CS747 FILA
PA 1

Lalit Saini
180070030

September 2021

Contents

1 Task 1

1.1 Implementation Notes

- Epsilon Greedy 3, UCB, KL UCB and Thompson Sampling implemented in the same manner as given in the lecture slides.
- For KL UCB while finding 'q' values, the binary search stops if lower and upper pivot are within acceptable difference or if the algorithm hits max number of iterations that is 50. C was taken to be 3.
- Running time for KL UCB is the highest. It took approximately 7 mins for single seed for horizon 102400 when run on instance 3 with 25 bandits.
- For initialising, each arm was pulled for one time. Arm to play was chosen using `np.argmax()` which breaks tie by returning min index element.

1.2 Plots

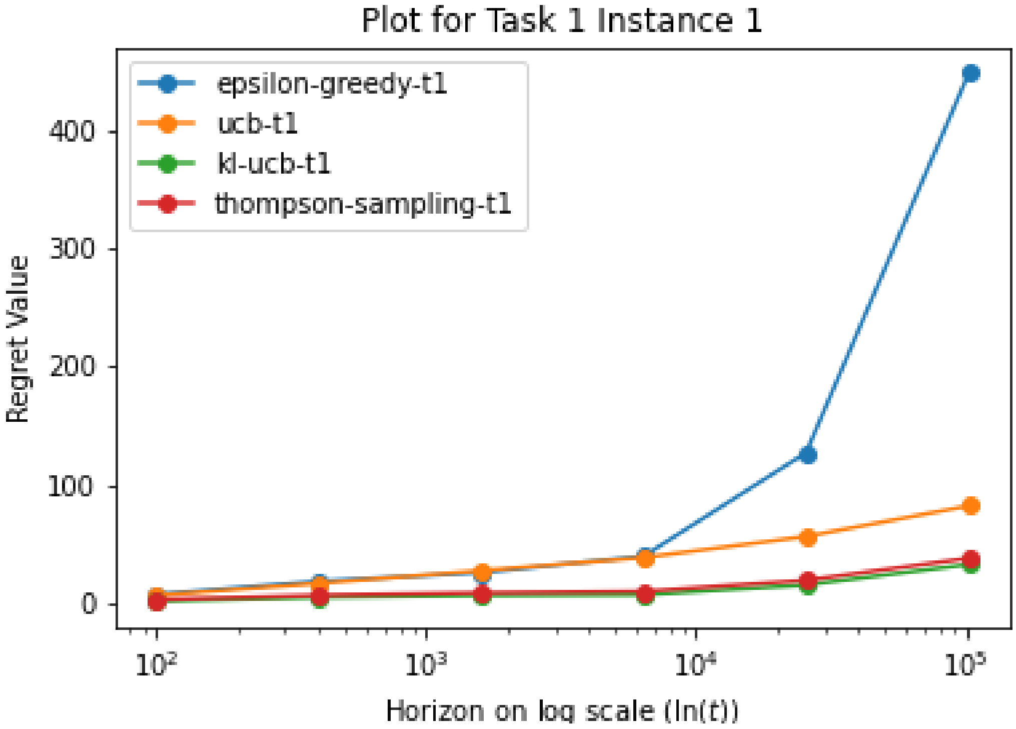

Figure 1: Performance of algorithms on bandit instance 1

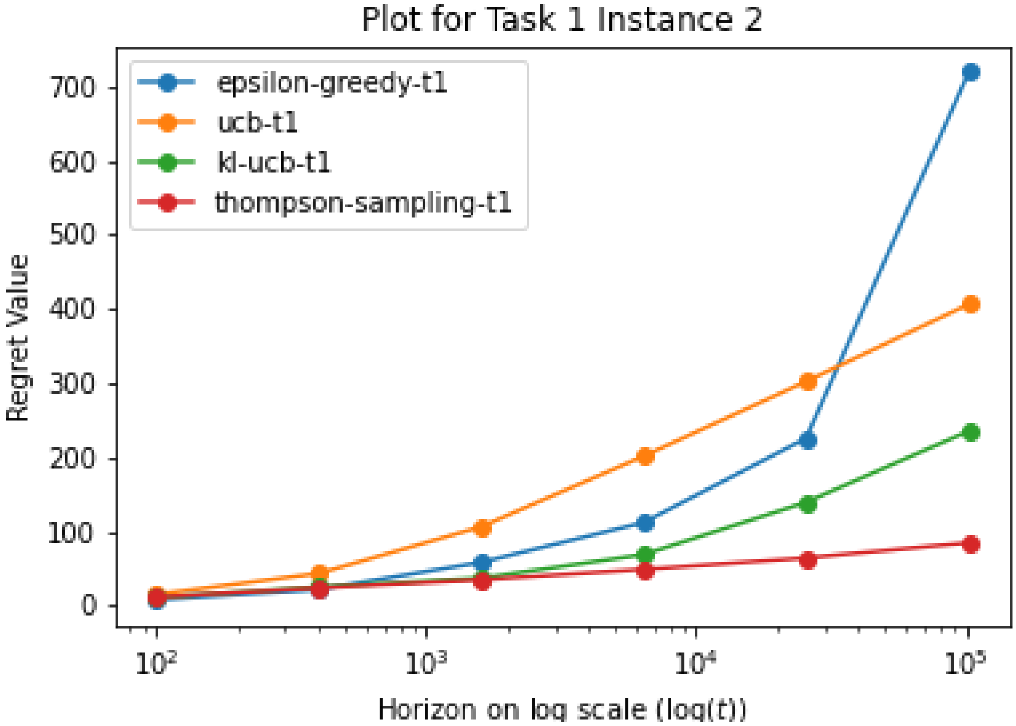

Figure 2: Performance of algorithms on bandit instance 2

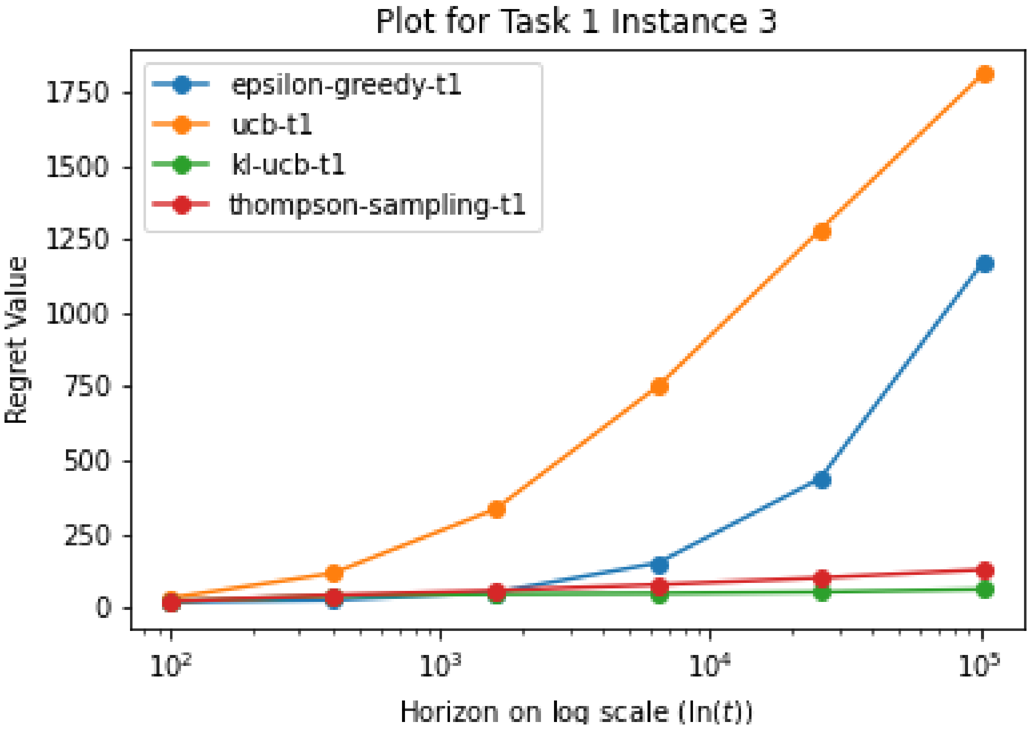

Figure 3: Performance of algorithms on bandit instance 3

2 Task 2

2.1 Observation

- For all instances using C=0.3 gave the best regret among the given scale values.

2.2 Plot

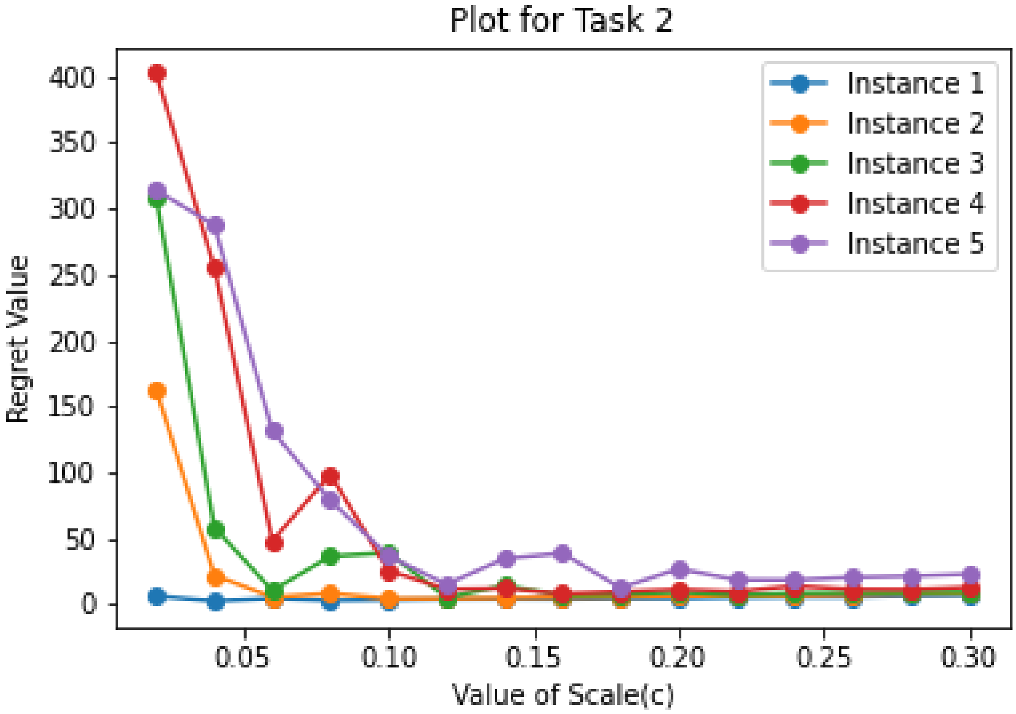

Figure 4: Performance of UCB for different scale values on different instances

3 Task 3

3.1 Modified Algorithm

I adapted the idea of UCB and extended it to finite support of rewards. For every arm the empirical mean of every reward was calculated. This empirical mean is used to calculate empirical expectation of reward.

$$\mathbb{E}[\hat{R_i(t)}] = \sum_{j=1}^{j=N} R_j \hat{P_{ij}(t)}$$

where N = number of rewards, R_j = value of j^{th} reward, $\hat{P_{ij}(t)}$ = empirical mean of j^{th} reward for i^{th} arm and $\mathbb{E}[\hat{R_i(t)}]$ = empirical expectation of award for i^{th} arm.
The exploitation part of UCB for an arm was defined using the empirical expectation , keeping the exploration part the same. UCB for i^{th} arm is given

as,

$$UCB_i(t) = \mathbb{E}[\hat{R_i(t)}] + \sqrt{\frac{C\ln(t)}{u_i(t)}}$$

where $u_i(t)$ is number of time arm i played till time t.
Taking the results of previous task, where we found out that C=0.3 worked well enough for all instances. I used the same values of C=0.3 in this algorithm too. At any time the arm with highest UCB values is played,

$$arm(t) = argmax_i UCB_i(t-1)$$

3.2 Plots

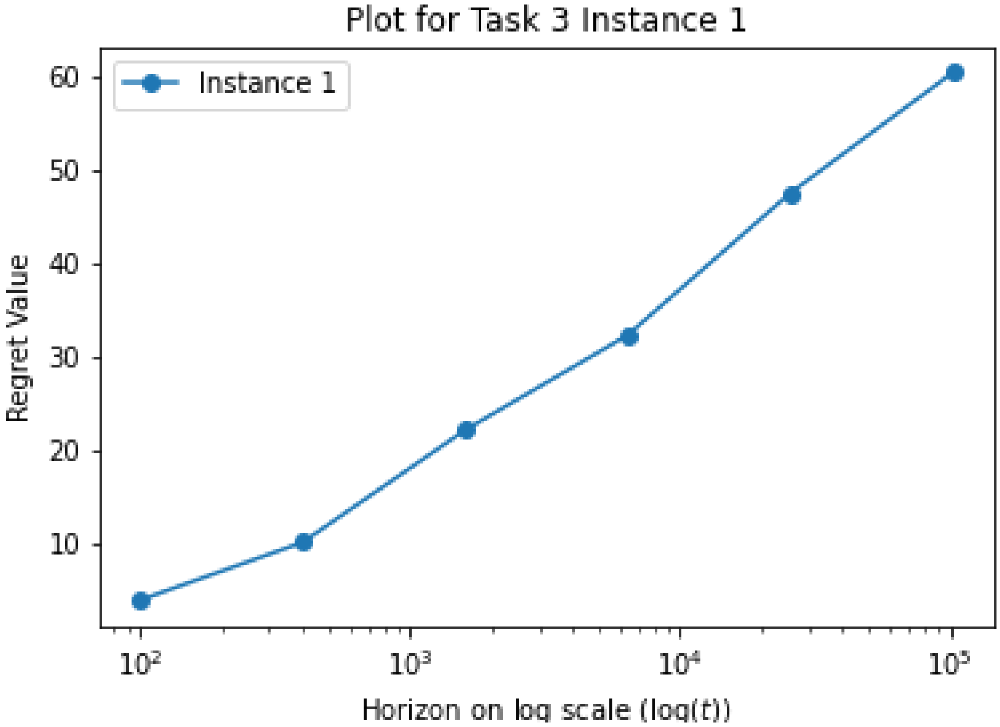

Figure 5: Performance of modified UCB on different instances

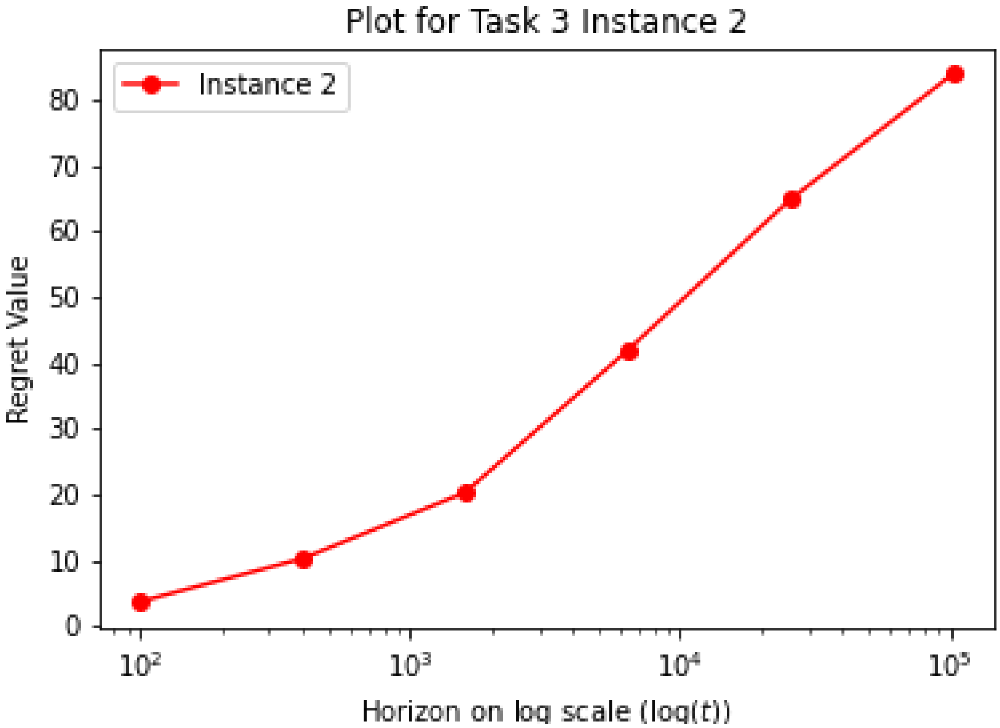

Figure 6: Performance of modified UCB on different instances

It is visible that the algorithm has regret of order $O(log(T))$.

4 Task 4

4.1 Modified Algorithm

For this task I implemented a modified Thompson sampling with success event as the event when reward greater than threshold observed else the event is failure.

We have a threshold above which the reward is high and below which or equal to that the reward is low. The probability of low is the sum of probability of getting rewards less than equal to threshold. Similarly by adding probability of rewards greater than threshold we can get probability of high event. Calculating probabilities for arm i is given as,

$$P(High_i) = \sum_{j, Reward_j > threshold} P_{ij}$$

$$P(Low_i) = \sum_{j, Reward_j <= threshold} P_{ij}$$

We will now sample from beta distribution for different arms, and will play the arm with largest sample value.

$$arm_i(t) = argmax_i beta(success_i + 1, failures_i + 1)$$

where $success_i$, $failures_i$ are number of success events and failure events respectively for arm i.
As regret we will be using MAX_REGRET , which is defined as

$$Regret = MAX\_REGRET = MAX\_HIGHS - HIGHS$$

MAX_HIGH for an instance will be $Horizon * (max_i P(High_i))$ and HIGHS are returned by the algorithm.

4.2 Plots

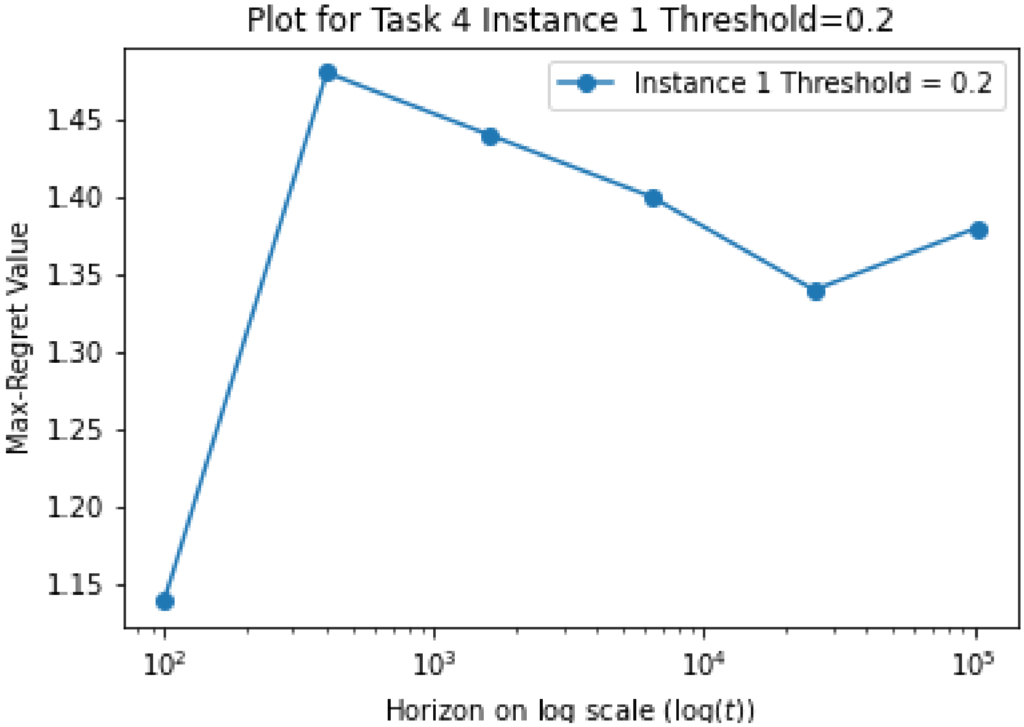

Figure 7: Performance of modified Thompson Sampling on instances 1 and 0.2 threshold

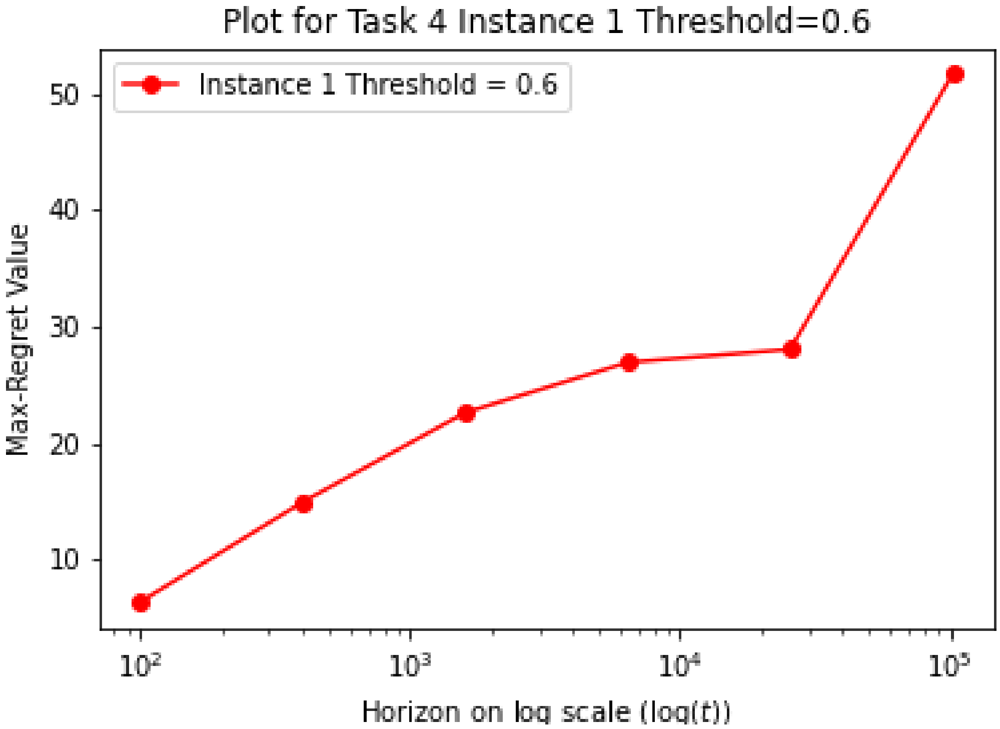

Figure 8: Performance of modified Thompson Sampling on instances 2 and 0.6 threshold

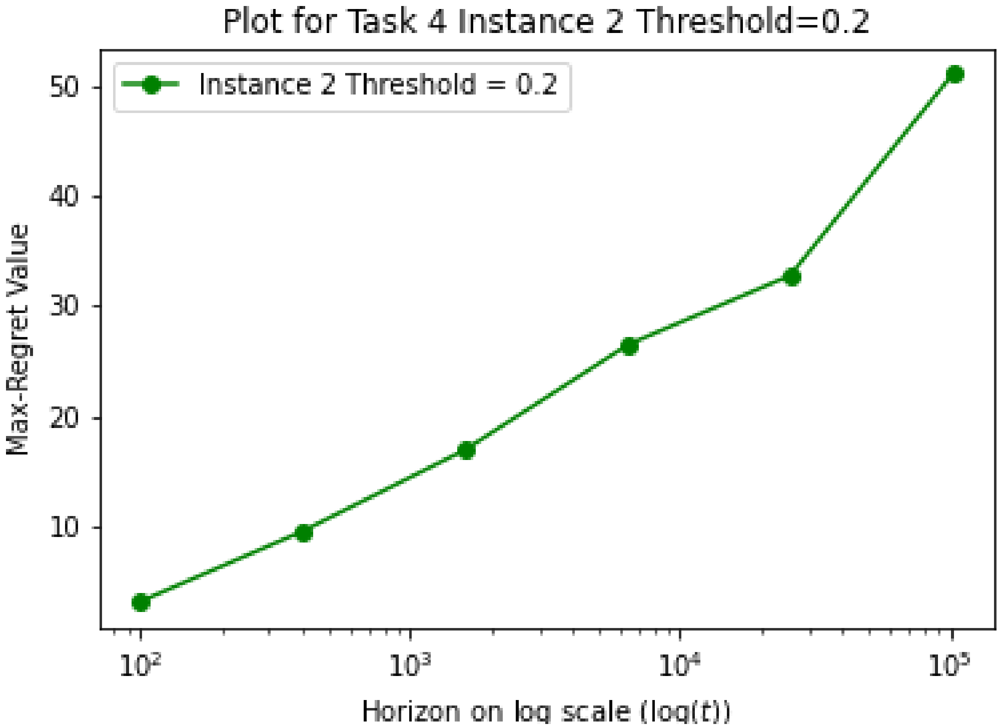

Figure 9: Performance of modified Thompson Sampling on instances 1 and 0.2 threshold

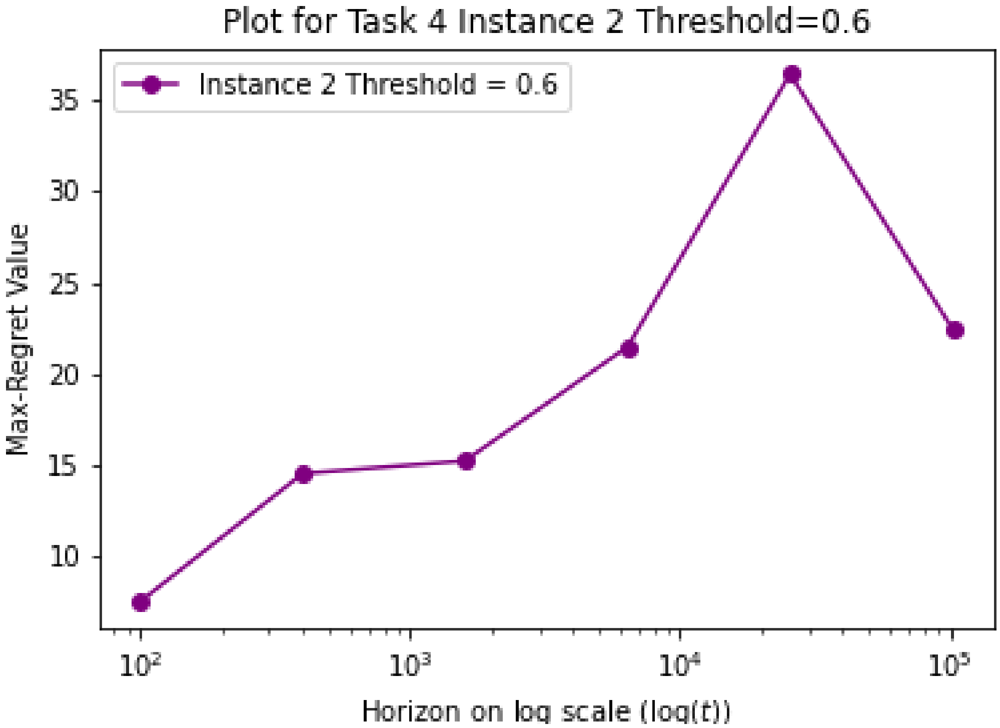

Figure 10: Performance of modified Thompson Sampling on instances 2 and 0.6 threshold

Regret values in every plot is averaged over 50 random seeds