1 Epsilon-Greedy Performance

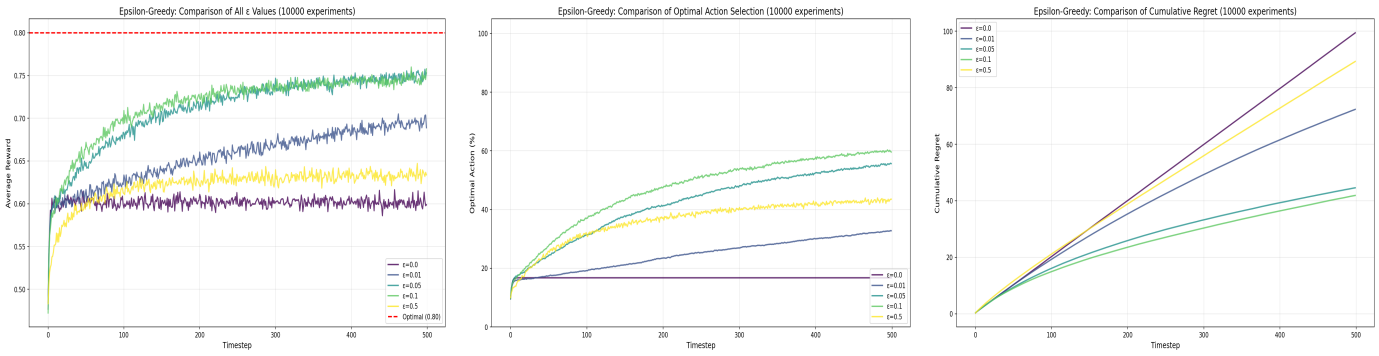

Figure 1: Epsilon-greedy reward, optimal action, and regret graphs.

Pure Exploitation Problem, in reinforcement learning, a **local minima** is a state where the agent settles on a suboptimal action because it believes it is the best possible choice. In our experiment, this happens when a arm such as $(a = 9, p = 0.75)$ gives a success $(R = 1)$ early, while the optimal arm $(a^* = 4, p^* = 0.80)$ gives a failure $(R = 0)$. Because the agent $\epsilon = 0.0$ does not explore, it stays stuck in this local minima forever.

The Ceiling Effect is a limit on the maximum average reward an agent can achieve. Even if the agent knows exactly which arm is the best, the ϵ-greedy algorithm forces it to explore random arms ϵ percent of the time. This results in a performance ceiling that is lower than the true success probability of the best arm.

To find $\mathbb{E}[R_t]$ for an ϵ-greedy agent after it has finished learning, we must look at the root questions for each part of the decision.

- **Exploitation Reward:** The probability of picking the optimal arm $(1-\epsilon)$ times its action value $(q^*(a))$.
- **Random Luck:** The probability of picking the optimal arm during a random exploration step $(\frac{\epsilon}{k})$.
- **Exploration Loss:** The probability of picking a other arms $(\epsilon - \frac{\epsilon}{k})$ times the average success rate of bad arms $(\bar{q}_{sub})$.

$$\mathbb{E}[R_t] = \underbrace{(1-\epsilon+\frac{\epsilon}{k})\cdot q^*(a)}_{\text{Selecting the Best Arm}} + \underbrace{(\epsilon-\frac{\epsilon}{k})\cdot \bar{q}_{sub}}_{\text{Selecting a Bad Arm}} \tag{3}$$

Lets $k=10, \epsilon=0.1, q^*(a)=0.80$. We find $\bar{q}_{sub}$

$$\bar{q}_{sub} = \frac{0.10+0.50+0.60+0.10+0.25+0.60+0.45+0.75+0.65}{9} = \frac{4.0}{9} \approx \mathbf{0.444} \tag{4}$$

Now, calculate the final expected reward

$$\mathbb{E}[R_t] = (0.9+0.01)\cdot 0.80 + (0.1-0.01)\cdot 0.444 = \mathbf{0.767} \tag{5}$$

This proof explains why our $\epsilon = 0.1$ graph levels off at approximately 0.77, failing to reach the 0.80 optimal line.

The Initial Convergence Gap. Observing the graph during the first 50 time steps, it is clear that the line for $\epsilon = 0.1$ increases much more rapidly compared to the other lines. This indicates that, in the Bernoulli experiment with high variance, frequent random selections in the early stages help the agent accumulate data to estimate the $Q(a)$ values closer to the true values more quickly.

On the other hand, $\epsilon = 0.01$ shows a slower ascent due to the lack of sufficient data to compare the arms. As a result, in the early stages of the experiment, the agent was unable to identify that the 4^{th} arm is the correct answer.

The Cumulative Regret Penalty. From the graph, it is evident that the different agent types generate distinct error, particularly in the group with a fixed ϵ. This results in linear regret, as there is no reduction in exploration even after the best arm is identified. Confirms that neither the greedy strategy nor the traditional ϵ-greedy strategy can achieve logarithmic regret in this experiment.

ϵ **value**	**Trend > 50 steps**	**Interpretation**
0.0	Straight	Agent settles on a suboptimal arm immediately.
0.01	Linear	Exploration is too infrequent to identify the optimal arm.
0.05	Slightly rising	Strong balance that identifies the best arm and begins to flatten the regret curve.
0.1	Slightly rising	Explores enough to learn quickly but exploits enough to keep regret low.
0.5	Slow curve	Agent spends 50% of its time choosing randomly, leading to high constant regret.

Table 5: Comparison of cumulative regret trends by ϵ-greedy

Lower ϵ Values often yield higher cumulative rewards in the long run because they reduce random mistakes once the optimal action is identified.

4.2 Upper Confidence Bound Performance

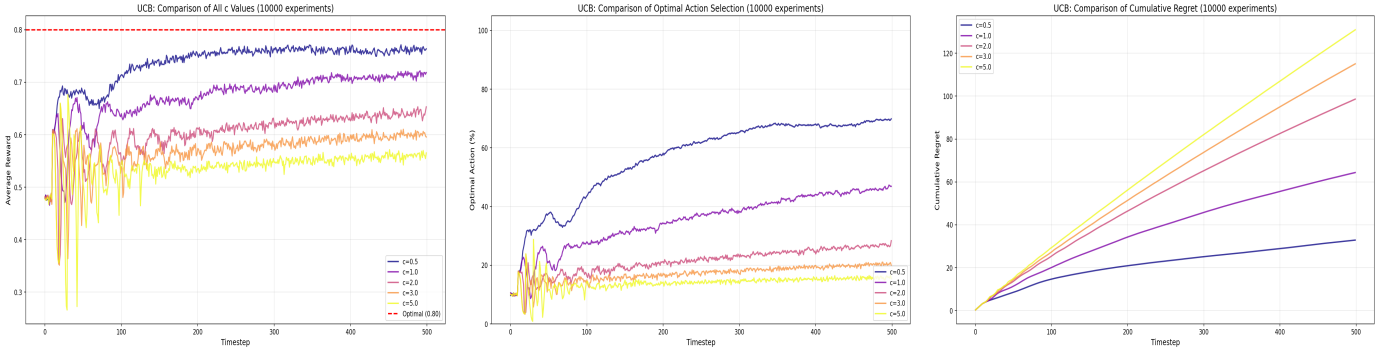

Figure 2: UCB reward, optimal action, and regret graphs.

Directed Exploration is a strategy that UCB uses to avoids local minima. Instead of picking a random arm, it calculates an uncertainty bonus for each arm. If an arm has not been pulled many times, the bonus becomes dominant, forcing the agent to try it and see if it is better than the current best. This is clearly illustrated in the reward and optimal action graph for the first 100 steps.

Optimal Actions Convergence. The value c also dictates how quickly the agent stops searching and starts exploiting the best-known arm. With a low value such as $c = 0.5$, the agent gives more weight to the actual rewards it has already seen. This allows the optimal action curve to rise quickly, reaching approximately 40% in the first 50 steps. In contrast, higher c values prevent this convergence, the agent distracted by the uncertainty terms of inferior arms, which prevents the average reward from ever reaching the optimal.

However, the linear regret $c = 5.0$ can be mathematically proven. A higher c increases the weight of the uncertainty term, which significantly alters the convergence properties of the agent in this experiment.

Non-Convergence Scenario. In theory, a high value c in the UCB may not *prevent* convergence to the optimal point in the long run $t \to \infty$. However, in practice, an excessively high value of c leads to significant performance degradation, causing the algorithm to reach optimal rewards more slowly, which may ultimately be inefficient and even becomes non-converge.

To prove *how a large value of c causes the agent to fail in selecting the optimal arm*, it is necessary to examine the condition under which the UCB value of a suboptimal arm a_{sub} surpasses that of the optimal arm.a^*.

$$Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}} > Q_t(a^*) + c\sqrt{\frac{\log t}{N_t(a^*)}} \tag{6}$$

Let $\Delta = Q_t(a^*) - Q_t(a)$ represent the value gap, and assuming the optimal arm has been sampled sufficiently $N_t(a^*) \gg 1$, its uncertainty term becomes negligible.

$$c\sqrt{\frac{\log t}{N_t(a)}} > \Delta \tag{7}$$

To find the number of times the agent is mathematically forced to pull the suboptimal arm before it notice that the arm is inferior, we solve for $N_t(a)$.

$$N_t(a) < \frac{c^2 \log t}{\Delta^2} \tag{8}$$

From this inequality, we able to translate into 3 main causes.

- **Quadratic Parameter:** c is squared (c^2), which increases the $N_t(a)$ exponentially.
- **Delayed Exploitation:** If c is very large (e.g., $c = 5$), the right side of the inequality might equal $10,000$. This means the agent is mathematically obligated to pull a suboptimal arm $10,000$ times before the uncertainty bonus finally drops enough to allow the agent to pick the optimal arm.
- **Time Run Out:** In most experiments, the time ends before the agent finishes these exploration pulls. The agent fails because it runs out of time while still trying to prove that a suboptimal arm is actually suboptimal.

5 Conclusion

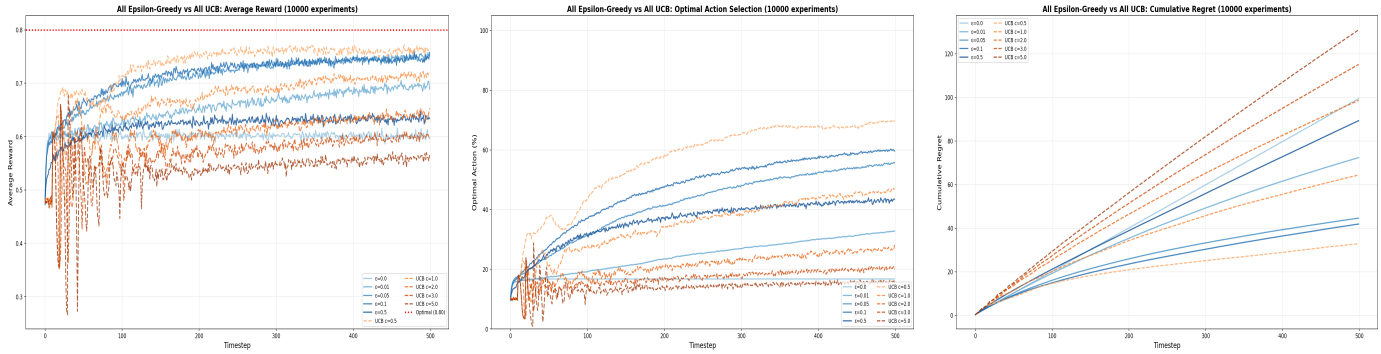

Figure 3: Both algorithm reward, optimal action, and regret graphs.

5.1 Why is UCB more effective than ϵ-greedy in Multi-Armed Bandit problems

Difference Exploration Mechanism. The ϵ-greedy method explores by randomly selecting actions that are not the best, without any prioritization of which action is likely to be better. In contrast, UCB explores by considering the potential of being the best option, calculated from how close it is to the maximum value along with the uncertainty term.

Handling Uncertainty. UCB uses a calculation formula that combines $Q_t(a)$ with the uncertainty term.

- If an action is selected fewer times, the uncertainty term increases, leading UCB to explore that action further for confidence.
- This mechanism helps UCB reduce the frequency of selecting suboptimal actions over time, while ϵ-greedy with a fixed ϵ continues to waste of the time on randomly selecting suboptimal actions indefinitely.

Explicit Performance. From Reinforcement Learning: An Introduction by Sutton and Barto [1], UCB tends to achieve higher average rewards than ϵ-greedy. In parameter studies, it was found that when tuned optimally, UCB outperforms other methods in terms of average rewards, as compared to other methods discussed in Chapter 2.

Limitations of UCB Compared to ϵ-greedy Although UCB performs better in bandit problems, it indicate that **ϵ-greedy remains important** because:

- **Difficulty in scaling:** Applying UCB in complex reinforcement learning problems (e.g., those with many states or requiring function approximation) is much harder and often less effective in practice.
- **Nonstationary Problems:** UCB struggles with problems where rewards change over time and requires more complex methods than usual.

In conclusion: UCB in terms of **more intelligent exploration**, leading to higher cumulative rewards in typical Bandit problems, but may lack the flexibility and simplicity that ϵ-greedy offers when dealing with large-scale learning problems.

Aspect	**ϵ-greedy**	**UCB**
Exploration Strategy	Treats all arms equally.	Prioritizes uncertainty and last sampling.
Convergence Rate	Linear Regret	Logarithmic Regret
Parameter Sensitivity	High ϵ randomizes, Low ϵ local minima.	High c delays exploitation, Low c local minima
Environment Fit	Non-stationary	Stationary.

Table 6: Comparative Analysis: ϵ-greedy vs. UCB

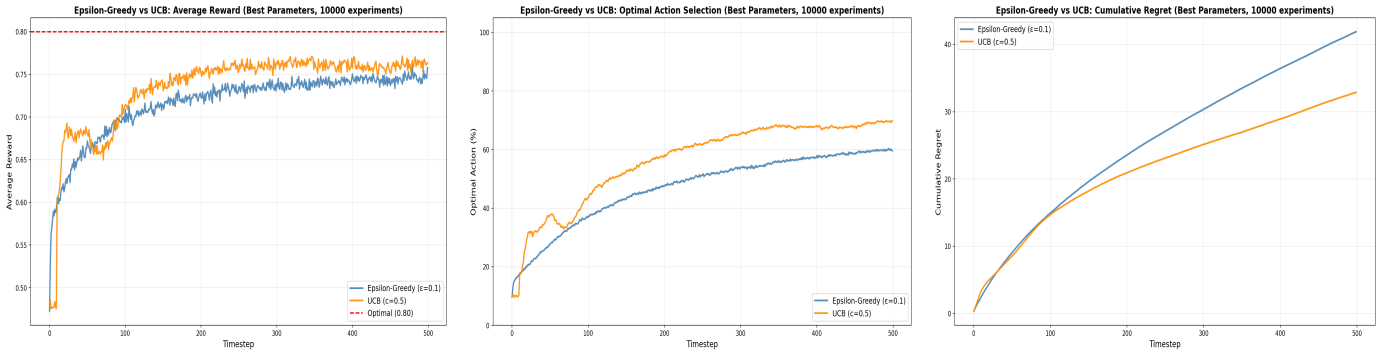

Figure 4: The best reward, optimal action, and regret for each algorithm.

5.2 Exploration–Exploitation Tradeoff

The **exploration–exploitation tradeoff** is the fundamental challenge in Reinforcement Learning (RL), specifically in Multi-Armed Bandit problems. Success depends on how effectively an agent balances gathering new information versus maximizing immediate rewards.

From this experiment, we gain the knowledge from observing the graphs, reviewing the recommended RL textbook, and exploring the internet world. That we can conclude the trade-off.

- **There is no single perfect number.** The best parameter depends on the reward distribution, and whether the environment changes over time.

Based on an RL reference textbook [1], the approaches used for finding the *sweet spot* for the parameters like ϵ or c are:

- **The Inverted-U Shape:** Performance typically follows an inverted-U curve. If the exploration parameter is too low, learning is slow and prone to error. If it is too high, the agent remains too distracted by random choices to ever perform well.
- **Stationary vs. Non-stationary Environments:**
  - **Stationary:** Rewards don't change over time. Exploration can be gradually reduced (decayed) as the agent becomes more confident.
  - **Non-stationary:** Rewards change over time. The agent **must maintain** a constant level of exploration to detect if a previously bad arm has become the new best arm.

5.3 Additional

Optimistic Initial Values Q_0. By setting initial reward estimates higher than any possible actual reward, the agent is naturally disappointed by its first few pulls. This forces it to explore every available option at least once before settling on a favorite.

- **Overestimated Initial Values:** The initial reward estimates ($Q_1(a)$) are set significantly higher than actual rewards (e.g., +5 instead of 0).
- **Disappointment Mechanism:** Initial rewards will always be lower than expected, causing the agent to feel disappointed and explore other actions.
- **Forced Exploration:** This causes every action to be tried repeatedly until estimates converge to real values.
- **Exploration Despite Greedy Approach:** Even with a greedy approach, actions tried previously will have lower values, prompting exploration of less-tried actions.

References

[1] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 2 edition, 2018.