CS 747-FILA
Programming Assignment - I

Sucheta, 160040100

September 3, 2019

1 Algorithms and Assumptions

1.1 Round-Robin

The selection or sampling of arms is basically done in a round-robin fashion ie, sample each arm once and then go to the next. This is a very naive algorithm. There was nothing to assume in this algorithm as it was straight-forward.

1.2 Epsilon-Greedy

In this algorithm, an arm is sampled according to the epsilon value give. With probability **epsilon**, one explores by sampling a random arm uniformly. And with probability **(1-epsilon)** one samples the arm with the highest empirical mean.

1.2.1 Assumptions:

I generated an array called **ber_epsilon** which is basically a series of 0s and 1s sampled from the Ber(epsilon) RV. 1 denotes exploration and 0 denotes exploitation. I also created another array called **explore_arm** which is a series if arm numbers generated uniformly required to sample in case of exploration.
The three epsilon values considered are **0.2, 0.02, 0.002**

1.3 UCB (Upper confidence Bound)

This algorithm first samples all the arms one by one until every arm is pulled once. From this step onwards, it calculates the UCB of every arm in every iteration and samples the arm which has the highest UCB. There weren't any new assumptions to make in this. UCB is calculated as follows:

$$\text{UCB} = p_a + \sqrt{\frac{2*\ln t}{u_a}}$$

Where p_a is the empirical mean and u_a is the number of pulls

1.4 KL-UCB

This algorithm is very similar to the above UCB algorithm except for the expression used to sample arms after all the arms have been pulled once. In this, we basically have to find a **q** that lies between p_a and 1 such that it is the maximum q that satisfies:

$$KL(p_a, q) \leq \frac{\ln t + 3 * \ln \ln t}{u_a}$$

1.4.1 Assumptions:

I used a for loop to generate the **q** values.

1.5 Thompson Sampling

This algorithm basically samples an arm based on samples of beta distribution (**Beta(successes+1, failures+1)**) values obtained from each arm's beta distribution. Whichever arm has the highest beta value will be sampled.

1.6 Common Assumptions and Running the Code:

Ties were broken by sampling the arm which has the largest index.
I have used **python 2.7** for this assignment and have also checked the working of my bandit.sh and related codes on the sl2 computers by remote login. The **check.sh** gives the outputs on the terminal screen, to re-direct them into a file, pipe can be used.

1.7 Plots, Inferences and Explanation

There are three plots given below each depicting each of the instances given. Every plot has lines corresponding to all the algorithms. x-axis is horizon in log2 scale and y-axis depicts the average regret.

1.7.1 Observations and Explanations:

1. UCB for small number of bandits more or less is behaving poorly. When I checked why this could be a reason, it turns out that for eg, in case of 2 arms, when arm 1 is sampled, arm 2's confidence level increases so much that it is pulled next and so on. So this 'kind of' behaves like a round-robin but not exactly. When I tried with arm 2's probability equal to 1 or arm 1's lesser, it was converging to the arm with higher probability.

2. Round-robin behaved like it has to. As in, the regret for instance 1 was (0.4 + 0.8)/2 * (horizon) as it is supposed to be. The sae behavior was observed for all the instances.

3. The trend has been followed in most cases except for the Instance 1 and Instance 2 where UCB is performing better than KL-UCB

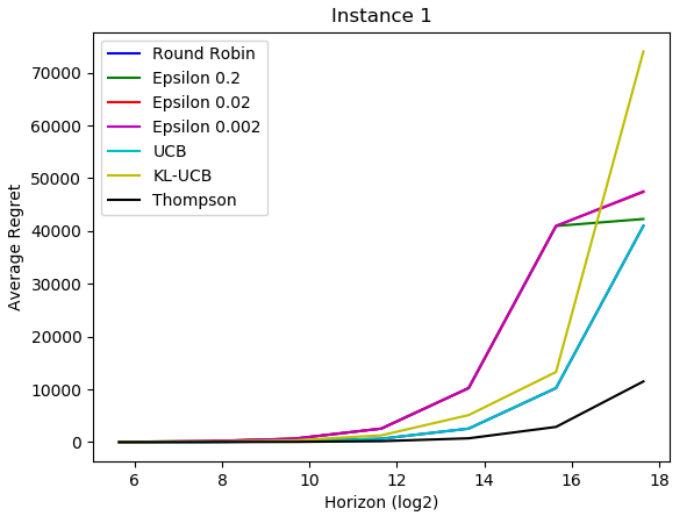
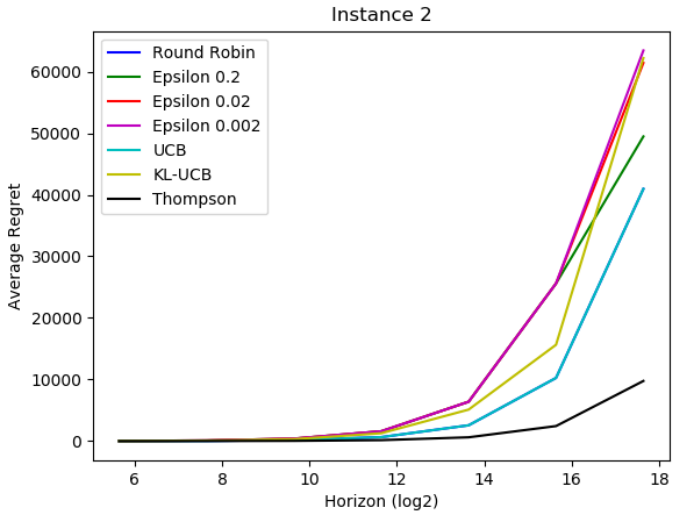


Figure 1: Instance-1



2.png

Figure 2: Instance-2

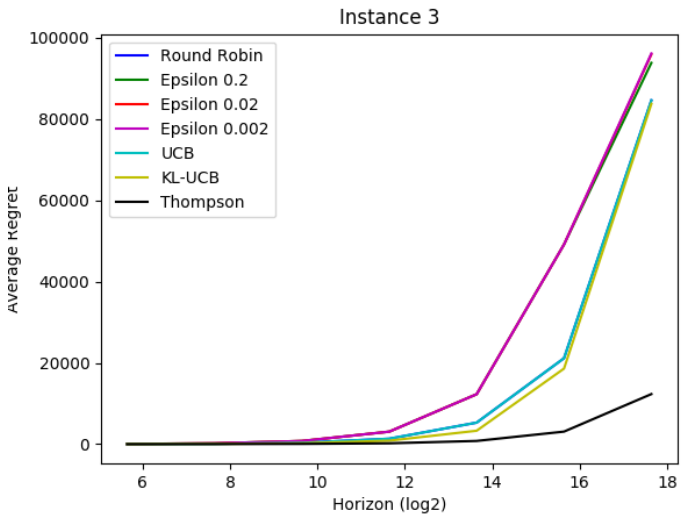


Figure 3: Instance-3