CS 747
Assignment 1 - Report

Goutham Ramakrishnan, 140020039

Note:
Epsilon Greedy Algorithm was simulted with a value of Epsilon=0.1.
All reported average values are the average of 100 runs.

Instance 1: 5 Arms

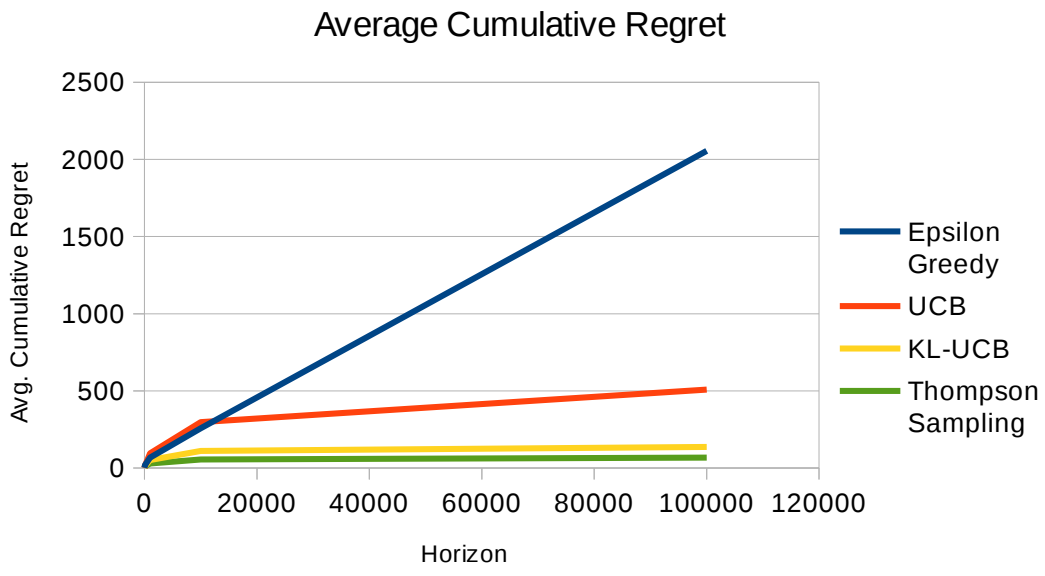
Arm values: 0.1, 0.2, 0.3, 0.4, 0.5

Average Cumulative Regret	Horizon				
Algorithm ↓	**10**	**100**	**1000**	**10000**	**100000**
Epsilon Greedy	2.91	19.74	67.42	257.77	2054.83
UCB	1.68	15.92	95.55	296.93	508.7
KL-UCB	1.23	11.69	52.38	110.74	136.01
Thompson Sampling	1.58	10.15	30.05	55.54	68.33

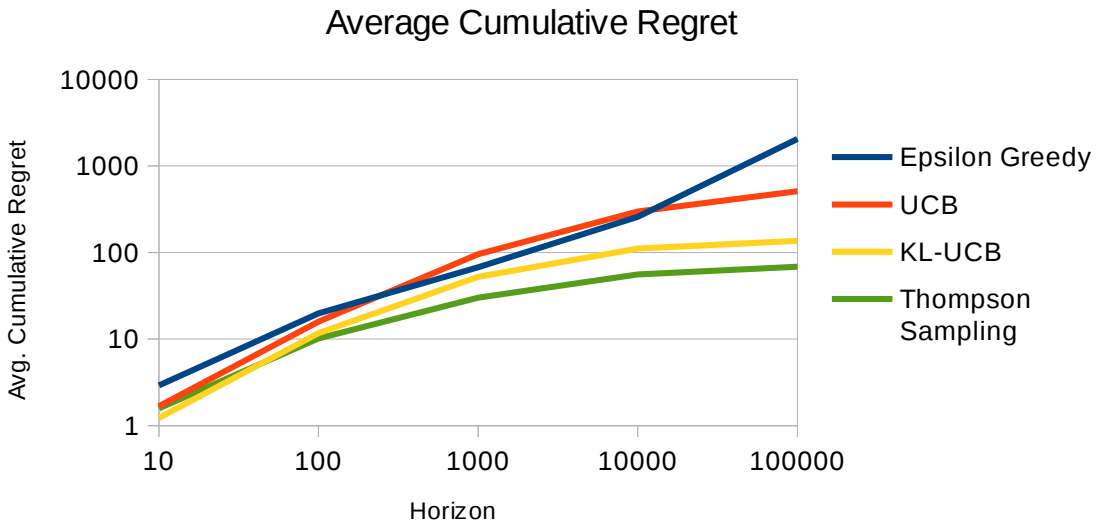
Average Regret per Pull	Horizon				
Algorithm ↓	**10**	**100**	**1000**	**10000**	**100000**
Epsilon Greedy	0.291	0.1974	0.06742	0.025777	0.0205483
UCB	0.168	0.1592	0.09555	0.029693	0.005087
KL-UCB	0.123	0.1169	0.05238	0.011074	0.0013601
Thompson Sampling	0.158	0.1015	0.03005	0.005554	0.0006833

(Std. Dev. of Regret)/T	Horizon				
Algorithm ↓	**10**	**100**	**1000**	**10000**	**100000**
Epsilon Greedy	0.1955	0.0974	0.0494	0.0084	0.0018
UCB	0.1588	0.0607	0.0191	0.0052	0.0016
KL-UCB	0.1984	0.0552	0.0200	0.0057	0.0016
Thompson Sampling	0.1695	0.0619	0.0195	0.0053	0.0015

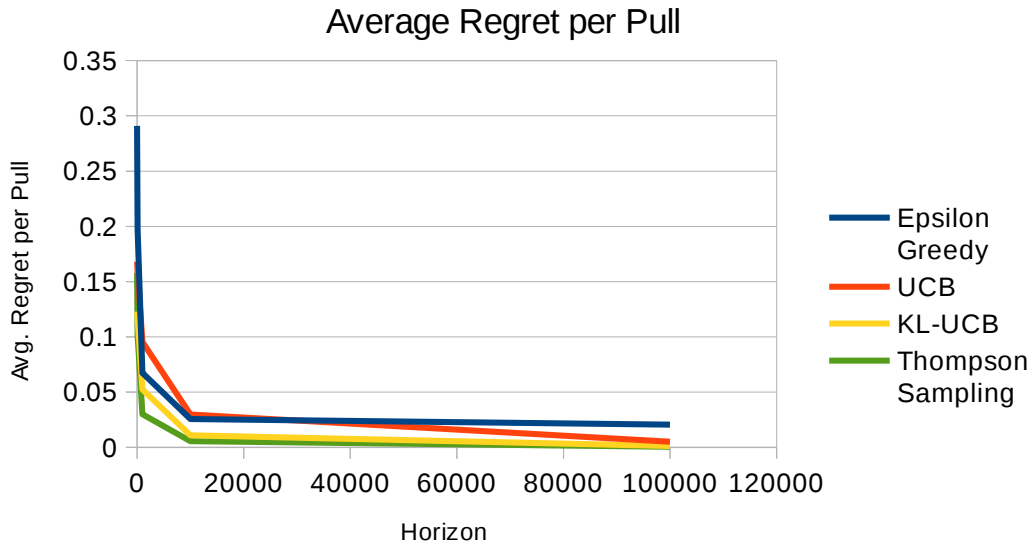
Average Cumulative Regret:
Linear Scale:



Log Scale:



Average Regret per Pull:



Instance 2: 25 Arms

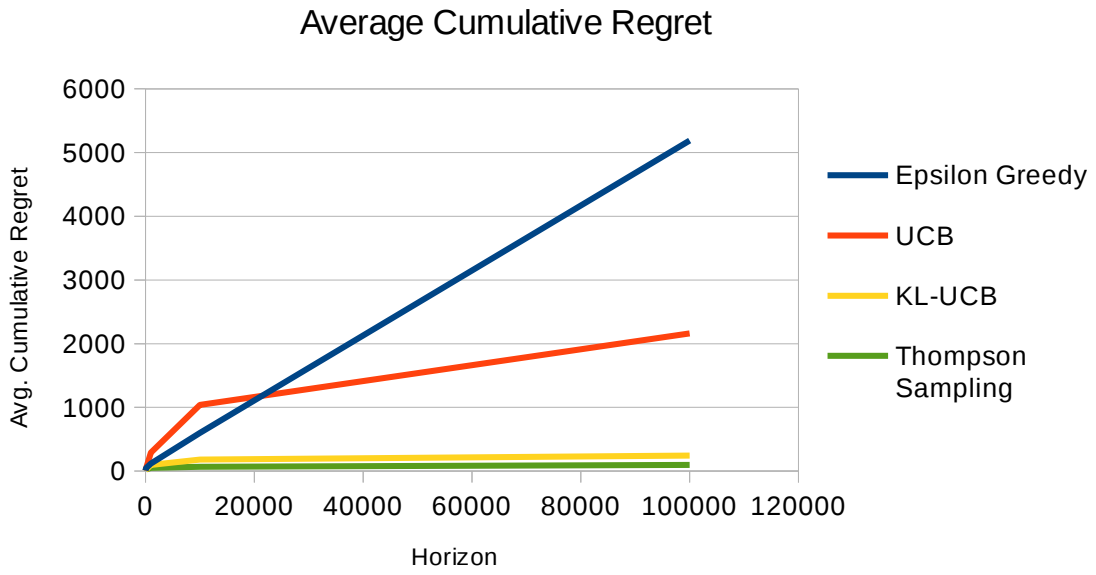
Arm values: 0.12, 0.14, 0.19, 0.21, 0.24, 0.24, 0.25, 0.27, 0.3, 0.31, 0.33, 0.36, 0.38, 0.4, 0.41, 0.46, 0.53, 0.56, 0.61, 0.71, 0.86, 0.87, 0.9, 0.92, 0.97

Average Cumulative Regret	Horizon				
Algorithm ↓	10	100	1000	10000	100000
Epsilon Greedy	7.62	44	115.09	598.23	5186.45
UCB	7.13	43.84	288.92	1037.5	2161.59
KL-UCB	7.14	21.61	95.46	178.6	242.81
Thompson Sampling	4.24	23.96	49.36	68.5	97.47

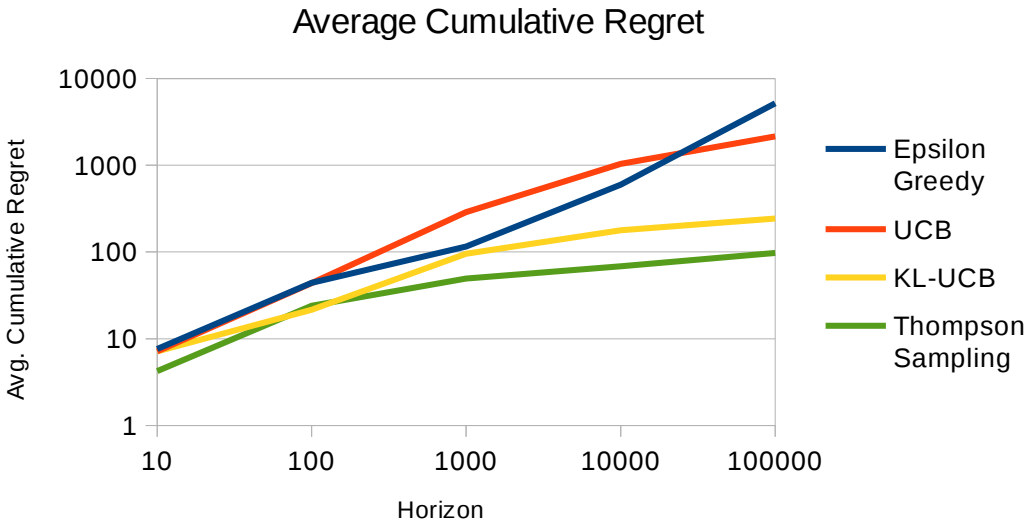
Average Regret per Pull	Horizon				
Algorithm ↓	10	100	1000	10000	100000
Epsilon Greedy	0.762	0.44	0.11509	0.059823	0.0518645
UCB	0.713	0.4384	0.28892	0.10375	0.0216159
KL-UCB	0.714	0.2161	0.09546	0.01786	0.0024281
Thompson Sampling	0.424	0.2396	0.04936	0.00685	0.0009747

(Std. Dev. of Regret)/T	Horizon				
Algorithm ↓	10	100	1000	10000	100000
Epsilon Greedy	0.1879	0.2567	0.0401	0.0097	0.0014
UCB	0.1519	0.0405	0.0109	0.0029	0.0007
KL-UCB	0.1459	0.0273	0.0136	0.0022	0.0006
Thompson Sampling	0.1834	0.0430	0.0146	0.0022	0.0005

Average Cumulative Regret:
Linear Scale:



Log Scale:



Average Regret per Pull:

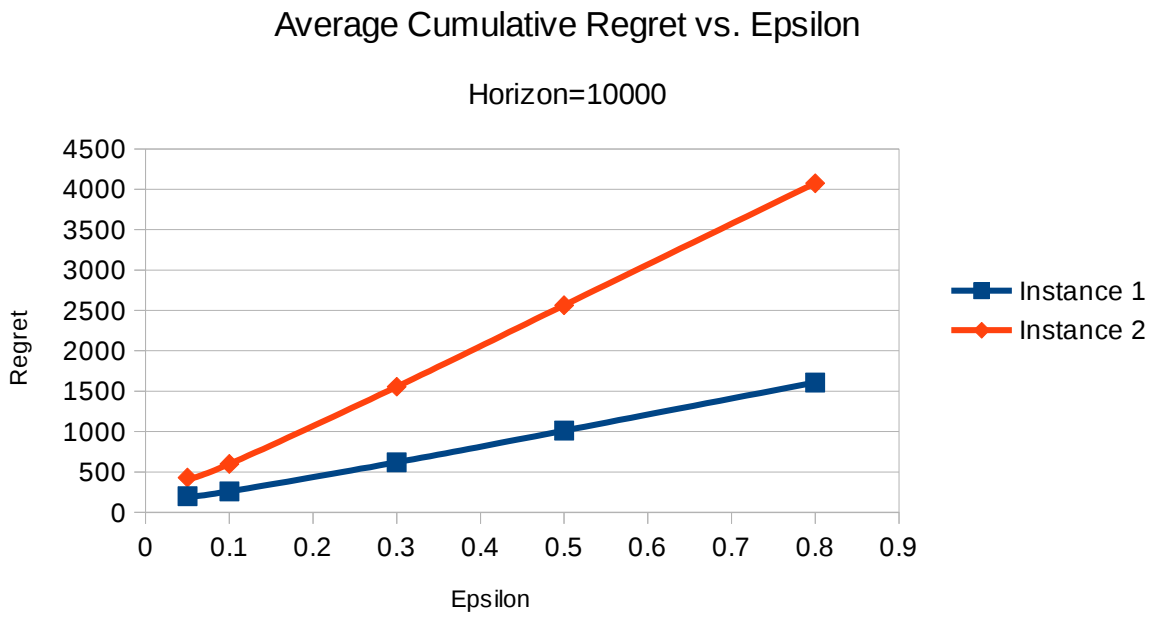
Observations:

- Saturation of Average Cumulative Regret:
  - → KL-UCB and Thompson Sampling result in early saturation of the regret.
  - → UCB, which is less optimal than the above two, also shows signs of saturation. But the value of regret is more than 10 times the corresponding value obatined for KL-UCB and Thompson Sampling.
  - → Thompson Sampling gives a cumulative regret ~ 50% of KL-UCB.
  - → As expected, the Cumulative Regret for Epsilon Greedy does not show signs of saturation. To improve its performace, we can use the Epsilon-t algorithm, to reduce exploratory behavior as time goes on.
- Average Regret per Pull:
  - → The graphs for the two instances look remarkably similar.
  - → Thompson Sampling consistently outperforms the rest.
  - → For both instances, there is an overlap between the curves for UCB and Epsilon Greedy.
- KL-UCB and UCB performances for Instance-2 with Horizon=10 underestimate the performance.
  - → This is because in the implementation all the arms are sampled in ascending order first before the algorithm starts.
  - → Since there are 25 Arms, the arms with the lowest values of true probability are sampled first. Therefore, the two algorithms perform poorly and almost identically.
  - → To tackle this issue, instead of sampling the arms in a specific order, we can randomize the round-robin arm pulling to sample each arm once.
- Robustness of Thompson Sampling:
  - → Thompson sampling consistently performs the best for both instances across horizons.
  - → The algorithm learns extremely fast.
  - → KL-UCB gives similar performance upto Horizon=100, but after that Thompson Sampling becomes significantly better.
- Choice of Log Base:
  - → Log2 was chosen for the implementation of UCB and KL-UCB.
  - → Perfomance marginally improved when Natutal Log was used. But the performance trends observed remained the same.
- Analysis of Standard Deviations:
  - → As expected, the standard deviation of the regret divided by horizon decreases for all four algorithms as the length of horizon increases.
  - → The value saturates to nearly identical values for UCB, KL-UCB and Thompson Sampling.

- Value of Epsilon:
  - → The value of epsilon chosen represents the tradeoff between exploitative and exploratory behavior in the epsilon greedy algorithm.
  - → Simulation was done for five values of epsilon: 0.05, 0.1, 0.3, 0.5, 0.8.

Average Cumulative Regret for Horizon = 10000.

Epsilon	Instance 1	Instance 2
0.05	199.83	431.16
0.1	259.97	598.66
0.3	620.01	1555.61
0.5	1011.65	2564.49
0.8	1607.86	4075.09



Remarkably, the cumulative regret appears to increase linearly with increase in epsilon (increase in exploratory behavior).
The small value of epsilon (0.05) appears to perform well for this horizon. For smaller horizons, a larger value of epsilon may be needed for good performance.
Intuitively, the slope of the line for Instance 2 is muchlarger because there is a lot more to lose by pulling non-optimal arms in this case. The greater the spread of the arm probability values, the greater the slope of this line.

References:

- → C++ Documentation
- → StackOverFlow
- → https://stackoverflow.com/questions/15165202/random-number-generator-with-beta-distribution
- → Boost Library was used to implement the beta distribution for Thompson Sampling.
- → Simulation was done using attached Bash Script, changing the value of Horizon each time.
- → Average value calculations and graph plotting was done in ODS Spreadsheet. Attached for reference.