First Exercise – Reinforcement Learning and Dynamic Optimazation

Student : Toganidis Nikos
Student id : 2018030085
Date : 20/3/2023

Introduction :

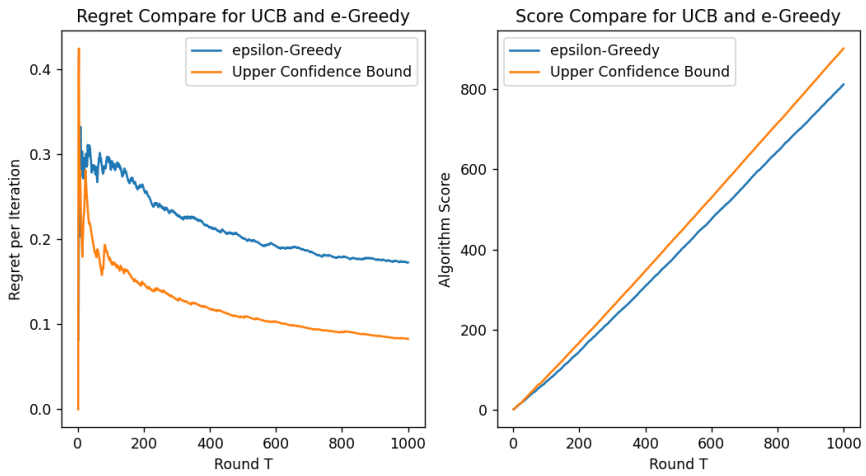
Multi-Armed bandit comes from a hypothetical experiment, where there are multiple actions (*or one-armed bandits*) with uknown reward (*payout*) and the person has to choose from them with the goal to achieve the best profit through a series of choices. The odds and rewards of every payout are uknown (*at the beggining*), and thus arises the problem of which arm and how many times should be selected. This problem is called Multi-Armed bandit problem and its solution was approached in the specific exercise with the use of algorithms :

1. **Epsilon-Greedy Algorithm**
2. **Upper Confidence Bound Algorithm (*UCB1*)**

With the application of the 2 algorithms, the quantities : **regret**, **algorithm_score**, were measured and compared, thus extracting some results, regarding the characteristics of the algorithms.

Observations :

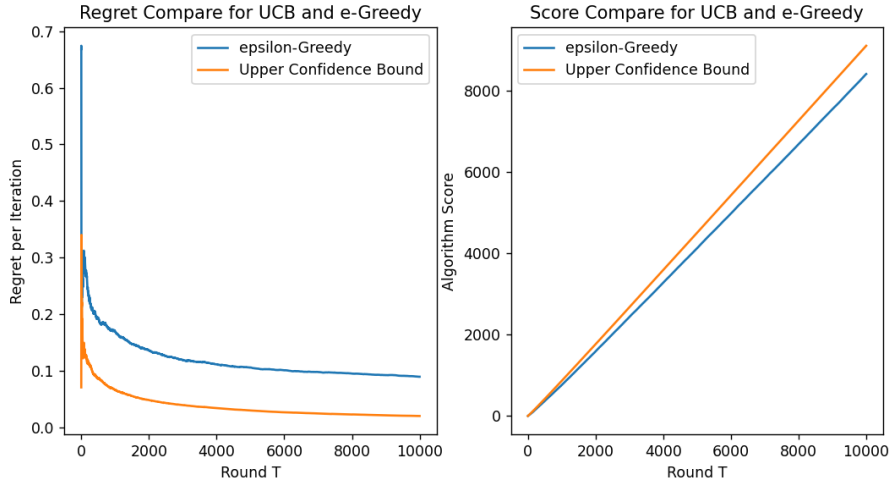
For better detection of differences of the 2 algorithms, the following graphs were ploted (*k = 10 bandits were used and the horizon was T = 1000*):

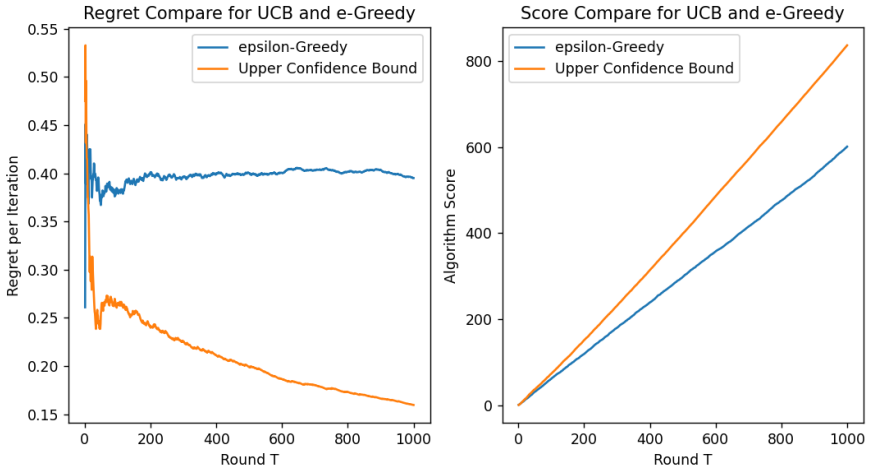


The first shows the regret through time (*descrete*). We can see clearly that in terms of respective linear regret (*the difference between expected reward of the best arm and the*

expected reward of the arm that the algorithm choosed), UCB1 algorithm is better (*we can see this better for tha cases where the k was large - we used k = 30 and 50*). Both algorithms' linear regrets grow logarithmically. This is the expected for the UCB1 algorithm. For the Epsilon greedy algorithm, the logarithmically grow was achieved by decreasing the epsilon through time. However, the difference between the linear regret of each algorithm is due to efficient exploration strategy of UCB1 algorithm.
The second shows the reward through time (descrete). It is obvious that in both cases the UCB1 algorithm is converging faster than the epsilon greedy algorithm (*we can see this better for tha cases where the k was large - we used k = 30 and 50*), due to is more efficient at exploring the arms with higher awards, while the epsilon greedy it is possible to continue select sub-optimal bandits.

It is worth presenting the graphs for k = 10 bandits (*low*) and T = 10000 (*large*) and for k=100 (*large*) and T = 1000 (*low*) respectively :





Worth to note that greedy algorithm can be suboptimal for large number of bandits