Reinforcement Learning Assignment-2
Multi-Armed Bandit Problem

Utkarsh Prakash
180030042

February 1, 2022

1 Experiment Setting

2 Bernoulli Reward Distribution

2.1 K=2 arm Problem

2.1.1 Greedy Algorithm

In this section, we compare the pure Greedy algorithm, ϵ-greedy with fixed $\epsilon = 0.1$ and $\epsilon = 0.01$ and variable ϵ with the following schedule:

$$\epsilon_t = min\{1, \frac{C}{t}\}$$

where $C = 10$ and t is the total number of plays. We observe the following graphs:

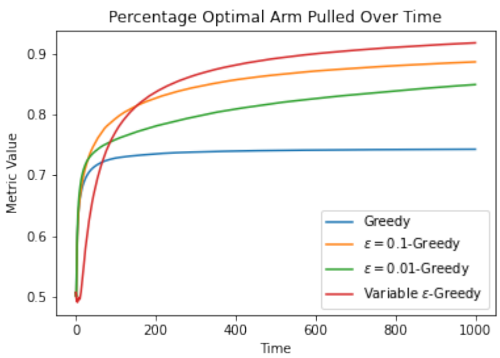

Figure 1: Percentage Optimal Arm Pulled Over Time

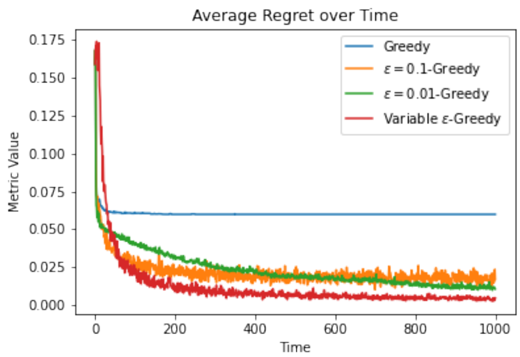

Figure 2: Average Regret over Time

Observations:

- The variable ϵ-Greedy tends to perform better than all of its counter-parts.
- In general, ϵ-Greedy performs better than pure Greedy because of its tendency to explore the arms along with exploting the current known knowledge.

2.1.2 Softmax Policy

In this section, we compare the Softmax Policy which is defined as follows:

$$\pi_t(a) = \frac{e^{q_t(a)/\tau}}{\sum_{i=1}^{K} e^{q_t(a_i)/\tau}}$$

where τ is the temperature hyperparameter. We compare the performance of Softmax Policy for $\tau = 0.01$, $\tau = 10000$ and variable τ with following schedule:

$$\tau_t = \frac{C}{t}$$

where $C = 10$. We reduce the τ overtime to control exploration-exploitation tradeoff. The graph for different metrics are obtained as follows:

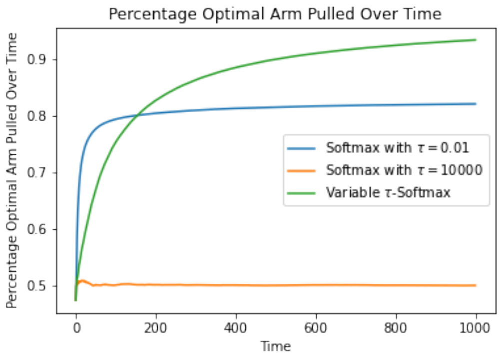

Figure 3: Percentage Optimal Arm Pulled Over Time

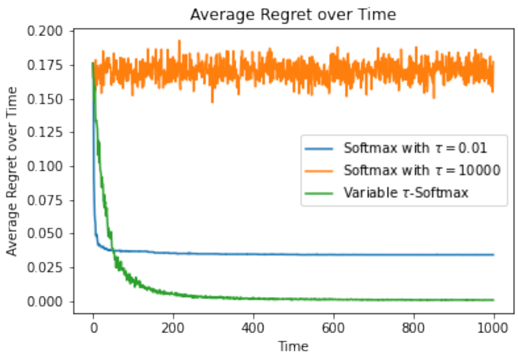

Figure 4: Average Regret over Time

Observations:

- For small values of $\tau = 0.01$, the algorithm performs poorly because in the limit when $\tau \to 0$, then the algorithm performs greedily.
- For large values of $\tau = 10,000$, the algorithm performs poorly because in the limit when $\tau \to \infty$, then the algorithm picks an arm uniformly at random. Therefore, for such large values of τ the algorithm explores at lot.
- The variable τ-Softmax tends to perform better than all of its counter-parts. This is because earlier when the value of τ is high, we tend to explore more, whereas as time progresses and we gather knowledge of different arms, we reduce the value of τ so as to exploit the knowledge that we have gathered (i.e., pull the arm which has highest estimated average reward with high probability).

2.1.3 UCB Algorithm

In this section we compare the UCB algorithm where we pick the arm which has the highest value of

$$\arg max_{a \in \mathcal{A}} \left(q_t(a) + C\sqrt{\frac{2\ln t}{n_t(a)}} \right)$$

where C is a hyperparameter which controls exploration-exploitation tradeoff. We used three different values of C (1, 100 and variable). The graphs obtained are as follows:

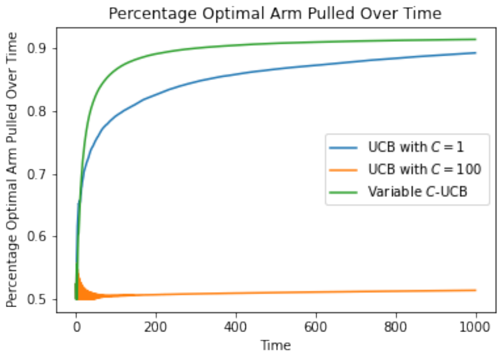

Figure 5: Percentage Optimal Arm Pulled Over Time

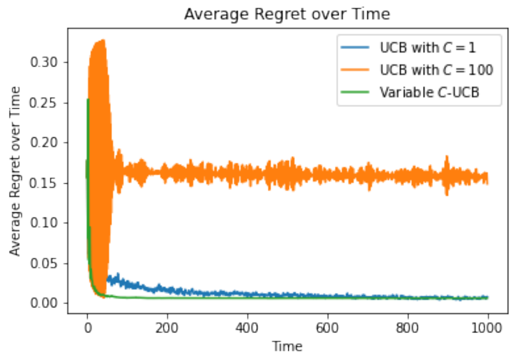

Figure 6: Average Regret over Time

Observations:

- We can see the variable-C-UCB algorithm outperforms its counter-parts.
- For large values of $C = 100$, the algorithm performs poorly because it explores too much.

2.1.4 Thompson Sampling

In Thompson Sampling, we maintain a Beta distribution prior over $q_t(a)$. We sample a value $Q_t(a)$ from this Beta distribution for each arm, i.e. $q_t(a) \sim Q_t(a)$. Now, we select the arm which has the highest value of $Q_t(a)$. The results obtained for this algorithm is as follows:

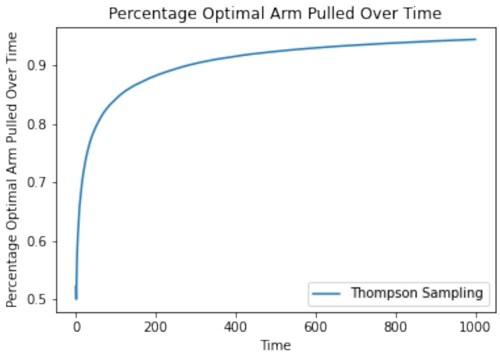

Figure 7: Percentage Optimal Arm Pulled Over Time

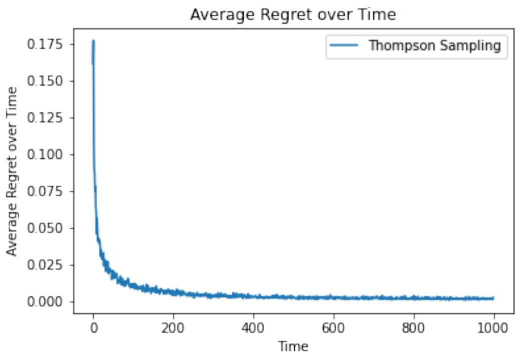

Figure 8: Average Regret over Time

2.1.5 Reinforce Algorithm

-