CS747 - Assignment 1

Kalpesh Krishna
140070017
kalpeshk2011@gmail.com

1 Implementation

All algorithms were implemented in C++ building upon the basecode provided. An initial class `BaseAlgorithm` has been created in `algorithm-data.h`. Each algorithm inherits from this class. The main implementations are located in `algorithm-data.cpp`.

1.1 Epsilon Greedy

Using `gsl_rng_uniform`, an initial Bernoulli distribution for ϵ is sampled. For the first pull, this choice is ignored and the system explores randomly. For the exploration step, `gsl_rng_uniform` has been used again. 10 values of $\epsilon \in \{0.1, 0.2, ...1.0\}$ were tried, and the best value $\epsilon = 0.1$ was chosen. Further reduction of ϵ should reduce long term regret.

1.2 UCB

The UCB implementation exactly follows *Auer et al. 2002* implementation with a initial round robin sampling round and subsequent maximization of the UCB objective.

1.3 KL-UCB

As outlined in the *Garivier et al. 2011* paper, I've taken $c = 0$. Taking $c = 3$ resulted in worse long term regret. To approximate the value of q, a binary search in $q_a \in [\hat{p}_a, 1]$ (where KL-Divergence a strictly increasing function) was conducted. This search concluded when

$$0 \leq \frac{1}{N_a}(\log T + c \log \log T) - KL(\hat{p}_a, q) \leq 10^{-6}$$

The threshold value and c can be adjusted in `algorithm-data.h`

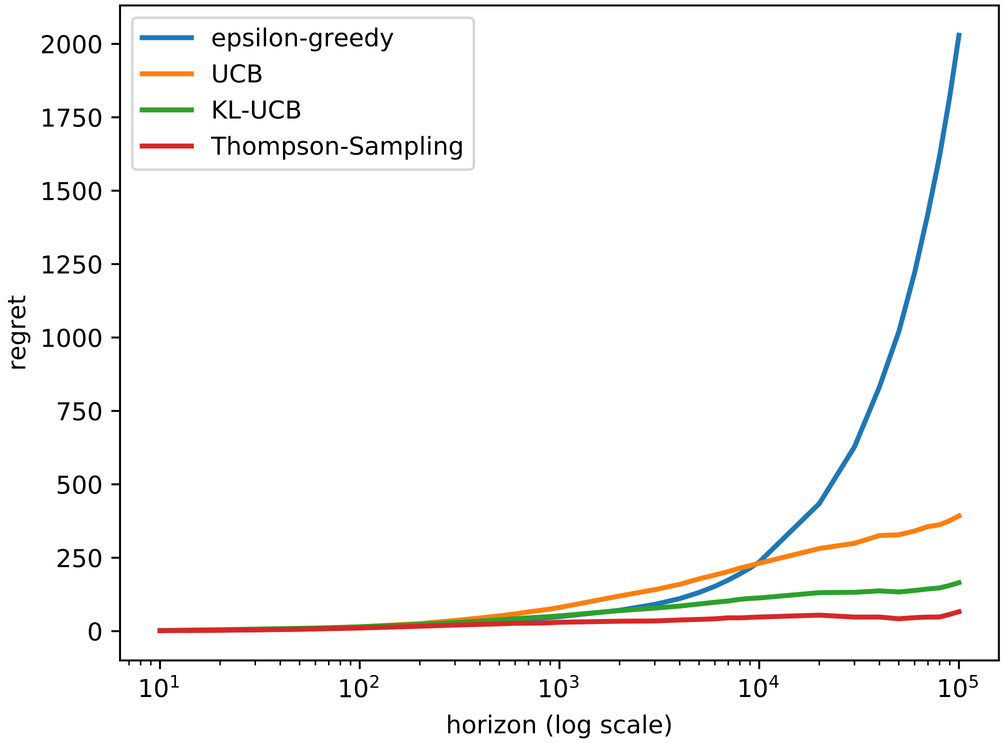
1.4 Thompson Sampling

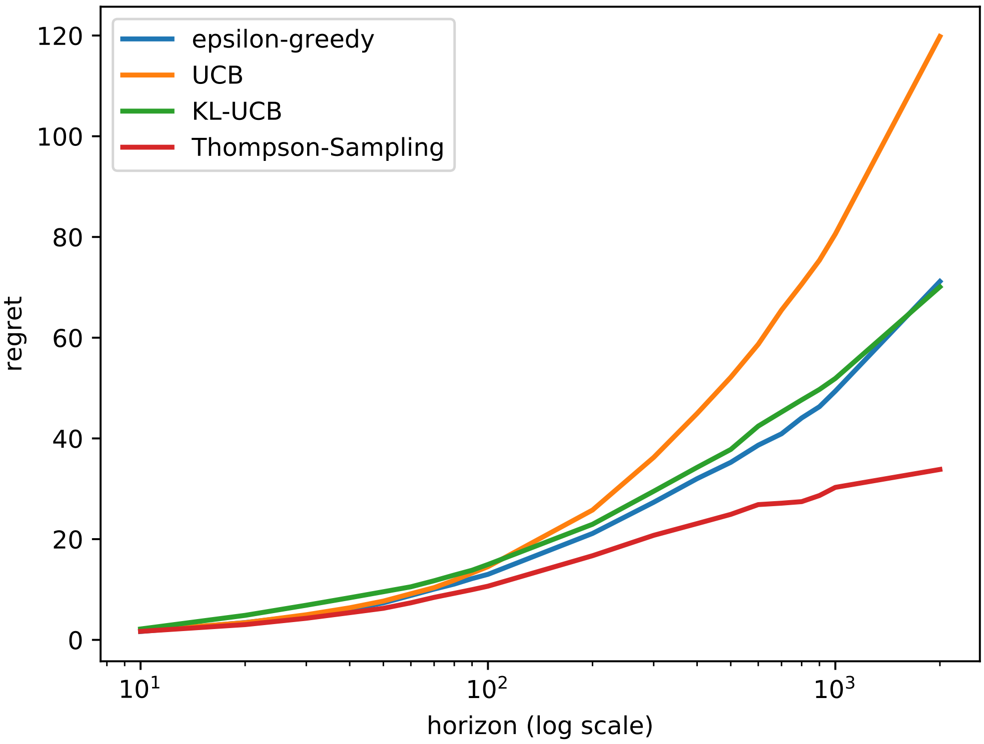
The `gsl_ran_beta` function in the GSL was used to sample from the beta distribution.

2 Results

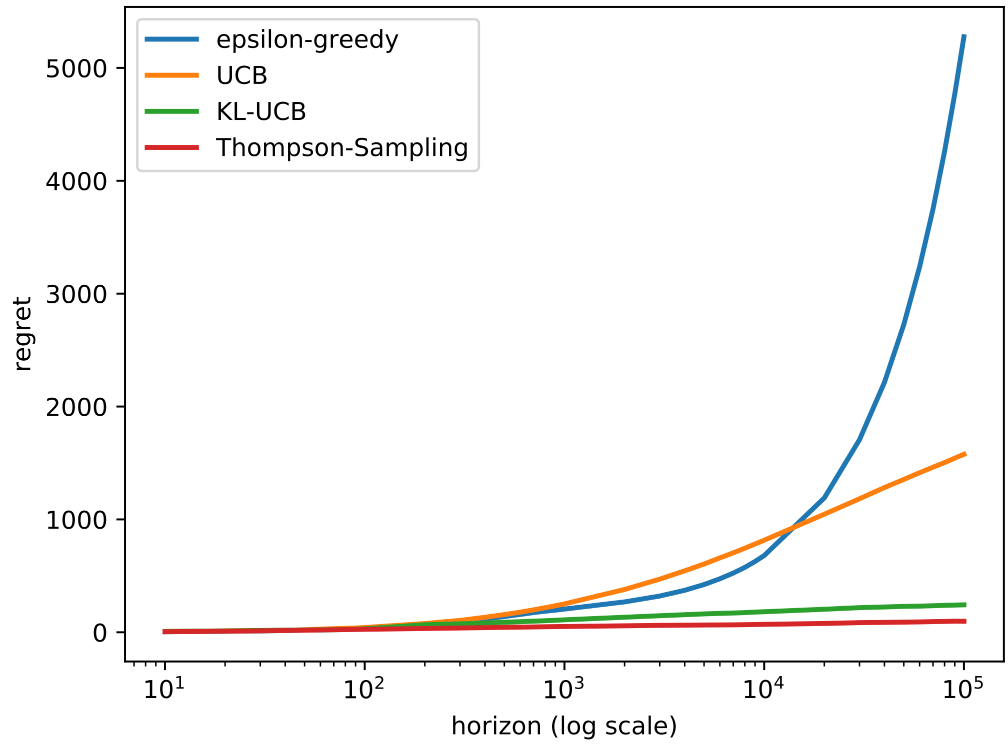
Each of the curves below consist of 37 horizon points.
$h = \{10, 20, ...90, 100, 200, ...900, 1000, 2000, ...9000, 10000, 20000, ...90000, 100000\}$. For each horizon point, 100 different random seeds were taken. The curves represent an average across these runs.
The results are consistent with the discussed theory. Since the `epsilon-greedy` algorithm has a linear regret, it's seen as an exponentially increasing function in a logarithmic scale. The other algorithms enjoy a roughly linear curve (logarithmic regret), with optimal results seen in the order, `Thompson-Sampling` > `KL-UCB` > `UCB` as we had initially expected. As expected, the regret is larger in the 25-arms case, since it's harder for the system to find the optimal arm.
It was surprising to see the zoomed in curve for a horizon upto 2000, with `UCB` and `KL-UCB` doing worse than `epsilon-greedy`, indicating larger constants with $O(\log(T))$ than $O(T)$. It was also surprising to see occasional dips in regret, which I later realized are possible stochastically.

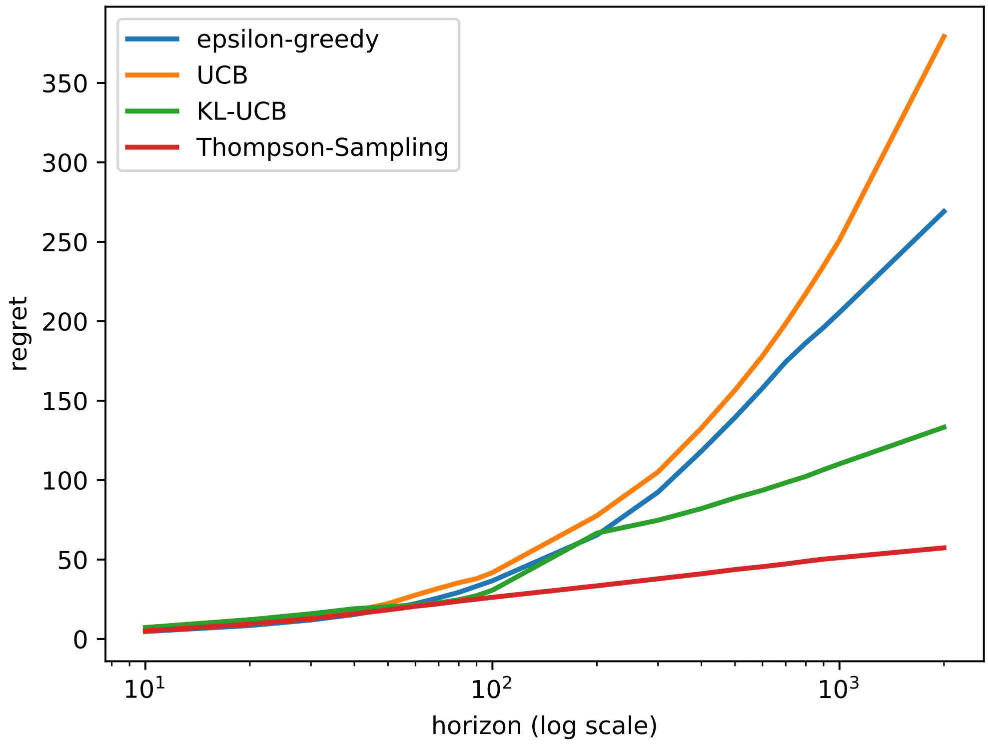
2.1 5 Arms


epsilon-greedy
UCB
KL-UCB
Thompson-Sampling
regret
0
250
500
750
1000
1250
1500
1750
2000
10^1
10^2
10^3
10^4
10^5
horizon (log scale)


epsilon-greedy
UCB
KL-UCB
Thompson-Sampling
regret
0
20
40
60
80
100
120
10^1
10^2
10^3
horizon (log scale)

2.2 25 Arms


epsilon-greedy
UCB
KL-UCB
Thompson-Sampling
5000
4000
3000
2000
1000
0
regret
10^1
10^2
10^3
10^4
10^5
horizon (log scale)


epsilon-greedy
UCB
KL-UCB
Thompson-Sampling
350
300
250
200
150
100
50
0
regret
10^1
10^2
10^3
horizon (log scale)