Multi-Armed Bandits: Case Study

Author: Pratish Mashankar, pmashank@gmu.edu
Guide: Dr. Sanmay Das, sanmay@gmu.edu

PROBLEM STATEMENT

To implement three different algorithms for multi-armed Bernoulli bandits: the greedy algorithm, UCB1, and Thompson Sampling. Write a function for each of the three, which takes as input (1) a vector containing the probabilities with which each arm gives a payoff of 1, and (2) the number of iterations to run for. Test your algorithms on two different settings: (1) an eleven-armed bandit with payoff probabilities 0, 0.1, 0.2, . . . , 1.0, and (2) a five-armed bandit with payoff probabilities 0.3, 0.5, 0.7, 0.83, 0.85. Use these test cases to analyze the empirical properties of these algorithms, at least in terms of regret over time and the probability of choosing the best action over time. Minimally, present a graph of average regret vs. time for each of the algorithms, and some kind of visualization of which action they are choosing over time. Be sure to label your axes and caption your figures appropriately. I would suggest trying the algorithms with at least 10^3, 10^4, and 10^5, time steps, and also re-running them from the start many times and averaging (so, for example, run the experiment with 10^3 time steps several hundred times). What do your results tell you about the properties of these three different algorithms? On this question in particular, I urge you to experiment and also present and write up any other interesting insights you may have (for example, playing with different probability vectors or different ways of analyzing the data). [**Adapted from Professor Sanmay Das' CS688 Fall 2023 Homework 5 - 7 December 2023**]

INTRODUCTION

I implemented three distinct algorithms for addressing multi-armed Bernoulli bandit problems: the greedy algorithm, UCB1, and Thompson Sampling. To evaluate their performance, I conducted extensive testing on two different bandit settings. The first setting involved an eleven-armed bandit with varying payoff probabilities from 0 to 1.0, while the second setting featured a five-armed bandit with specific probabilities of 0.3, 0.5, 0.7, 0.83, and 0.85. Employing these scenarios, I systematically examined the empirical properties of the algorithms, focusing on regret over time and the probability of selecting the optimal action over time. The experimentation involved testing the algorithms at different time step counts, specifically 10^3, 10^4, and 10^5 once, and then conducting multiple runs by re-initiating them from the start 100 times and averaging the results—that is running the experiment with 10^3, 10^4, and 10^5 time steps 100 times.

EXPERIMENT SETUP

I defined a *simulate* function that simulates the bandit algorithm for a specified number of iterations (10^3, 10^4, and 10^5 in this case). It takes a bandit instance, an algorithm instance, and the number of iterations as input. During each iteration, it selects an arm, pulls it to get a reward, updates the algorithm, and stores

the chosen arm and reward for later analysis. My *BernoulliBandit* class represents the multi-armed Bernoulli bandit. It is initialized with a list of probabilities for each arm. The pull method simulates pulling a specified arm and returns a binary reward based on whether a randomly generated number is less than the arm's probability. I defined the three algorithms in three classes namely *Greedy*, *UCB1*, and *ThompsonSampling.*

The *Greedy* class represents the Greedy algorithm. It maintains counts and values arrays for each arm, where counts track how many times an arm has been played, and values track the total reward of each arm. The *select_arm* method initially plays each arm once and then always selects the arm with the highest average reward. The *update* method updates counts and values based on the chosen arm and the received reward. The *UCB1* class represents the UCB1 algorithm. Similar to the Greedy algorithm, it maintains counts and values arrays for each arm. The *select_arm* method uses the UCB1 formula to balance exploration and exploitation. The *update* method updates counts and values based on the chosen arm and the received reward. The *ThompsonSampling* class represents the Thompson Sampling algorithm. It maintains arrays for successes (alpha) and failures (beta) for each arm. The *select_arm* method samples theta values from a Beta distribution for each arm and chooses the arm with the highest sampled value. The *update* method updates successes and failures based on the chosen arm and the received reward.

My implementation is first set up for a single run of the bandit problem with specified probabilities for the arms. It initializes instances of the bandit and each algorithm, simulates the algorithms for a specified number of iterations, calculates regrets, and plots the results. Then the bandit problem is implemented for 100 repetitions, accumulating regrets and the probability of choosing the best arm over time. It then averages these results over all repetitions and plots the averaged data.

The *plot_all_graphs* function takes the regrets and probability of choosing the best arm for each algorithm and plots three subplots: (1) Regret Over Time, (2) Probability of Choosing the Best Arm Over Time, and (3) Scatter Plot of Chosen Arms Over Iterations.

The entire experiment was performed for two test cases:

1. The eleven-armed bandit with probabilities: 0, 0.1, 0.2, 0.3…1.0
2. The five-armed bandit with probabilities 0.3, 0.5, 0.7, 0.83, 0.85

The analysis is given below

RESULTS

We identify two cases, first as the 11-armed bandits and second as the 5-armed bandits. For each case, we define two runs, a single pass, and 100 runs with values initialized from the start at each time and averaged over all runs. For each run, we define three iterations or timesteps- 10^3, 10^4, and 10^5. For each timestep, we observe three plots

1. **Regret Over Time:** The *regret* obtained at each time step. In the case of multiple runs, *average regret* obtained over 100 runs at each time step.

2. **Probability of Choosing the Best Arm Over Time:** In the case of multiple runs, *average probability* obtained over 100 runs at each time step.
3. **Scatter Plot of Chosen Arms Over Iterations:** The arms chosen by Greedy, UCB1, and Thompson Sampling strategies over iterations

CASE STUDY 1: 10 ARMED BANDITS

SINGLE PASS

1. **Regret Over Time**
    a. **1,000 Iterations:** The regret is relatively low for all algorithms, with the greedy algorithm showing higher regret as compared to UCB1 and Thompson Sampling. Thompson Sampling exhibits the best performance with the lowest regret, indicating that it balances exploration and exploitation efficiently even in the short term.
    b. **10,000 Iterations:** As the iterations increase, the total regret for the UCB1 increases significantly, but Thompson Sampling still maintains a comparatively lower regret. This suggests that Thompson Sampling continues to effectively identify the optimal arm over time, whereas the UCB1 accrues higher regret possibly due to less effective exploration.
    c. **100,000 Iterations:** At this scale, the differences become more pronounced. The regret for the UCB1 continues to rise steeply, while Thompson Sampling's regret increases at a much slower rate, reinforcing its superiority in long-term performance in this setup.
2. **Best Arm Selection Probability**
    a. **1,000 Iterations:** All algorithms improve their probability of selecting the best arm over time, with Thompson Sampling reaching near-optimal selection probability faster than UCB1 and the greedy algorithm.
    b. **10,000 Iterations:** The selection probability for Thompson Sampling almost converges to 1, indicating a high level of certainty in selecting the best arm. Greedy also shows improvement, but the UCB1 algorithm still lags behind, suggesting that it may get stuck on suboptimal arms due to insufficient exploration.
    c. **100,000 Iterations:** At this point, Thompson Sampling and Greedy both exhibit high probabilities of selecting the best arm, with Thompson Sampling showing a slight advantage. The UCB1 algorithm still has not achieved a selection probability as high as the other two
3. **Chosen Arms Over Iterations**
    a. **1,000 Iterations:** The scatter plot shows that UCB1 explores more diversely than the greedy and Thompson algorithms, which tend to settle on a single arm earlier. Thompson shows moderate exploration.
    b. **10,000 Iterations:** Thompson Sampling's exploration appears to stabilize as it consistently selects the optimal arm. UCB1 shows reduced exploration over time, but the greedy algorithm's exploration is minimal.
    c. **100,000 Iterations:** Thompson Sampling has settled on the optimal arm with very few deviations. The UCB1 algorithms' choices are sporadic, suggesting that when it does

explore, it is less directed and effective in this case.

Thompson Sampling demonstrates superior performance consistently across all iterations, effectively balancing exploration and exploitation, which leads to both lower regret and a higher probability of selecting the best arm. The greedy algorithm performs better than the UCB1 algorithm, especially as the number of iterations increases, indicating that its exploitation mechanism is more effective in the long run. Figures 1, 2, and 3 for reference.

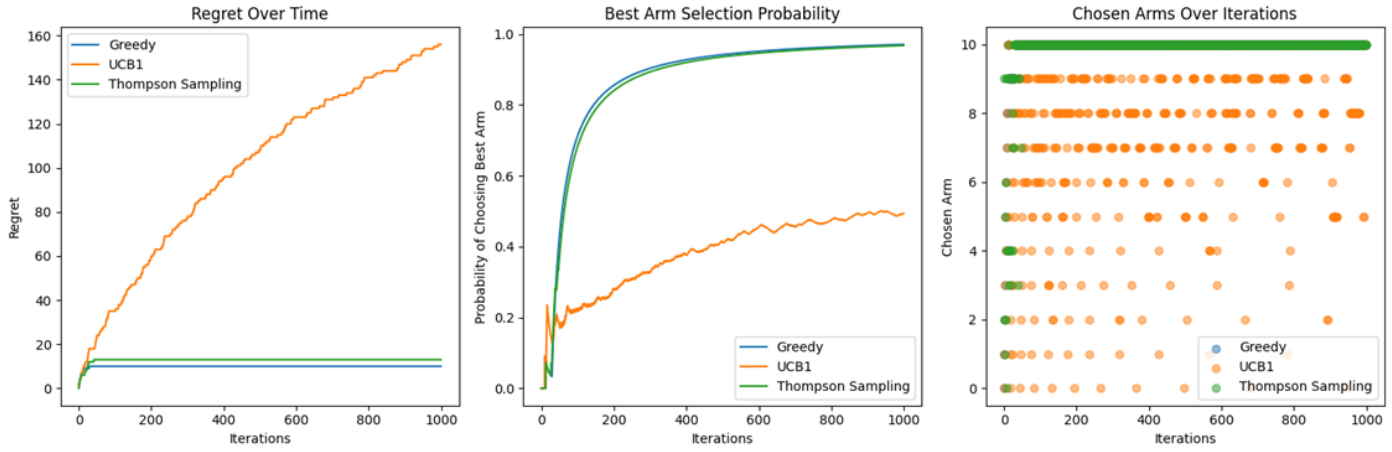

Figure 1: Single Pass result plots for **1000** iterations of three algorithms for 11 armed bandit.

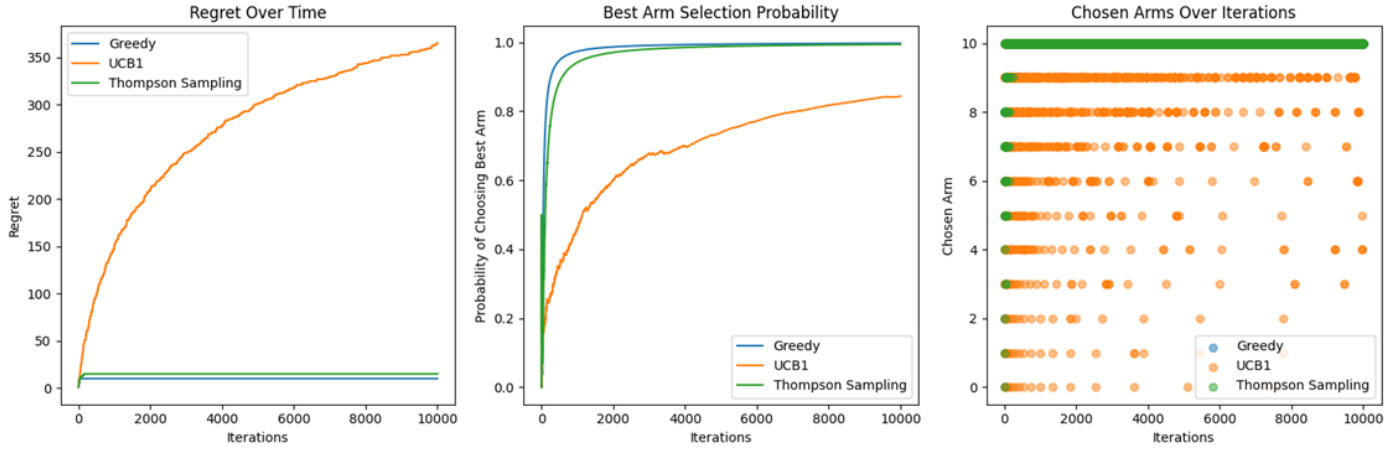

Figure 2: Single Pass result plots for **10000** iterations of three algorithms for 11 armed bandit.

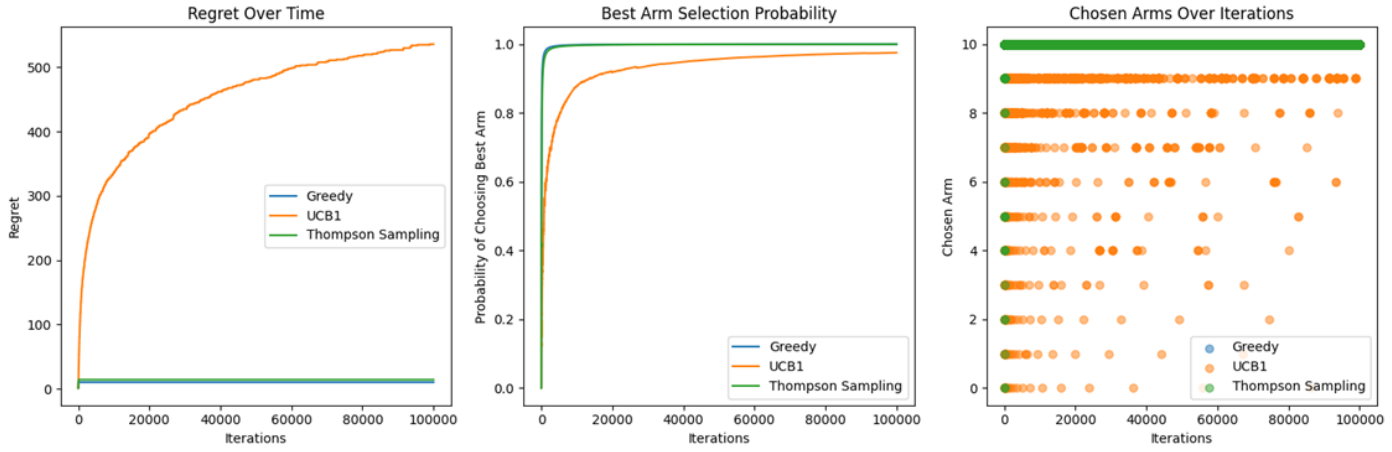

Figure 3: Single Pass result plots for **100000** iterations of three algorithms for 11 armed bandit.

100 RUNS

1. **Regret Over Time**
    a. **1,000 Iterations:** The average regret for the UCB1 algorithm is substantially higher than for greedy and Thompson Sampling. This implies that on average, the UCB1 algorithm fails to balance exploration and exploitation as effectively as the other two. Thompson Sampling shows the least regret, indicating its proficiency in quickly converging towards the optimal arm.
    b. **10,000 Iterations:** As we progress to 10,000 iterations, the disparity in regret becomes more pronounced. The UCB1 algorithm continues to accumulate regret at a much faster rate than greedy and Thompson Sampling. Thompson Sampling, once again, maintains the lowest average regret, showcasing its strength in consistently identifying the optimal arm.
    c. **100,000 Iterations:** At this level, the regret curves for Greedy and Thompson Sampling begin to plateau, indicating that both algorithms have mostly converged to the optimal arm and are accruing minimal additional regret. The UCB1 algorithm's regret, on the other hand, continues to rise significantly, which could indicate that it gets stuck in a suboptimal choice due to its exploitative nature.
2. **Best Arm Selection Probability**
    a. **1,000 Iterations:** The probability of selecting the best arm for Thompson Sampling increases sharply and outperforms UCB1 and the greedy algorithm. Greedy also shows a steep increase but doesn't reach as high as Thompson's Sampling. The UCB1 algorithm lags behind, suggesting that it may quickly converge to a suboptimal arm and explore less thereafter.
    b. **10,000 Iterations:** Thompson Sampling almost perfectly selects the best arm, as indicated by the selection probability nearing 1. UCB1 also demonstrates a high probability, though slightly less than Thompson Sampling. The UCB1 algorithm's probability is noticeably lower, reinforcing the importance of exploration in addition to exploitation.
    c. **100,000 Iterations:** Both Thompson Sampling and Greedy show probabilities very close to 1, indicating that both algorithms are highly likely to choose the best arm. The UCB1 algorithm shows improvement over time but still does not match the selection probabilities of the other two algorithms.
3. **Chosen Arms Over Iterations**
    a. **1,000 Iterations:** The scatter plot shows that Thompson Sampling and UCB1 explore more diversely than the greedy algorithm. The greedy algorithm quickly converges to fewer arms, which might not always be the optimal ones.
    b. **10,000 Iterations:** Thompson Sampling and Greedy have mostly settled on the optimal arm, as seen by the concentration of selections. UCB1 also shows a pattern of settling on fewer arms, but not as pronounced as Thompson Sampling.
    c. **100,000 Iterations:** Thompson Sampling and Greedy almost exclusively select the optimal arm, while the UCB1 algorithm's selections are still somewhat spread out, indicating that it may sometimes explore but not as efficiently.

The aggregated results from 100 experiments suggest that Thompson Sampling consistently outperforms the greedy algorithm and UCB1 in terms of both average regret and the probability of selecting the best arm. It is effective in learning the optimal strategy quickly and maintains its lead even as the number of iterations increases. Greedy also performs well, especially in longer runs, but it cannot quite match the efficiency of Thompson Sampling. The UCB1 algorithm accumulates the highest regret and has the lowest probability of selecting the best arm.

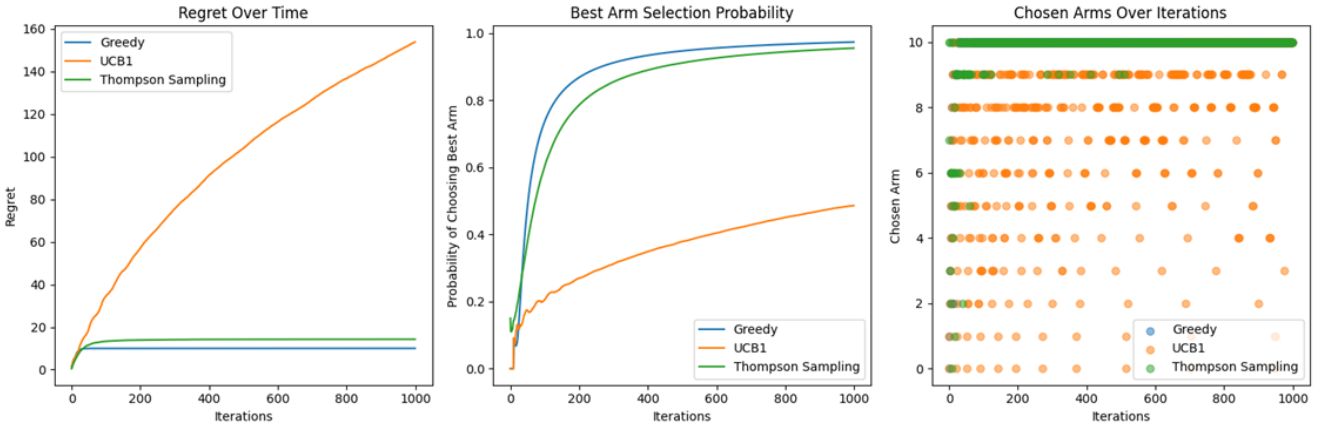

Figure 4: 100 runs result plots for **1000** iterations of three algorithms for 11 armed bandit.

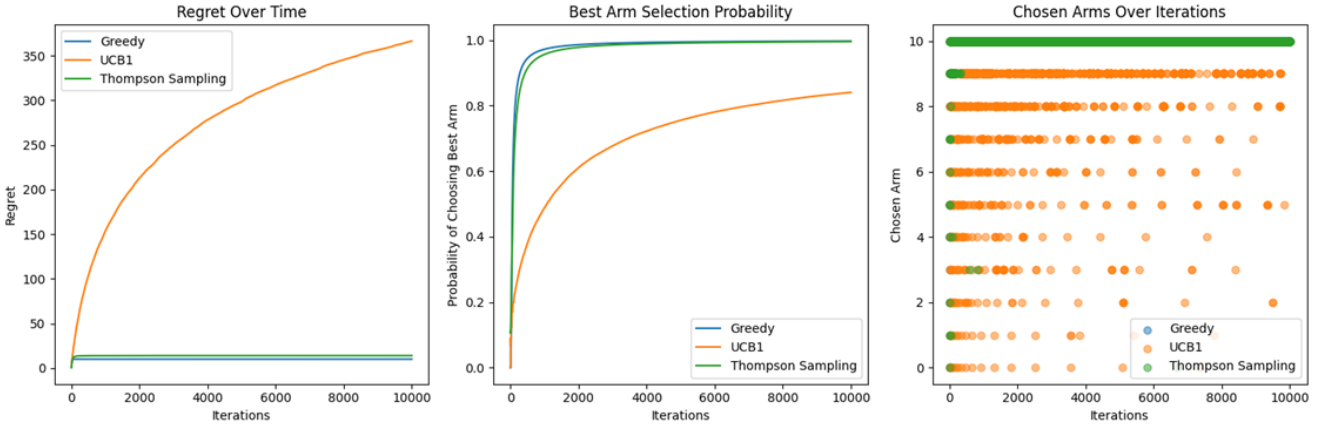

Figure 5: 100 runs result plots for **10000** iterations of three algorithms for 11 armed bandit.

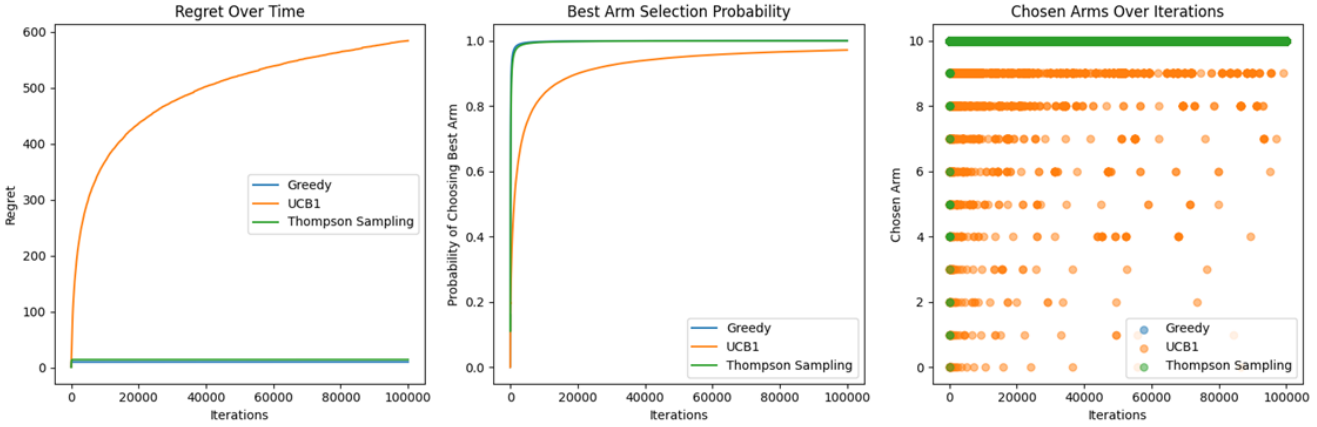

Figure 6: 100 runs result plots for **100000** iterations of three algorithms for 11 armed bandit.

CASE STUDY 2: 5 ARMED BANDITS

SINGLE PASS

1. **Regret Over Time**
    a. **1,000 Iterations:** The regret for all algorithms shows variability, with none consistently outperforming the others throughout the iterations. There are spikes and dips which indicate that the performance of each algorithm can vary greatly in any given run.
    b. **10,000 Iterations:** UCB1 and Thompson Sampling appear to stabilize compared to the 1,000 iteration case, but there are still significant fluctuations in regret, particularly with the greedy algorithm. These fluctuations can be attributed to the stochastic nature of the rewards in each arm pull.
    c. **100,000 Iterations:** The regret for Thompson Sampling shows a downward trend at certain points, which is not expected in a cumulative regret plot. This unusual behavior suggests that the regret calculation may be influenced by the randomness in a single run. Negative regret is also observed which is an anomaly and is not scientifically sound. Averaging over multiple runs is thus essential in this case
2. **Best Arm Selection Probability**
    a. **1,000 Iterations:** The probability for each algorithm rises and falls unpredictably, demonstrating the impact of randomness in each trial.
    b. **10,000 Iterations:** While Thompson Sampling appears to stabilize at a high probability, both the greedy algorithm and UCB1 show drops in performance at different points.
    c. **100,000 Iterations:** Extreme fluctuations indicate a potential issue in processing the single run.
3. **Chosen Arms Over Iterations**
    a. **Across Iterations:** In all cases, the scatter plots show which arms are chosen across iterations. Notably, the behavior of the algorithms seems consistent across different iteration scales, but without multiple runs, it's difficult to discern whether this consistency is due to the algorithm's strategy or random chance.

Averaging over multiple runs is crucial to mitigate the variance inherent in stochastic problems. The negative regret observed in the 100,000 iterations plot, which is theoretically impossible as regret should be non-negative, underscores the importance of this practice. The single-run results provide a snapshot of how the algorithms can perform under specific conditions, but they do not offer the stability and reliability necessary for a comprehensive evaluation. Averaging over multiple runs will provide a more stable and accurate measure of the algorithms' long-term performance, helping to avoid anomalies such as negative regret and providing insights that are more reflective of the expected outcomes in real-world applications.

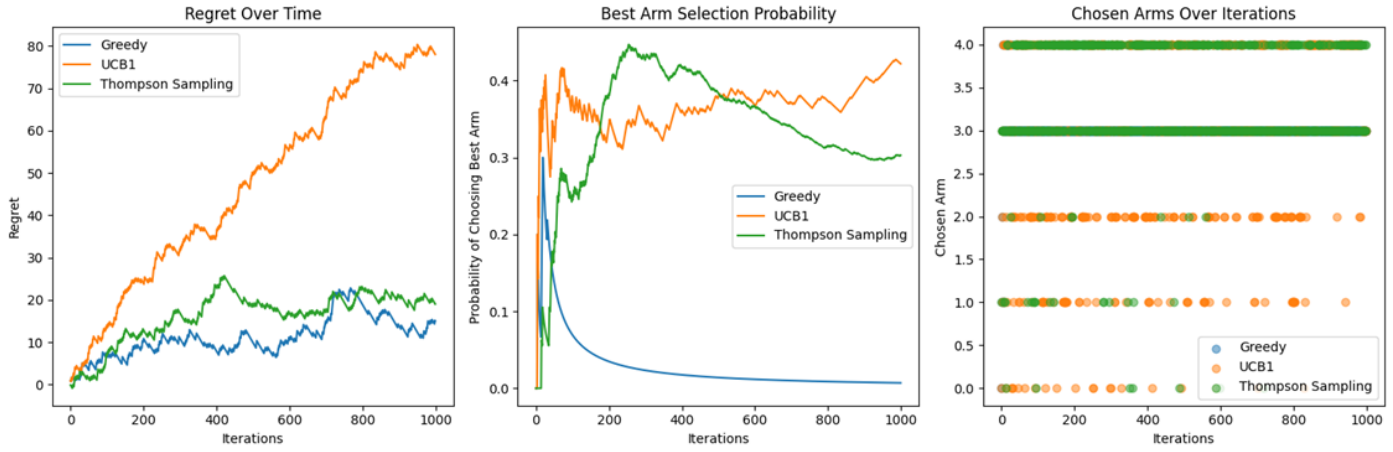

Figure 7: Single Pass result plots for **1000** iterations of three algorithms for 5 armed bandit.

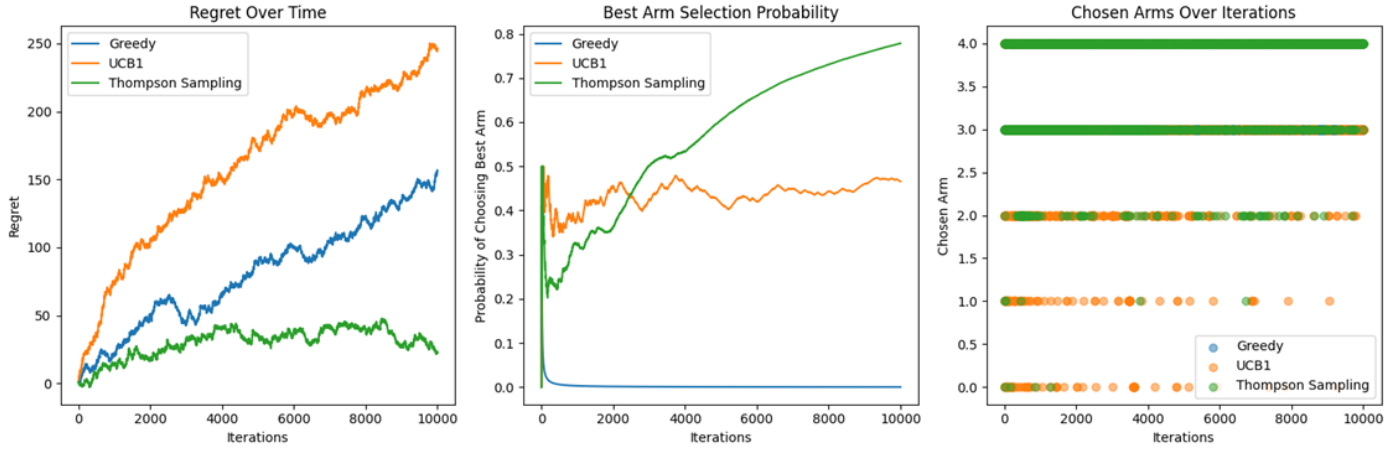

Figure 8: Single Pass result plots for **10000** iterations of three algorithms for 5 armed bandit.

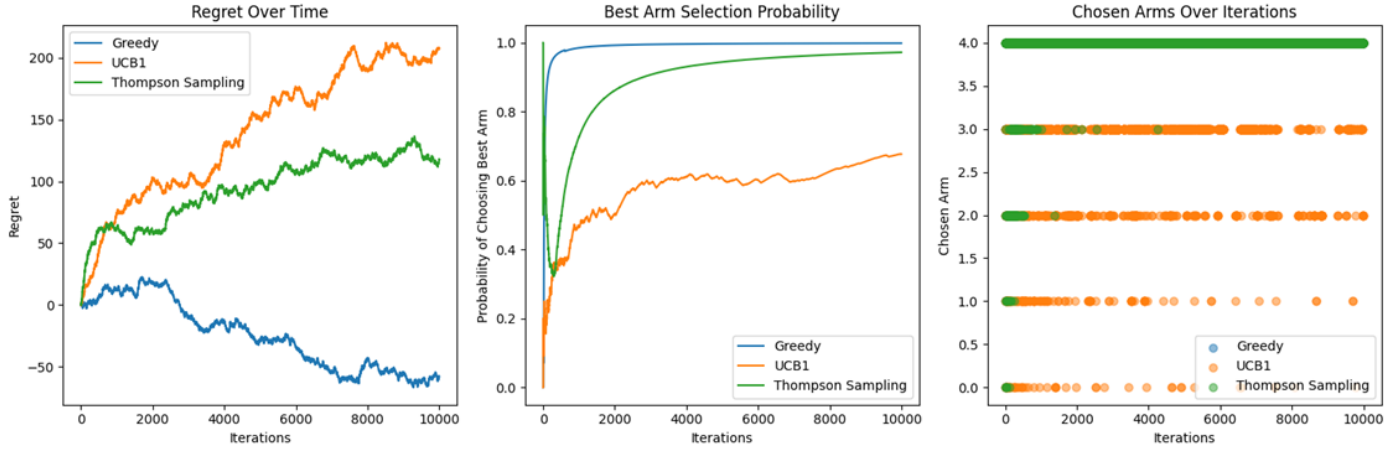

Figure 9: Single Pass result plots for **100000** iterations of three algorithms for 5 armed bandit.

100 RUNS

1. **Regret Over Time**
    a. **1,000 Iterations:** The regret curves for all algorithms are ascending, with the UCB1 algorithm showing the highest regret, followed by Greedy and Thompson Sampling. The steadier increase in regret for Thompson Sampling suggests that it performs better in terms of optimizing choices over time, while the greedy algorithm does not improve much after

initial exploration.

b. **10,000 Iterations:** The regret for the greedy algorithm continues to climb steadily, whereas UCB1 and Thompson Sampling show signs of plateauing, indicating that they are identifying the best arm more consistently as iterations increase. Thompson Sampling maintains the lowest regret, suggesting it is the most efficient in this setting.

c. **100,000 Iterations:** The regret for Thompson Sampling remains the lowest, demonstrating that it consistently and efficiently converges to the best arm. The greedy algorithm's regret increases linearly, which implies that it rarely diverts from an initially selected arm, even if it is not optimal. UCB1 shows better performance than the greedy algorithm but does not match Thompson Sampling's efficiency.

2. **Best Arm Selection Probability**

   a. **1,000 Iterations:** Thompson Sampling quickly reaches a high probability of selecting the best arm, while UCB1 gradually catches up. The greedy algorithm lags behind, indicating its tendency to stick with suboptimal choices due to inadequate exploration.

   b. **10,000 Iterations:** Both UCB1 and Thompson Sampling have high probabilities of selecting the best arm, with Thompson Sampling slightly higher. This indicates that both algorithms are quite effective over a medium number of iterations.

   c. **100,000 Iterations:** Thompson Sampling and UCB1 both achieve high selection probabilities, close to 1, indicating a very high likelihood of choosing the best arm. The greedy algorithm's probability, while plateaued at the 1,000-iteration case, is still significantly lower.

3. **Chosen Arms Over Iterations**

   a. **Across Iterations:** The scatter plots illustrate the consistency of the chosen arms for each algorithm. Thompson Sampling and UCB1 tend to concentrate on the higher-value arms, while the greedy algorithm seems to explore less, especially in the long term.

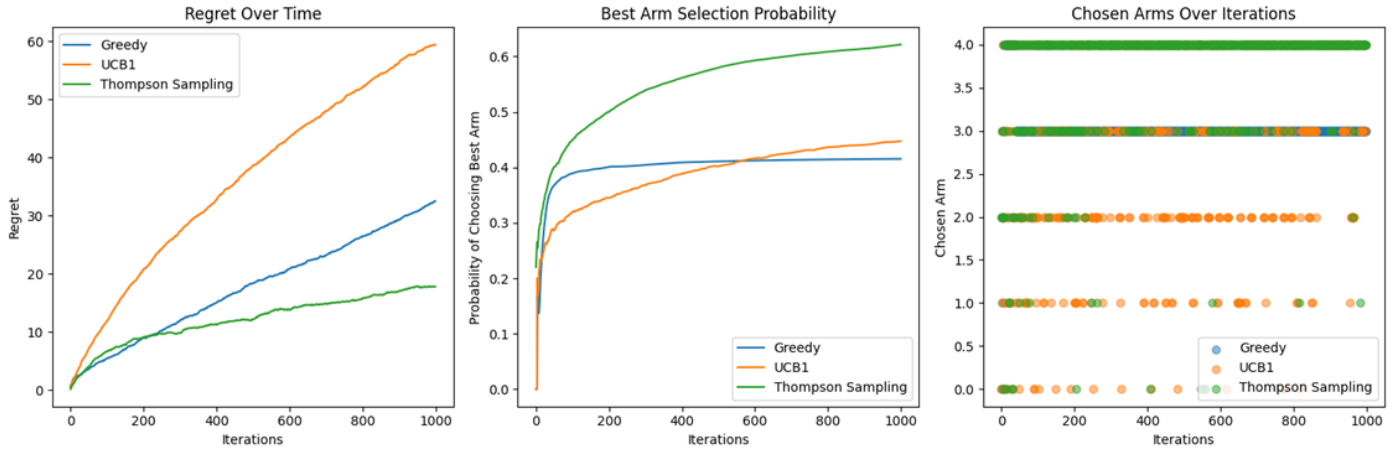

Figure 10: 100 runs result plots for **1000** iterations of three algorithms for 5 armed bandit.

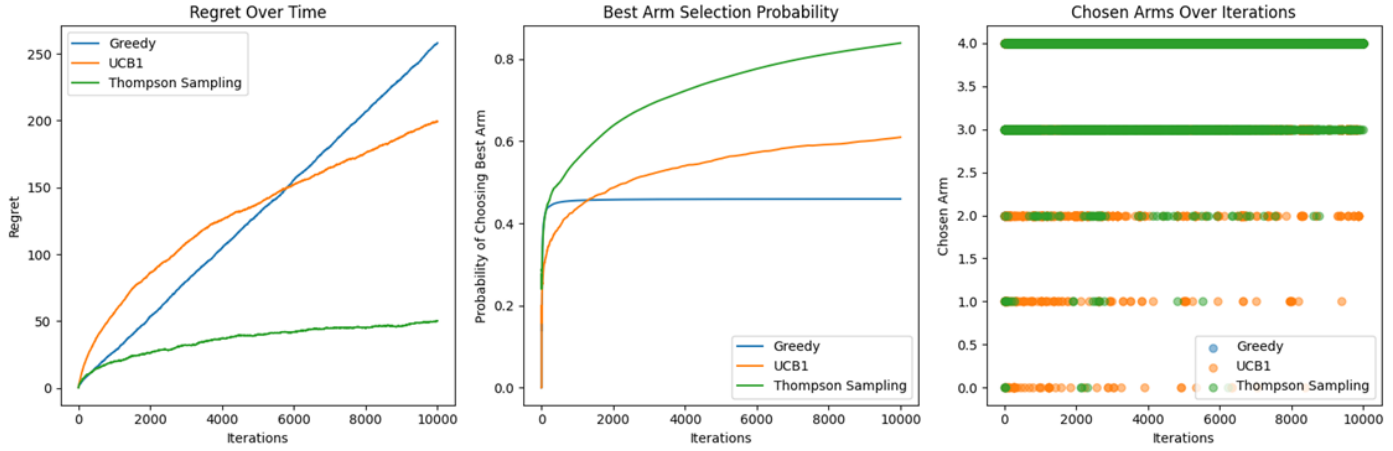

Figure 11: 100 runs result plots for **10000** iterations of three algorithms for 5 armed bandit.

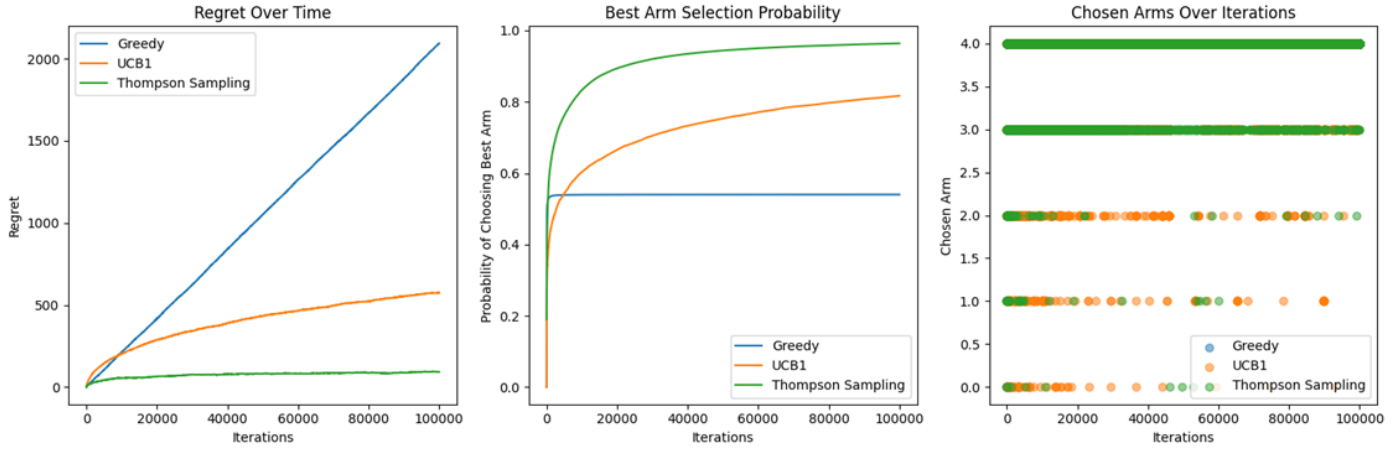

Figure 12: 100 runs result plots for **100000** iterations of three algorithms for 5 armed bandit.

DISCUSSION

We observe that the empirical properties of the three algorithms—greedy, UCB1, and Thompson Sampling—vary significantly when analyzed over short-term, medium-term, and long-term iterations and across single-run and multiple-run experiments.

Analysis over iterations

In the short term, with 1,000 iterations, we observe that Thompson Sampling tends to outperform both the greedy algorithm and UCB1 in terms of regret minimization and the best arm selection probability. This suggests that Thompson Sampling is more adept at quickly learning and exploiting the optimal strategy, even in the face of uncertainty. The greedy algorithm, on the other hand, accumulates more regret, likely due to its exploitative nature and insufficient exploration early on.

As we extend the horizon to 10,000 iterations, we observe that Thompson Sampling and UCB1 tend to converge toward selecting the optimal arm with a high probability in the case of 5 armed bandits. However, Thompson Sampling usually achieves this more rapidly than UCB1 and Greedy, indicating a more efficient exploration-exploitation trade-off. The greedy algorithm shows improvement over time but fails to reach the performance levels of the other two, supporting the idea that exploration is critical in bandit problems, especially as the number of opportunities to learn from the environment increases.

Looking at the long-term results from 100,000 iterations, we observe that the performance differences between the algorithms become more pronounced. Thompson Sampling and UCB1 show high levels of proficiency in selecting the best arm, with probabilities nearing 1. The greedy algorithm, while improving, does not exhibit the same level of performance, reinforcing the notion that it is less suited for complex tasks requiring significant exploration.

100 runs analysis

When comparing single-run experiments to those averaged over 100 runs, we observe that the latter provides a more robust and reliable analysis, smoothing out anomalies and offering a clearer view of each algorithm's performance. This is particularly evident in the case of negative regret values observed in some single-run experiments for the greedy algorithm at 100,000 iterations, which are corrected when averaged over multiple runs, reflecting the theoretical expectation that regret should not decrease over time.

The averaging process also helps to clarify the true empirical properties of these algorithms by mitigating the random fluctuations inherent to the stochastic nature of bandit problems. It becomes evident that while all algorithms improve with more iterations, the rate and consistency of improvement are critical differentiators.

Different setups of Probabilities

We observe that the varying probabilities of the arms in the multi-armed bandit problem have a significant impact on the performance of the algorithms. The setup with 11 arms ranging from 0 to 1.0 probabilities and the five-armed bandit with specific probabilities of 0.3, 0.5, 0.7, 0.83, and 0.85 offer distinct challenges that test the algorithms' ability to discern and exploit the best arm.

The observed performance differences between Thompson Sampling, Greedy, and UCB1 algorithms in the 11-armed bandit scenario might be attributed to the specific characteristics of the probability distribution. In a scenario where probabilities range from 0 to 1.0 with intervals of 0.1,

Thompson Sampling and Greedy algorithms might exploit the information about the probability distribution more effectively. Thompson Sampling, for instance, incorporates uncertainty by sampling from the posterior distribution of arm probabilities, allowing it to adapt better to the changing environment. Greedy algorithms, while being less explorative, might still perform well in situations where the probabilities are well-behaved and the algorithm can quickly converge to the optimal arm.

On the other hand, UCB1 might perform suboptimally in this context due to its inherent exploration-exploitation tradeoff. The algorithm tends to prioritize arms that have shown promising outcomes in the past, potentially neglecting exploration of other arms with uncertain but potentially higher rewards. The choice of the exploration-exploitation parameter in UCB1 might also impact its performance; a suboptimal setting could lead to a biased exploration strategy.

In contrast, in the 5-armed bandit scenario with specific probabilities (0.3, 0.5, 0.7, 0.83, and 0.85), Greedy algorithm's poor performance could be attributed to its deterministic nature. If the true optimal arm has a relatively lower probability, the Greedy algorithm might get stuck in a suboptimal choice and struggle to explore the higher probability arms. In contrast, Thompson Sampling's probabilistic nature allows it to explore and adapt to different arms, potentially leading to better performance in this context. These observations highlight the importance of considering both the algorithmic characteristics and the specific characteristics of the bandit problem when selecting the appropriate algorithm for a given scenario.

Effect of initial probabilities

The performance of these algorithms is heavily influenced by the initial probabilities assigned to each arm. If the best arm has a significantly higher probability of reward, the difference in performance between the algorithms may be less pronounced in the short term. However, as the number of iterations increases, the superior exploration strategies of Thompson Sampling become more evident. Conversely, if the probabilities are very close, the need for effective exploration becomes critical even in the short term, and the differences in performance between the algorithms become apparent more quickly.

DECLARATION

The ideas in this submission are original and were generated by **Pratish Mashankar**. ChatGPT was used as an editorial assistant, however, I take full responsibility for the originality and accuracy of the content. Case study submitted towards the partial procurement of credits for CS688 Machine Learning at GMU under Professor Sanmay Das.

COLLABORATION

I collaborated with the following students to discuss the ideas. No code base or report was shared between us.

1. Saurav Singh

REFERENCES

1. https://jamesrledoux.com/algorithms/bandit-algorithms-epsilon-ucb-exp-python/
2. https://learn.microsoft.com/en-us/archive/msdn-magazine/2019/august/test-run-the-ucb1-algorithm-for-multi-armed-bandit-problems