CS 747 : Assignment 1
Report
Roll No. 15D170013

Notes and answers to question :

1) Observations about the results:

In general we see the trend that the

Regret for epsilon_greedy > UCB > KLUCB > Thompson Sampling

In the epsilon greedy methods : increase in epsilon -> increase in regret

2) Patterns Displayed:

The graphs are increasing as expected and the ordering on an average as mentioned above. The 10,000 ones show varied trend after time of around 4000.

3) Effect of epsilon and intuition:

Increasing epsilong increases the regret. This because exploration is higher and for greater values of Horizon (1000, 10000) exploitation seems like a better way forward.

4) Unexpected Results:

As can be seen from the graphs increasing the epsilon results in increase in the regret. The Thomson sampling as read oline has lower regrets practically but in my case the KL UCB algorithm performs better in some instances.
Sometimes for smaller regrets and horizons the epsilon-greedy method performs better than the remaining but on an average for different horizons. Also in the histogram plots with a horizon of 10,000 we see that Thompson's regret is higher for time < 4000 but reverses and beats KLUCB and epsilon greedy for higher regrets.

5) Further improve performance?

Optimizations : For the epsilon we should have some way in the exploitation to sample certain random arms initially. Also in the client side if we could have implemented horizon dependent epsilons for better exploitation vs exploration. Also parallelising the code to run on different ports would result in faster improvements.

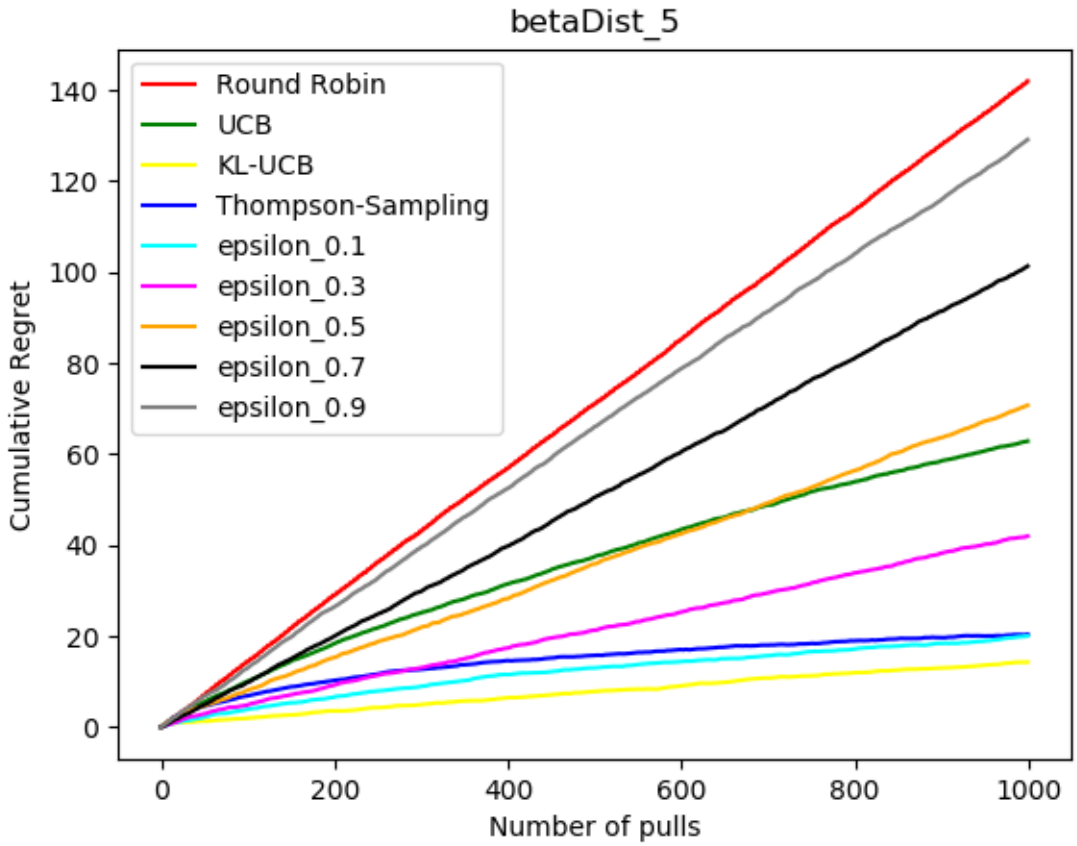
6) Changes for Non Bernoulli reward :

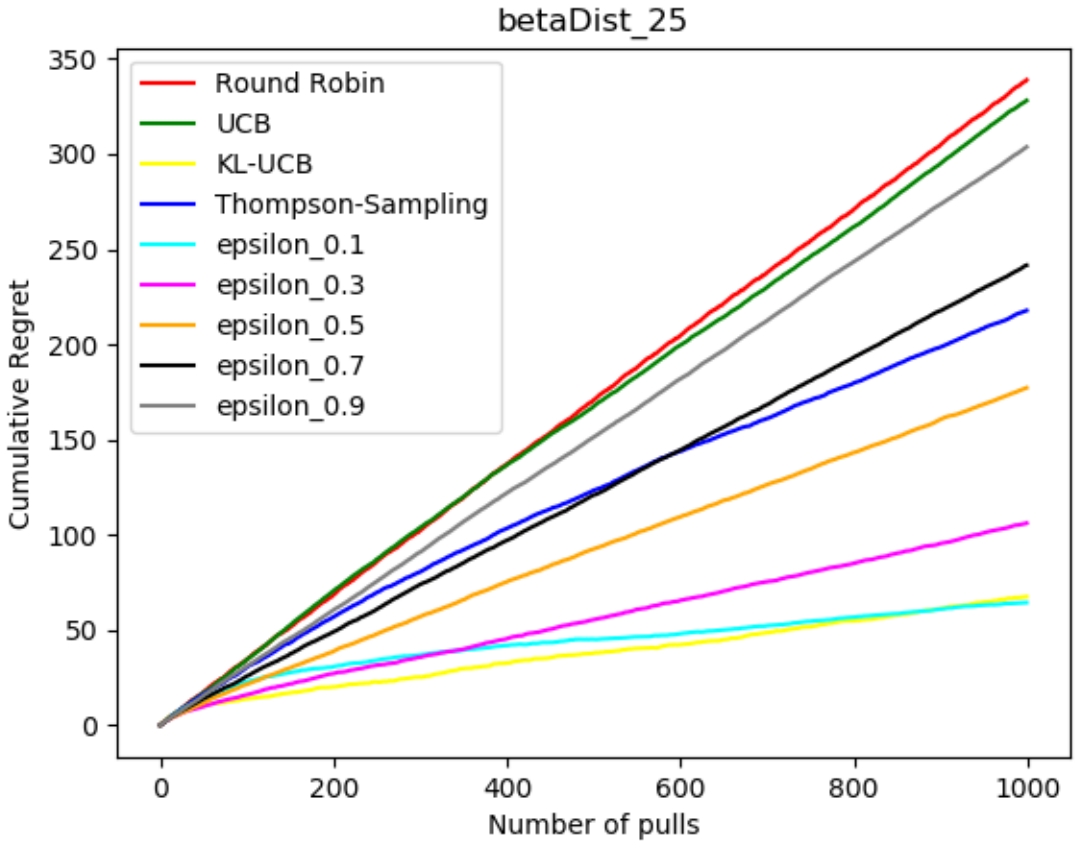
Thomsons Sampling : Handling the cases of non Bernoulli rewards is done in the following way. We sample from a Bernoulli distribution keeping the reward received as the parameter. If the sample is 1 then we consider it as a success and a failure vice versa.
For the other algorithms such cases are inherently handled.

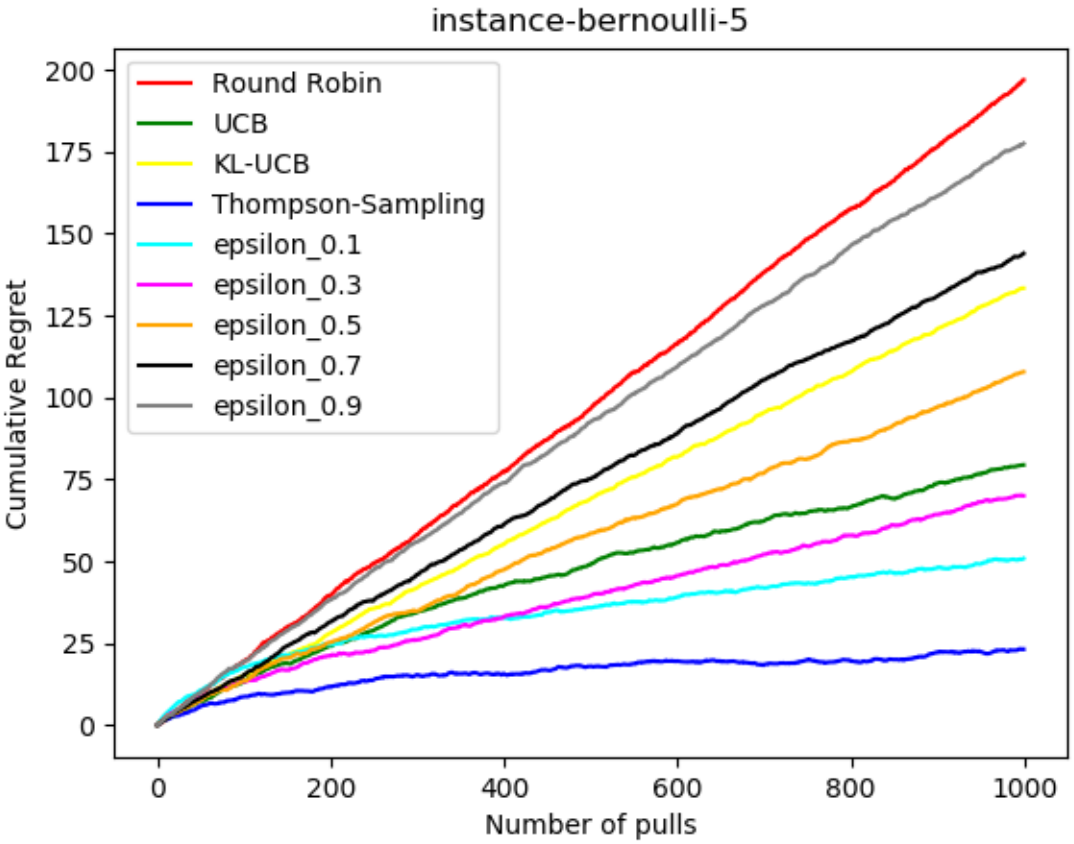
7) Implementation Details Specific:

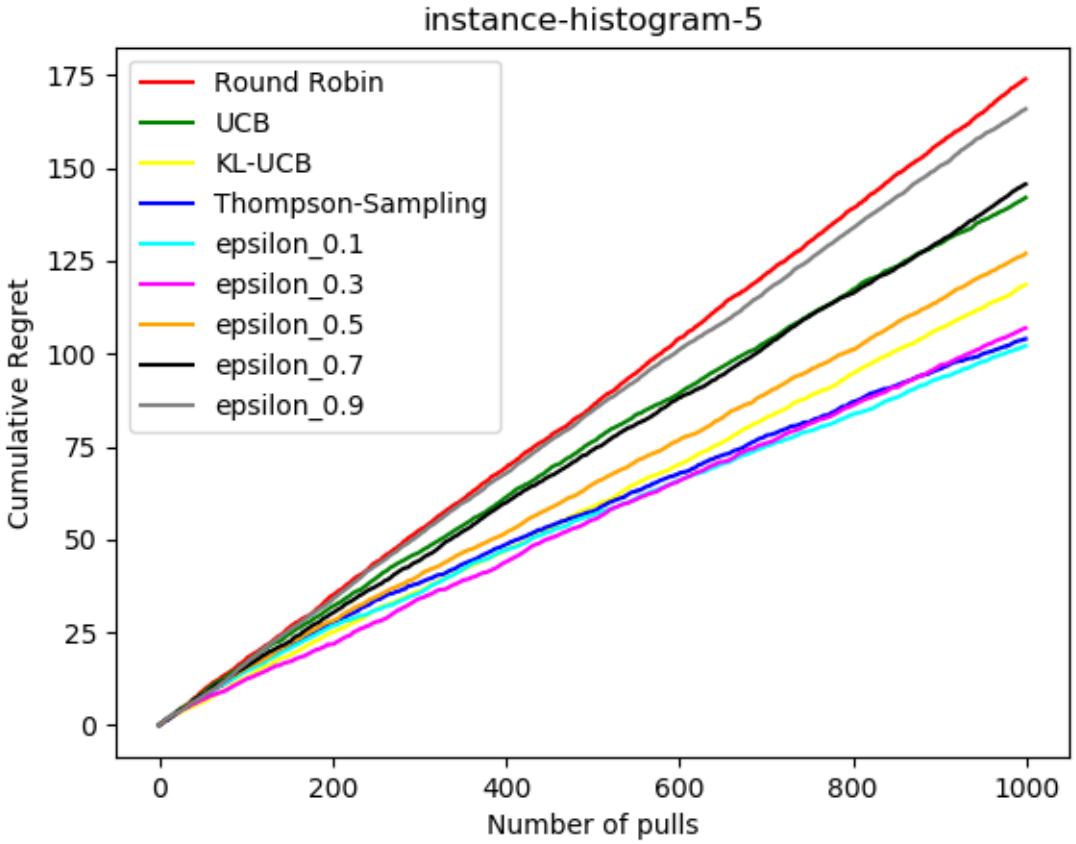
KLUCB : For running the algo the step of finding q is done using the Binary Search between p_hat and 1 and not using Newton iteration method as suggested in the paper. This is done up to an accuracy of 10^-6. i.e. 20 loops.

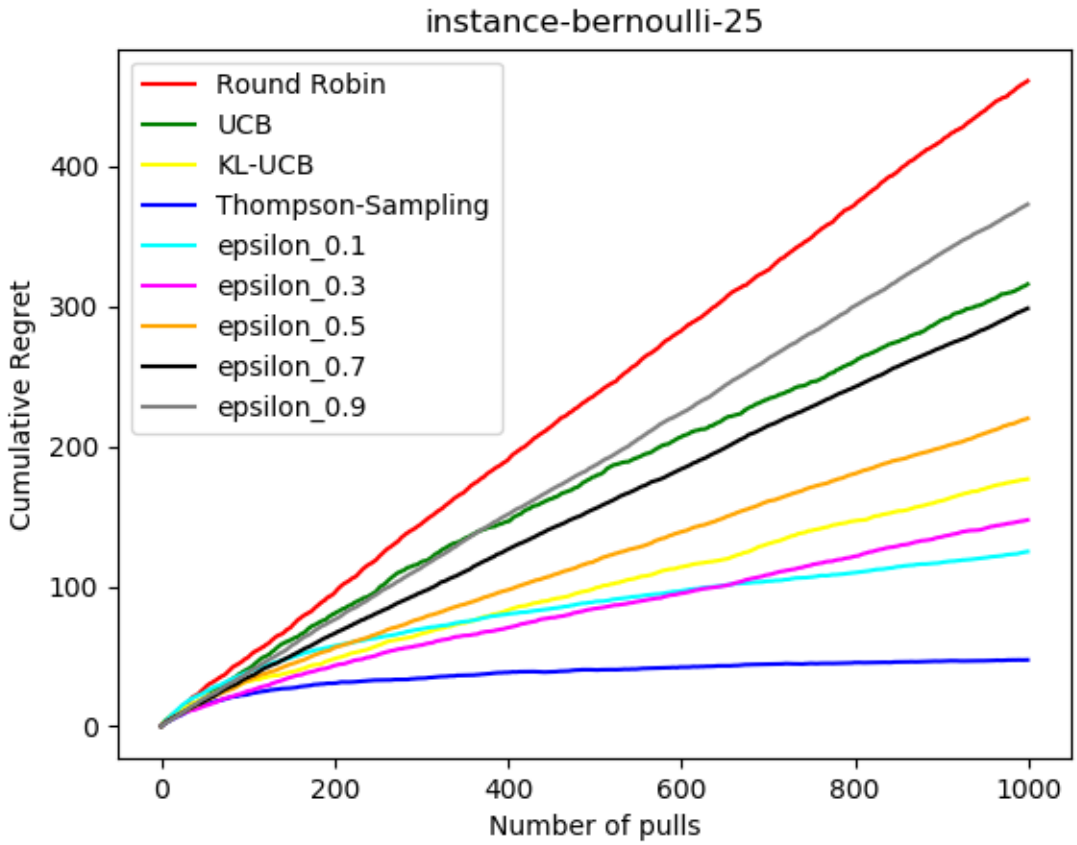
Following are the graphs for horizon set to 1000:

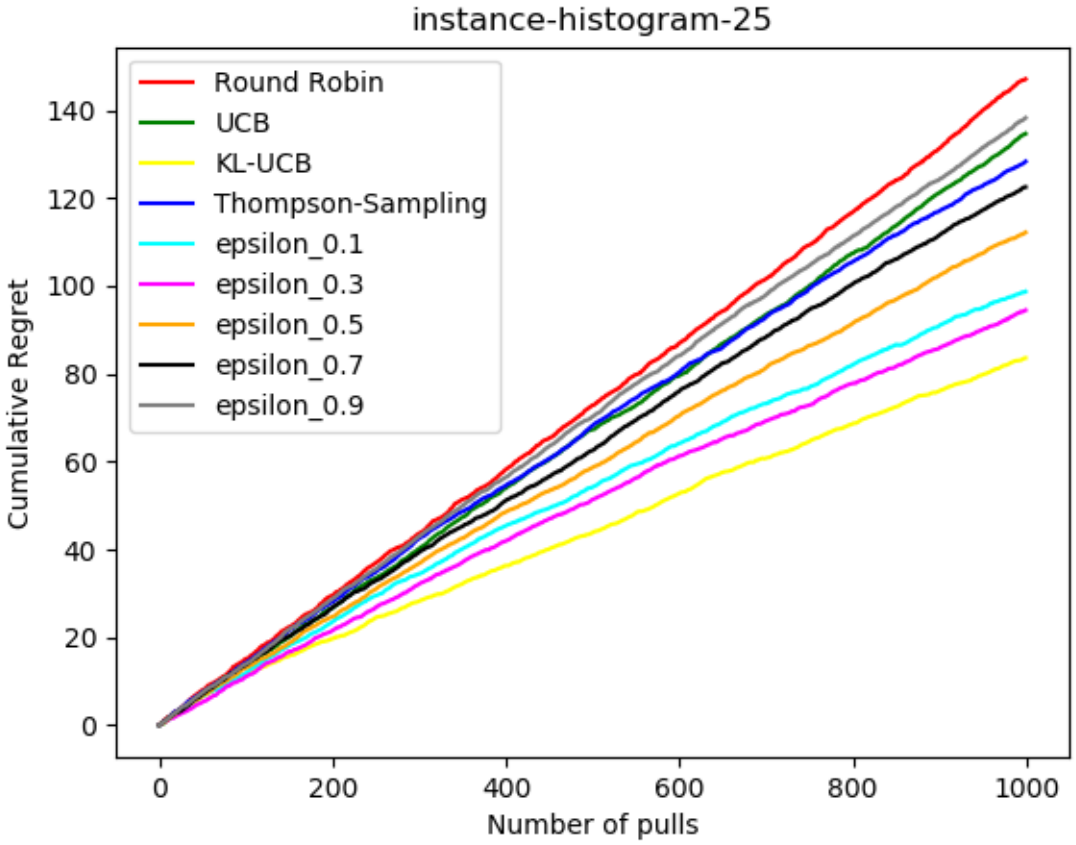




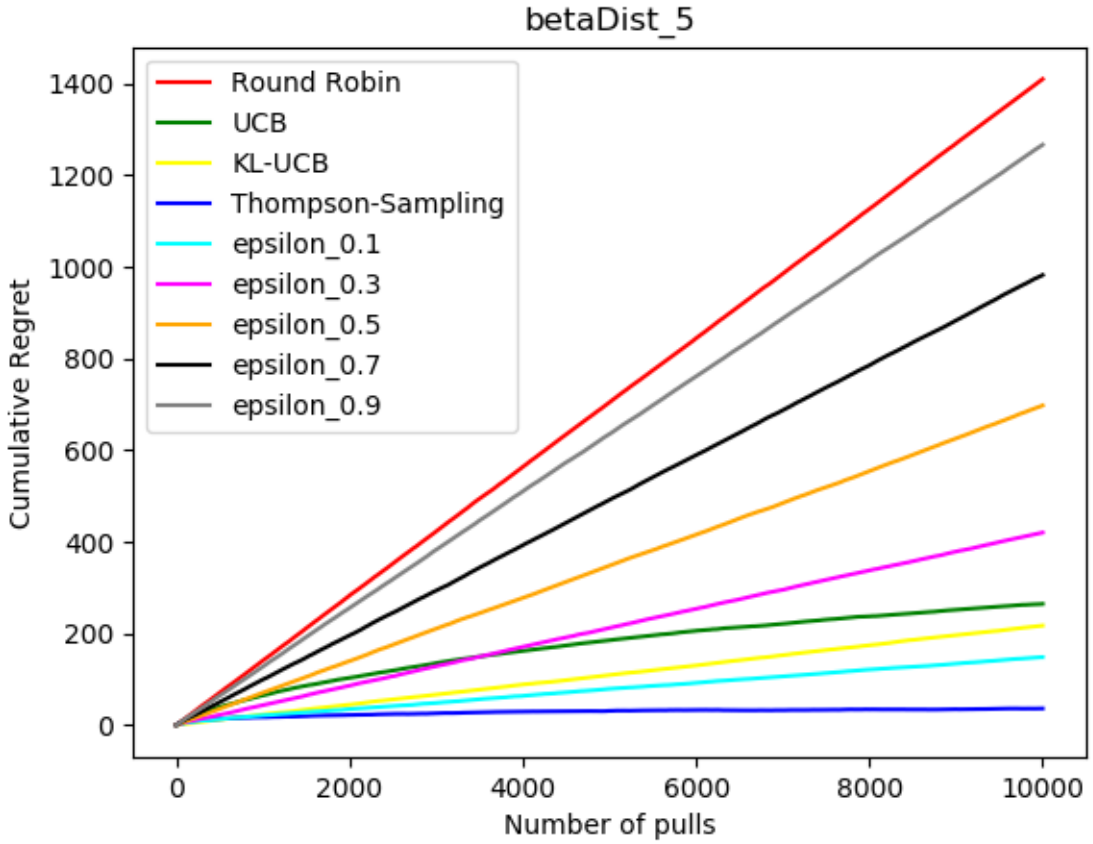

instance-bernoulli-5
Round Robin
UCB
KL-UCB
Thompson-Sampling
epsilon_0.1
epsilon_0.3
epsilon_0.5
epsilon_0.7
epsilon_0.9
Cumulative Regret
Number of pulls
0
25
50
75
100
125
150
175
200
0
200
400
600
800
1000

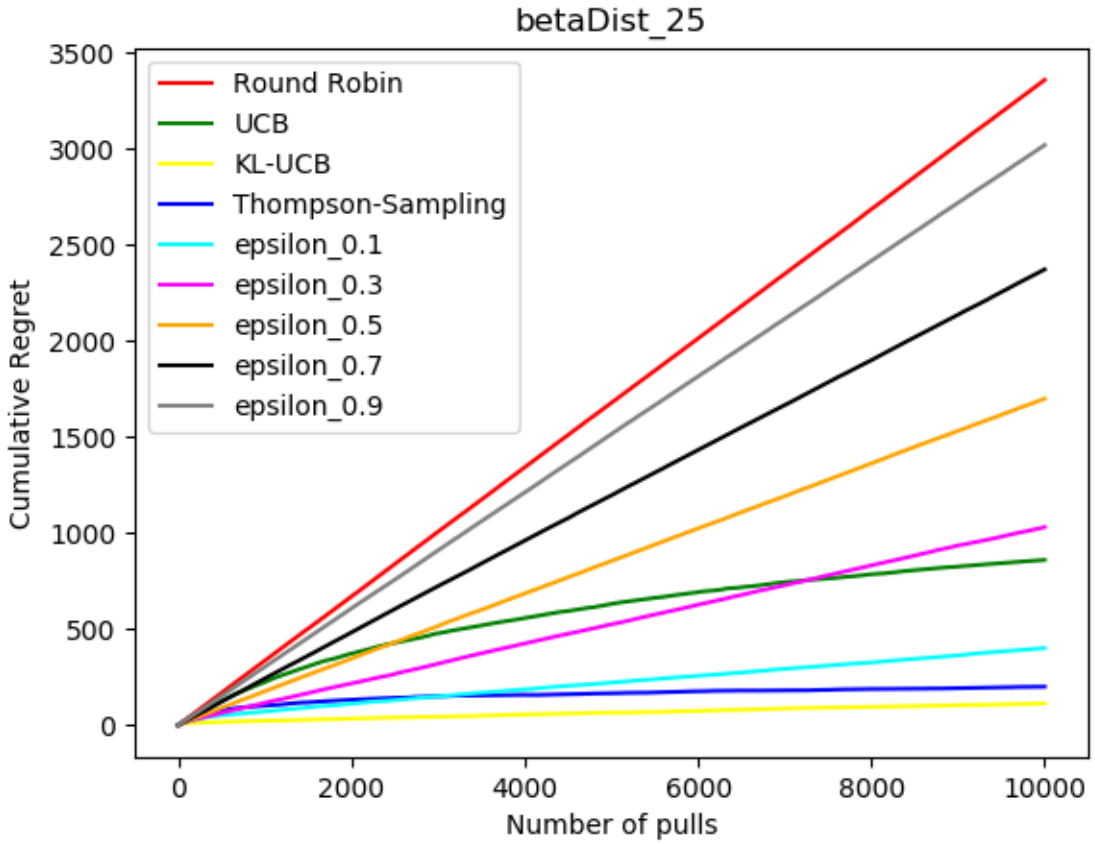

instance-histogram-5
Round Robin
UCB
KL-UCB
Thompson-Sampling
epsilon_0.1
epsilon_0.3
epsilon_0.5
epsilon_0.7
epsilon_0.9
Cumulative Regret
Number of pulls
0
25
50
75
100
125
150
175
0
200
400
600
800
1000

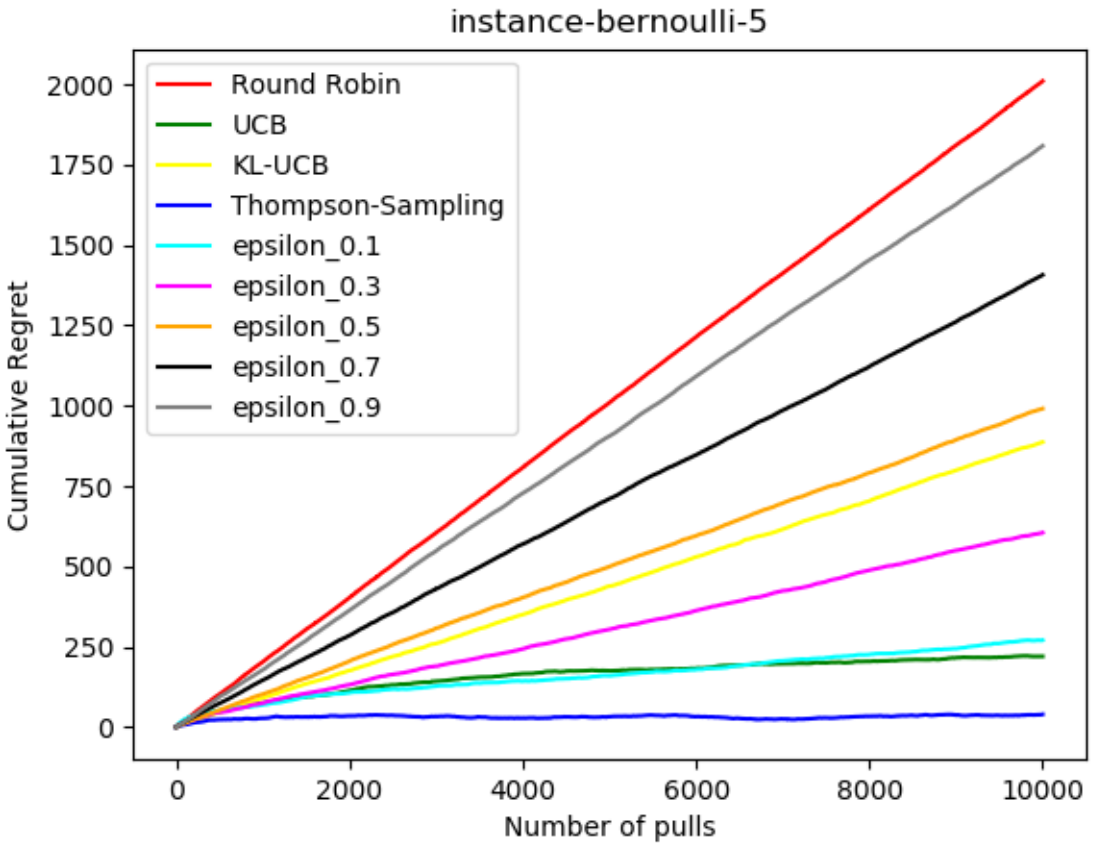

instance-bernoulli-25
Round Robin
UCB
KL-UCB
Thompson-Sampling
epsilon_0.1
epsilon_0.3
epsilon_0.5
epsilon_0.7
epsilon_0.9
Cumulative Regret
0
100
200
300
400
0
200
400
600
800
1000
Number of pulls

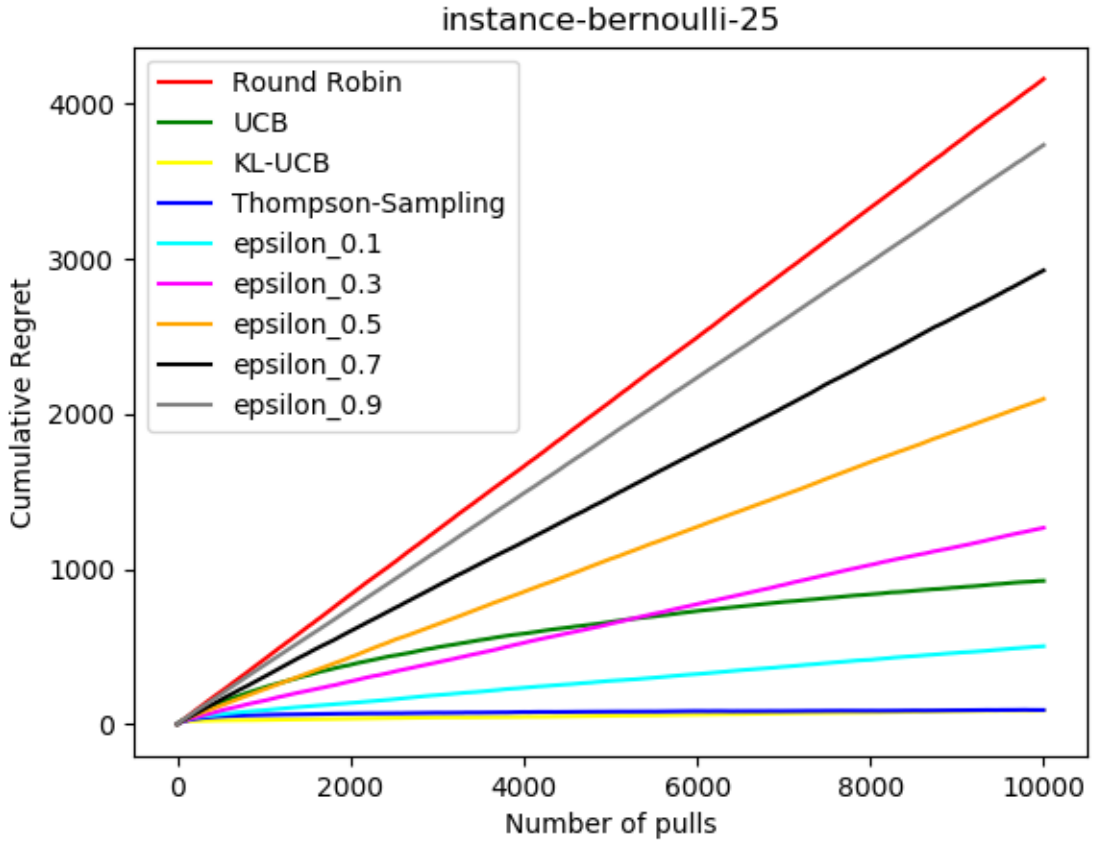

instance-histogram-25
Round Robin
UCB
KL-UCB
Thompson-Sampling
epsilon_0.1
epsilon_0.3
epsilon_0.5
epsilon_0.7
epsilon_0.9
Cumulative Regret
0
20
40
60
80
100
120
140
0
200
400
600
800
1000
Number of pulls

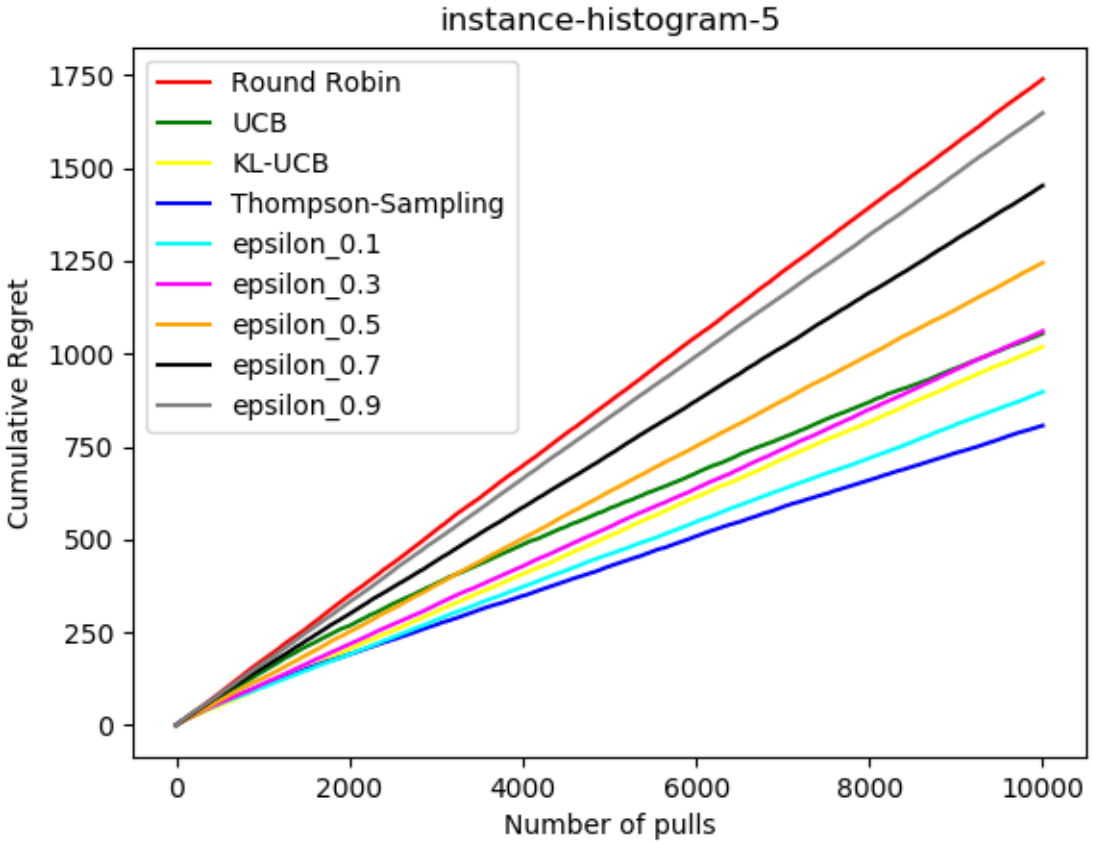
Following are the graphs for horizon set to 10000:

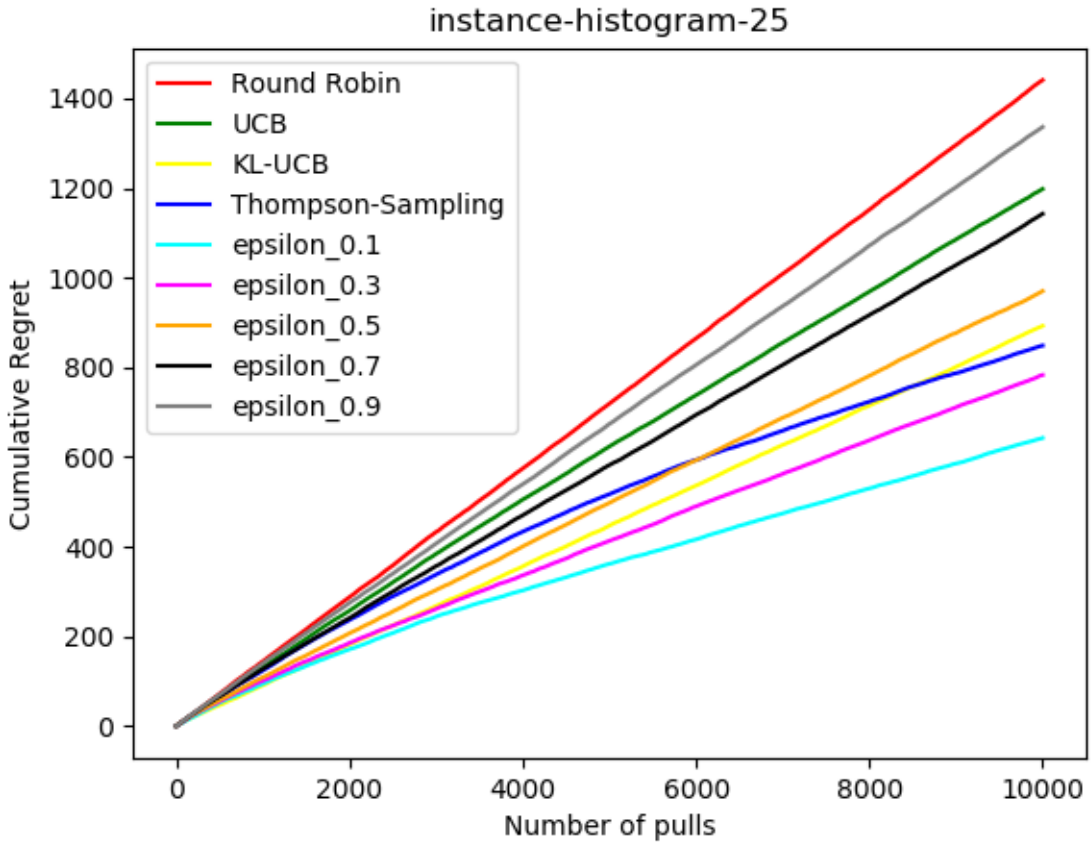





instance-bernoulli-5
Round Robin
UCB
KL-UCB
Thompson-Sampling
epsilon_0.1
epsilon_0.3
epsilon_0.5
epsilon_0.7
epsilon_0.9
Cumulative Regret
Number of pulls
0
250
500
750
1000
1250
1500
1750
2000
0
2000
4000
6000
8000
10000


instance-bernoulli-25
Round Robin
UCB
KL-UCB
Thompson-Sampling
epsilon_0.1
epsilon_0.3
epsilon_0.5
epsilon_0.7
epsilon_0.9
Cumulative Regret
Number of pulls
0
1000
2000
3000
4000
0
2000
4000
6000
8000
10000


instance-histogram-5
Round Robin
UCB
KL-UCB
Thompson-Sampling
epsilon_0.1
epsilon_0.3
epsilon_0.5
epsilon_0.7
epsilon_0.9
Cumulative Regret
Number of pulls
0
250
500
750
1000
1250
1500
1750
0
2000
4000
6000
8000
10000


instance-histogram-25
Round Robin
UCB
KL-UCB
Thompson-Sampling
epsilon_0.1
epsilon_0.3
epsilon_0.5
epsilon_0.7
epsilon_0.9
Cumulative Regret
Number of pulls
0
200
400
600
800
1000
1200
1400
0
2000
4000
6000
8000
10000