CS6700 : Reinforcement Learning
Programming Assignment : 1

Pragneshkumar Rana - ME17301

Indian Institute of Technology, Madras

Abstract

Experiment of 10 arm-bandit testbed was conducted to find out optimal strategy. Greedy approach, soft-max exploration, Upper confidence bound and Median elimination algorithm are discussed here. The objective of this algorithm to identify the maximum reward given set of trials by trade-off between exploration and exploitation.

Keywords: Greedy approach, Soft-max exploration, Upper confidence bound, Median elimination algorithm , Multi-arm bandit

1. (Q:1) ε - greedy:

Ref: [1] [2]

ε- greedy algorithm was implemented on 10-arm testbed. The simulation run for 1000 different steps in which each step gives reward based on the randomly picked arm and its associated probability distribution $\mathcal{N}$(Distribution associated with arm,1). Same procedure was repeated for 2000 different bandit problems in which each of the 10-arm has distribution from $\mathcal{N}$(0,1).

The expected performance of ε-greedy algorithm was obtained on different ε value. In which ε times exploration happens and $1-\varepsilon$ times exploitation was done. Experiments was conducted with values of $\varepsilon = [0, 0.02, 0.1, 1]$. As mention earlier generated graphs are average over 2000 different bandit runs.

The figure-1 first graph shows that with $\varepsilon = 0$, for fully greedy simulation ended up in sub-optimal exploitation with average reward 1 whereas it clear from the graph that maximum reward can be obtained around 1.5. With slight increase in $\varepsilon = 0.02$ increment in reward is quite high. Further increment in $\varepsilon = 0.1$ did not give much increment, which shows the trade off between the exploration and exploitation. $epsilon = 1$ shows if you do extreme level of exploration then the average reward fluctuates lot and expected reward is least.

The similar observation can be obtain from the figure-2. Second graph shows that with $\varepsilon = 0$ the optimal arm is chosen $\approx \frac{1}{3}$ times whereas non-optimal arms are picked up around $\approx \frac{2}{3}$ times. With $\varepsilon = 0.02$, given chance of exploration, chance of picking the optimal arm also increases. With $\varepsilon = 0.1$ chance of picking the best arm is around 80%. But $\varepsilon = 1$ it explores all the arm equally likely so, given 10 arm the optimal arm is picked up around 10% of the times.

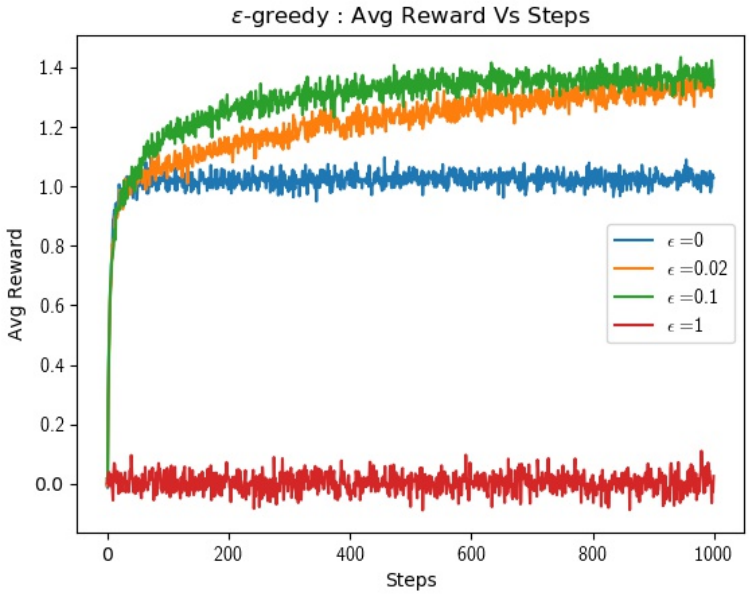

Figure 1: Average performance on REWARD using ε greedy algorithm over 2000 different 10-armed bandit problem

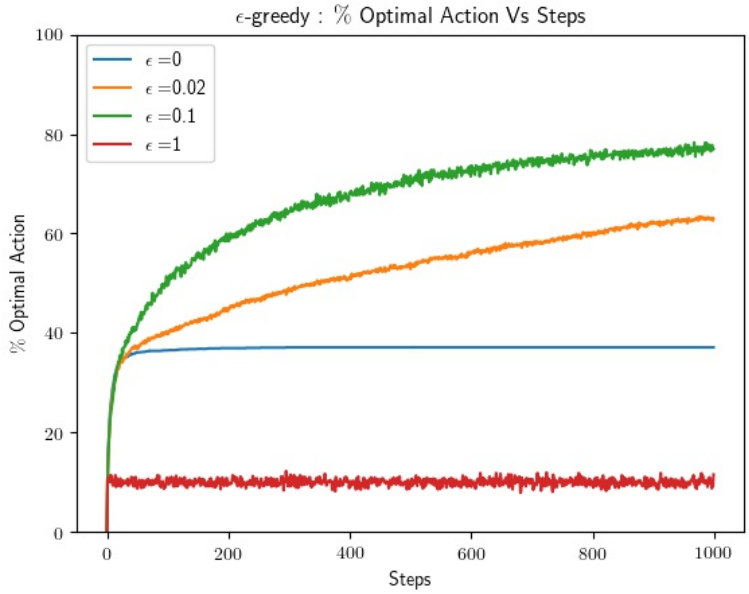

Figure 2: Average performance on OPTIMAL ACTION using ε greedy algorithm over 2000 different 10-armed bandit problem

2. (Q:2) Softmax Algorithm:

Softmax algorithm was implemented on Gibbs distribution on similar problem of 10 arm test-bed. Simulator run for 1000 steps and 2000 different bandit problem.

The empirical formula of softmax is used to find out the probability based on the reward.

$$p_i(t+1) = \frac{\exp(\frac{Q(t)}{\tau})}{\sum_i \frac{Q_i(t)}{\tau}} \tag{1}$$

From obtained probability using soft-max, random values was picked up from the distribution and associated arm-index was obtained from it. **In epsilon-greedy algorithm, in exploration stage all the arms has equal probability to get selected which is quite unfair as bad arm has same probability as best arm. To overcome this problem gibbs distribution is used to probability generated based on past rewards which gives more weightage to the arm with better rewards. so, Gibbs distribution is fair on the 10-arm testbed.**

Temperature parameter in softmax algorithm controls the randomness of the choice. Simulation has been done using temperature-$\tau = [0.02, 0.1, 0.5, 10]$. When $\tau \to 0$ softmax act as almost greedy algorithm. Whichever arm has highest reward will get highest probability.

From the figure-3 it can be observed that, for $\tau = 0.02$ the average reward obtain is around 1 which same as pure greedy algorithm reward. For $\tau = 0.1$ Average reward is maximum around 1.4 but as τ increases expected reward decreases and for very high value of $\tau = 10$ more exploration took place so average reward is least in that case.

From figure-4, it clear that for very low value τ, optimal arm was picked up around $\frac{1}{3}$ times. For slightly high value of $\tau = 0.1 and 0.5$ optimal arm was picked up aound 60% times. But very high value of temperature causes more exploration and due to high exploration all arm were equally picked up so, optimal arm was picked up around 10% times.

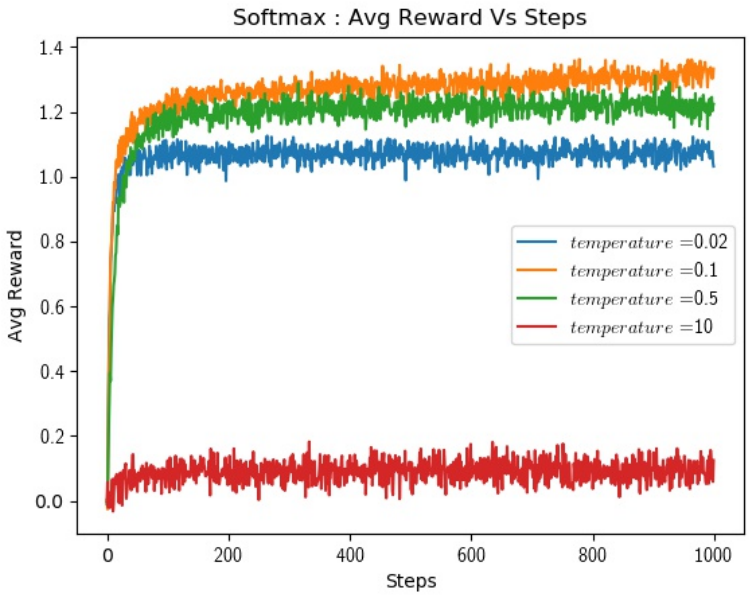

Figure 3: Average performance on REWARD using softmax algorithm over 2000 different 10-armed bandit problem

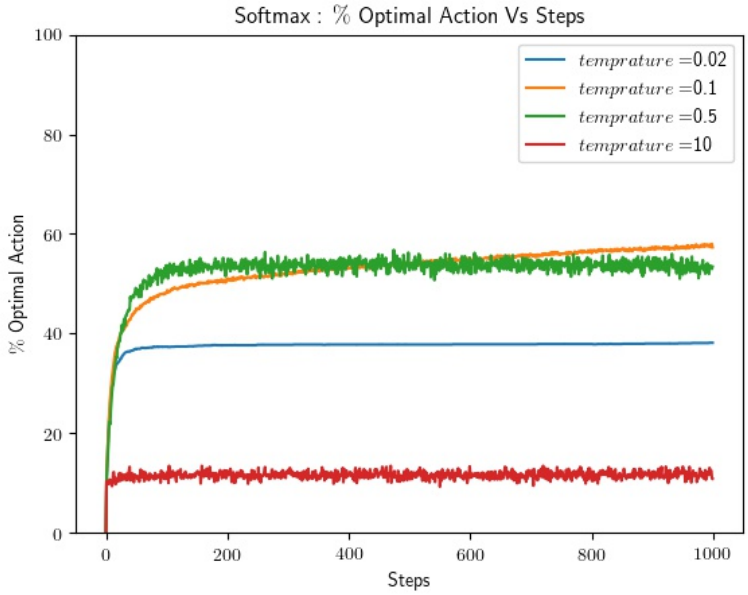

Figure 4: Average performance on OPTIMAL ACTION using softmax algorithm over 2000 different 10-armed bandit problem

3. (Q:3) UCB-1 Algorithm:

UCB-1 algorithm was implemented on 10-armed testbed, for which it run for 1000 steps and expected performance over 2000 bandit problem was compared with $\varepsilon - greedy$ and Softmax algorithm. UCB1 algorithm is a famous as used the fact that any system output has always uncertainty associated with the expected value of reward which generates need of exploration.The concept is to select non optimal arms according to their potential for actually being optimal, by considering closeness of their maximal mean estimates and uncertainties.

$$A_t = \text{argmax}\left[Q(t) + c * \sqrt{\frac{\ln t}{N_t(a)}}\right] \tag{2}$$

where,

$N_t(a)$ denotes the number of times that action a has been selected prior to time t,

$Q_t(a)$ is the mean estimate of arm a, the number c > 0 controls the degree of exploration.

In this equation square root term, determines the uncertainty associated in the estimation of a.Lower value $N_t(a)$ in denominator shows the higher uncertainty means the arm which pulled less time has poor estimate of mean with high uncertainty. so we should pull the arm a which increases $N_t(a)$ and reduces uncertainty. This guarantees that all arms will eventually be pulled, and arms with lower value estimates will be selected less frequently and optimal arm should be pulled more frequently.

from the figure-5 we can observe that UCB1 performs better than $\varepsilon - greedy$ & softmax algorithm except in the initial stage due to high variance with optimal reward it select arm randomly.

For almost all parameter this algorithm guarantee convergence unlike other algorithm in which proper parameter control is required.

From figure-1,3,5 it clear that for any moderate value of parameter algorithm performs better than $\varepsilon - greedy$ & softmax algorithm. For the worst case of parameter value 10 other algorithm did not give good expected reward but UCB give better than other even in worst case. UCB1 adjusts its upper confidence bound of each arm, which helps algorithm to do both exploration and exploitation for each arms, which makes UCB1 performance better than ε greedy & softmax algorithm.

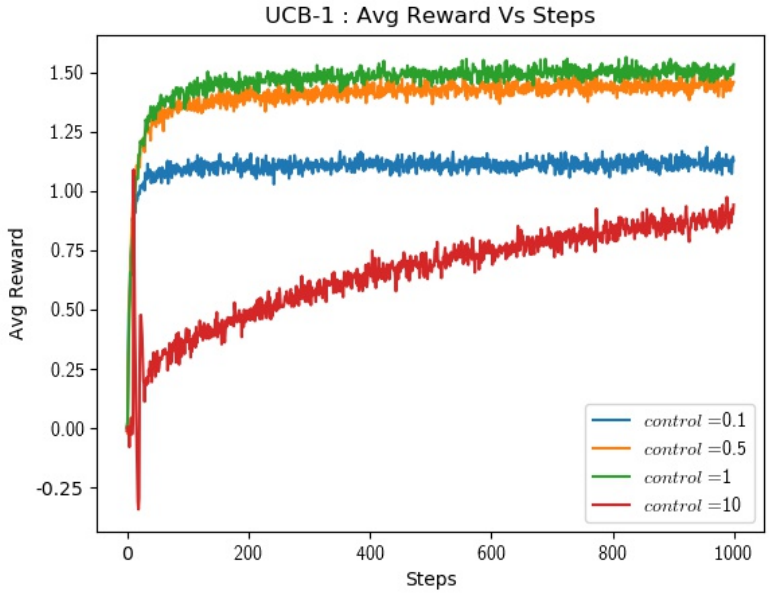

Figure 5: Average performance on REWARD using UCB-1 algorithm over 2000 different 10-armed bandit problem

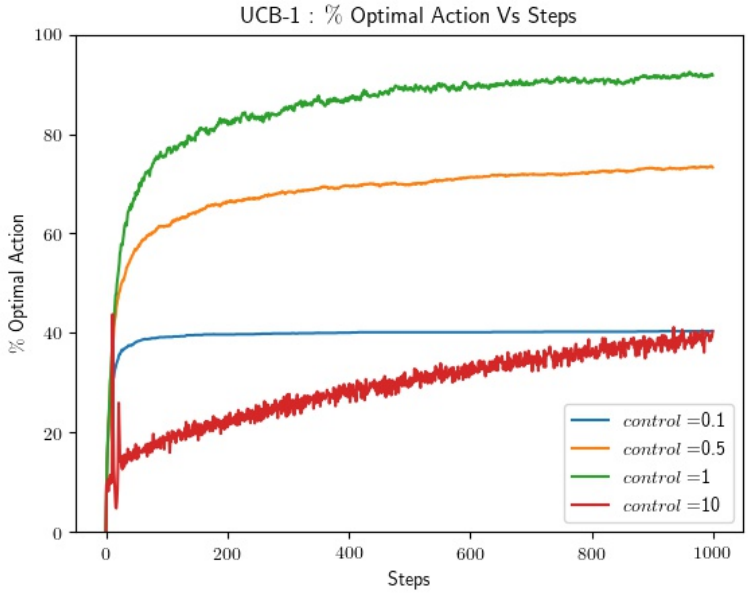

Figure 6: Average performance on OPTIMAL ACTION using UCB-1 algorithm over 2000 different 10-armed bandit problem

4. (Q:5) For 1000 Arms:

The result obtain by running the same simulation for 1000 arms are given in figures. By increasing the number of arms for same setup the obtained reward is quite high compared to other less number of arms but chances of selection of optimal arm in all cases is very less.

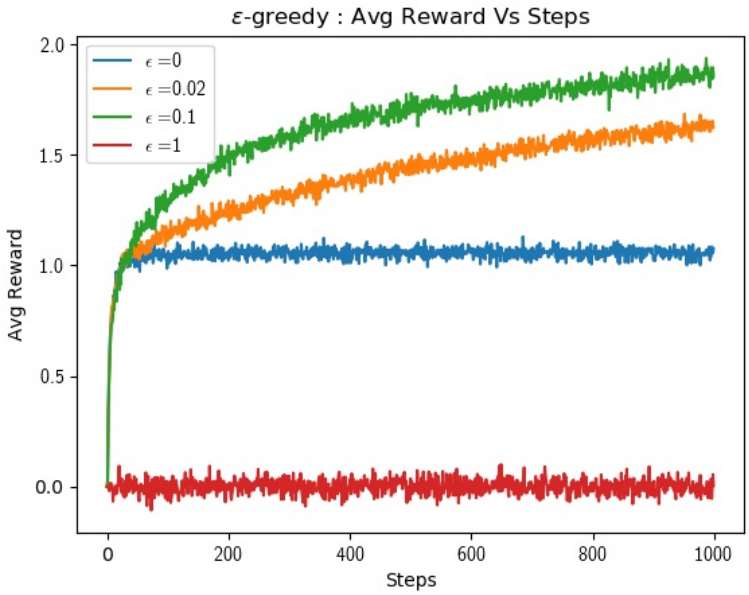

Figure 7: Average performance on REWARD using ε greedy algorithm over 2000 different 1000-armed bandit problem

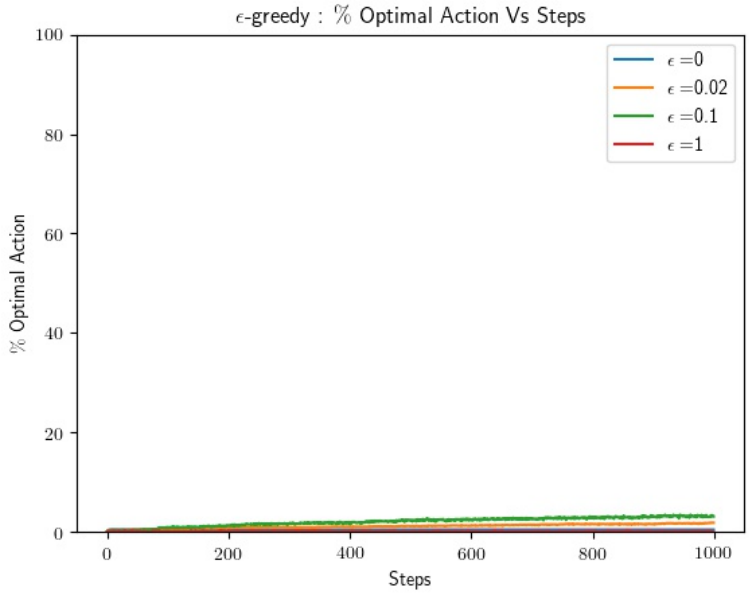

Figure 8: Average performance on OPTIMAL ACTION using ε greedy algorithm over 2000 different 1000-armed bandit problem

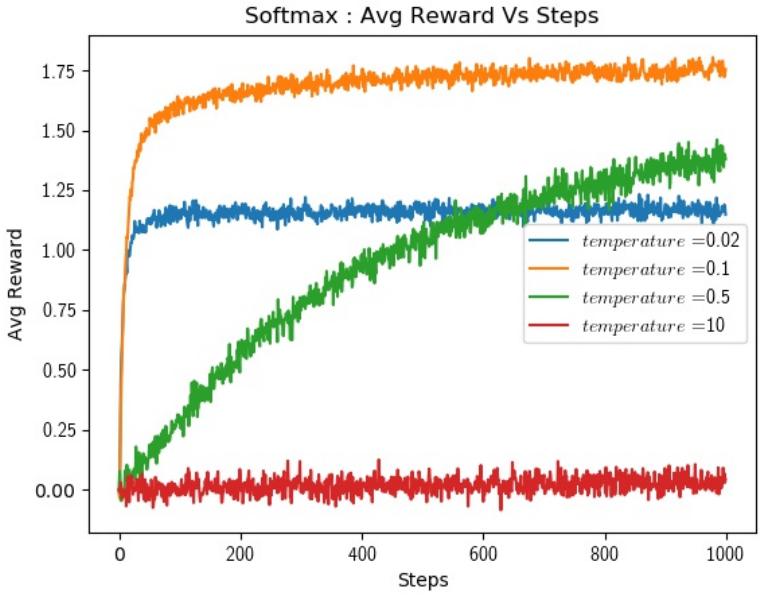

Figure 9: Average performance on REWARD using softmax algorithm over 2000 different 1000-armed bandit problem

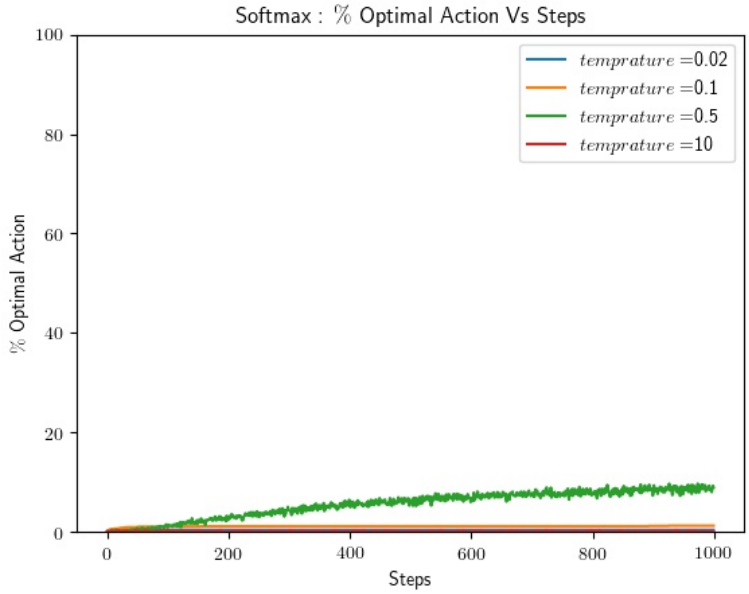

Figure 10: Average performance on OPTIMAL ACTION using softmax algorithm over 2000 different 1000-armed bandit problem

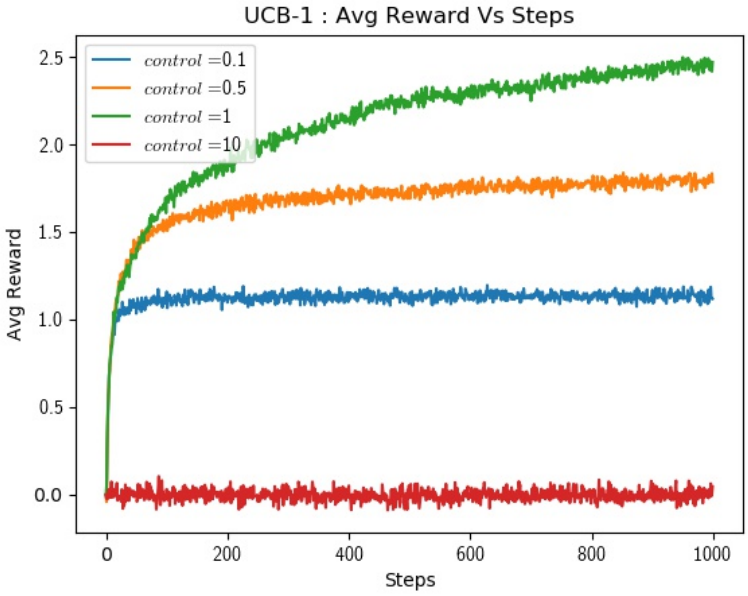

Figure 11: Average performance on REWARD using UCB-1 algorithm over 2000 different 1000-armed bandit problem

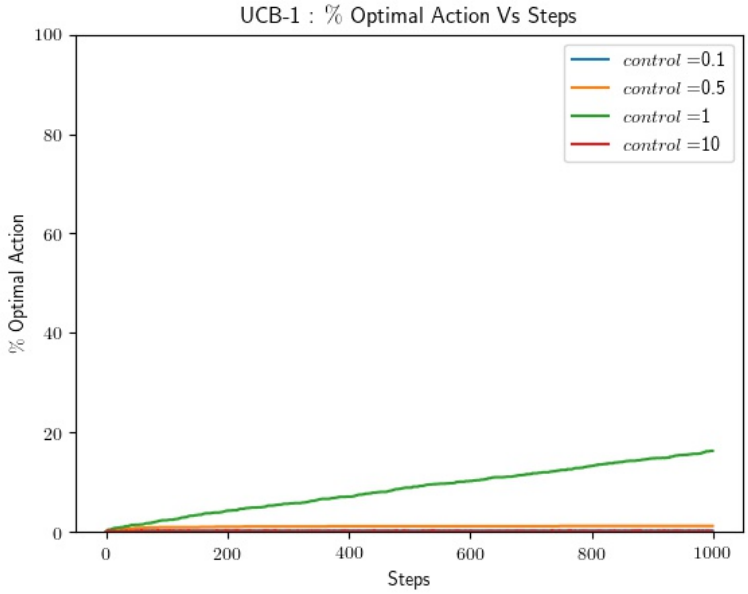

Figure 12: Average performance on OPTIMAL ACTION using UCB-1 algorithm over 2000 different 1000-armed bandit problem

It clear from the graph that obtain reward in each case in high compared with its previous setup of less arms. For 1000 arms same result interpretation is obtained as before explained. In case average reward UCB gives maximum rewards.

For almost all parameter selection of optimal arm in $\varepsilon - greedy$ algorithms is least compared to all algorithms. There is slight high chance of picking up optimal arm in soft-max algorithm

for $\tau = 0.5$ which at max pick optimal arm 10% in whole setup. UCB outperformance compared to all other algorithm. With number of increment in number of steps selection of optimal arm increases which clearly **shows that for more number of arms number of run/play required for all algorithm is quite high and UCB still performs better in all three algorithms.**

References:

[1] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. 2018.

[2] "Reinforcement learning guide: Solving the multi-armed bandit problem from scratch in python,"