Reinforcement Learning

Charu Agarwal (160010038)
Varshha Thandu (160010027)
Manoj Kumar (160010019)
Ishan Srivastava (160010013)
Rahul Kulkarni (193061002)

Analysis of algorithms for the multi-armed bandit problem

3rd September 2019

OVERVIEW

The multi-armed bandit problem is a popular reinforcement learning problem to study the tradeoff between exploration and exploitation. The problem is described as follows: An agent has to repeatedly choose between n arms or options. Each arm yields a reward which follows a stationary probability distribution. The objective is to maximize the expected total rewards over a time period. In this study, we empirically analyze the popular algorithms for the multi-armed bandit problem including both value-based and policy-based approaches.

ALGORITHMS

Value-Based Approaches

1. Epsilon-Greedy

   At each time step estimates for each action are given by:

   $$Q_t(a) = (\sum_{i=1}^{t-1} R_i \cdot 1_{A_i=a}) / (\sum_{i=1}^{t-1} 1_{A_i=a})$$

   $Q_t(a)$ is initially set to default value as 0

   As denominator in $Q_t(a)$ tends to ∞, $Q_t(a) \rightarrow q_*(a)$ [true value of action]

   With epsilon probability, we will choose a random action (exploration) and choose an action with maximum $Q_t(a)$ with probability 1-**ε**.

   With probability **ε** – we randomly choose an action from a set of all actions A, independent of action value estimates.

   With probability 1- **ε** – we choose action with the maximum value ($A_t = argmax_a \ Q_t(a)$)

2. Softmax rule:

It does both exploration and exploitation. The temperature parameter is for greedy exploration. As the temperature is reduced, the highest-valued arms are more likely to be chosen and, in the limit as τ -->0, the best arm is always chosen

Let $\hat{\mu}_1$, $\hat{\mu}_2$ ..... $\hat{\mu}_k$ be estimates of arms

Initially are choosen as $\hat{\mu}_1[0]$, $\hat{\mu}_2$ ..... $\hat{\mu}_k$

P(Arm 1) = e^($\hat{\mu}_1$ /τ)/($\sum_{i=1}^{n}$ e^($\hat{\mu}_i$ /τ))

& similarly for all K-arms and all τ temperatures.

3. Upper Confidence Bound (UCB)

The Upper Confidence Bounds (UCB) algorithm measures this potential by upper confidence bound of the reward value so that the true value is below the bound with high probability. The upper bound is a function of N_t(i); a larger number of trials N_t(i) should give us a smaller bound.

In UCB algorithm, we always select the greediest action to maximize the upper confidence bound:

I(t) = argmax(R(i) +sqrt(2*log(t)/N(i)))

4. Thompson Sampling

The basic idea of Thompson sampling is that in each round, we take our existing knowledge of the arms, which is in the form of a posterior belief about the unknown parameters, and we "sample" the parameters from this posterior distribution. This sampled parameter yields a set of expected rewards for each arm, and now we bet on the one with the highest expected return, under that sampled parameter.

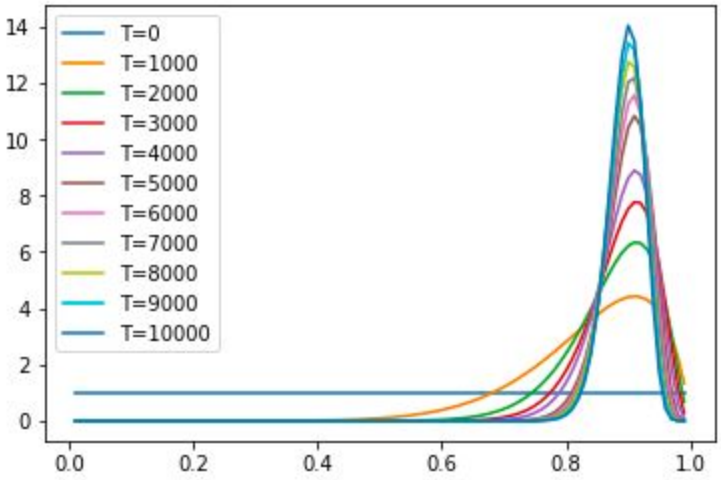


Figure: Convergence of beta distribution to true Bernoulli parameter p = 0.9 using Thompson Sampling

Policy-Based Approaches

1. Reinforcement Comparison

   The probability of each arm is calculated before each turn using Boltzmann distribution preference of arms. Preference of arm getting a reward is updated in each turn, with respect to empirical mean and expected reward. The new expected reward is updated as a function of earlier expected reward and empirical mean.

TESTBED

We considered Bernoulli reward distributions for the arms and three different settings for the number of arms (K) = 2, 5 and 10.

Following are the test bed parameters chosen:

T = 10,000 steps, K = 2 arms, P = [0.9, 0.6]

T = 10,000 steps, K = 2 arms, P = [0.45, 0.55]

T = 10,000 steps, K = 5 arms, P = [0.1, 0.4, 0.6, 0.3, 0.7]

T = 10,000 steps, K = 10 arms, P = [0.1, 0.2, 0.3, 0.4, 0.003, 0.6, 0.25, 0.8, 0.55, 0.7]

EVALUATION METRICS

We used the following performance criteria to compare and rank the algorithms:

1. Total regret accumulated over the experiment.
2. Regret as a function of time.

3. Percentage of plays in which the optimal arm is pulled.

RESULTS

These were the results obtained:

Algorithm	**Total Regret**
Thompson Sampling	314.14
UCB	359.23
Reinforce Comparison (with baseline)	377.51
Reinforce Comparison (no baseline)	385.3
Softmax (T=0.1)	3497.77
Epsilon-Greedy (Variable,epsilon=1/t)	5715.49
Epsilon-Greedy (Fixed, epsilon=0.1)	8193.77
Softmax (T = 1)	8534.95
Softmax (T = 10)	8952.35

Table 1: The total regret incurred by each algorithm for K=10 arms.

Algorithm	**Total Regret**
Thompson Sampling	193.55
UCB	201.45
Reinforce Comparison (with baseline)	416.9
Reinforce Comparison (no baseline)	468.85
Epsilon-Greedy (Variable,epsilon=1/t)	963.45
Softmax (T = 0.1)	2713.6
Softmax (T = 1)	7431.2
Epsilon-Greedy Fixed,epsilon=0.1)	7547.1
Softmax (T = 10)	7933.7

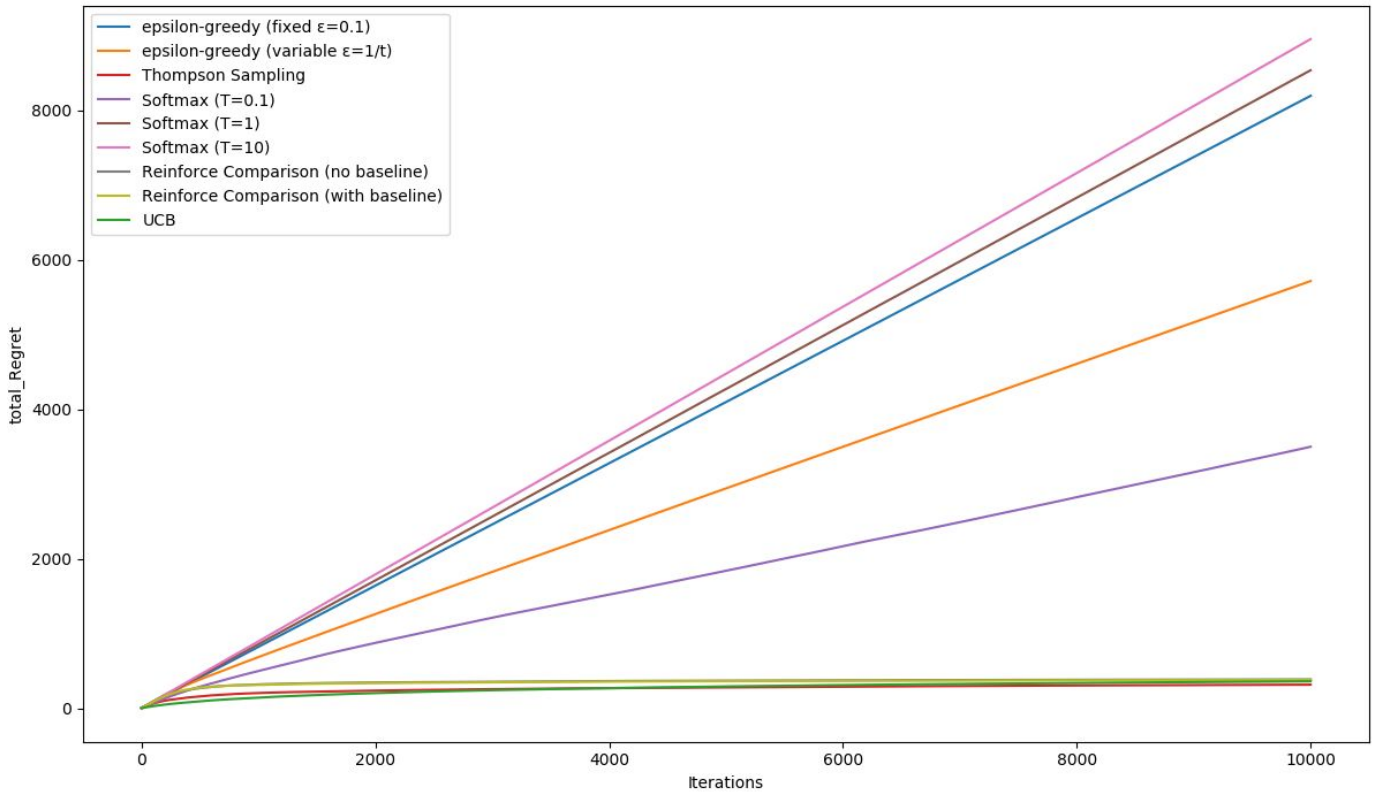
Table 2: The total regret incurred by each algorithm for K=5 arms.

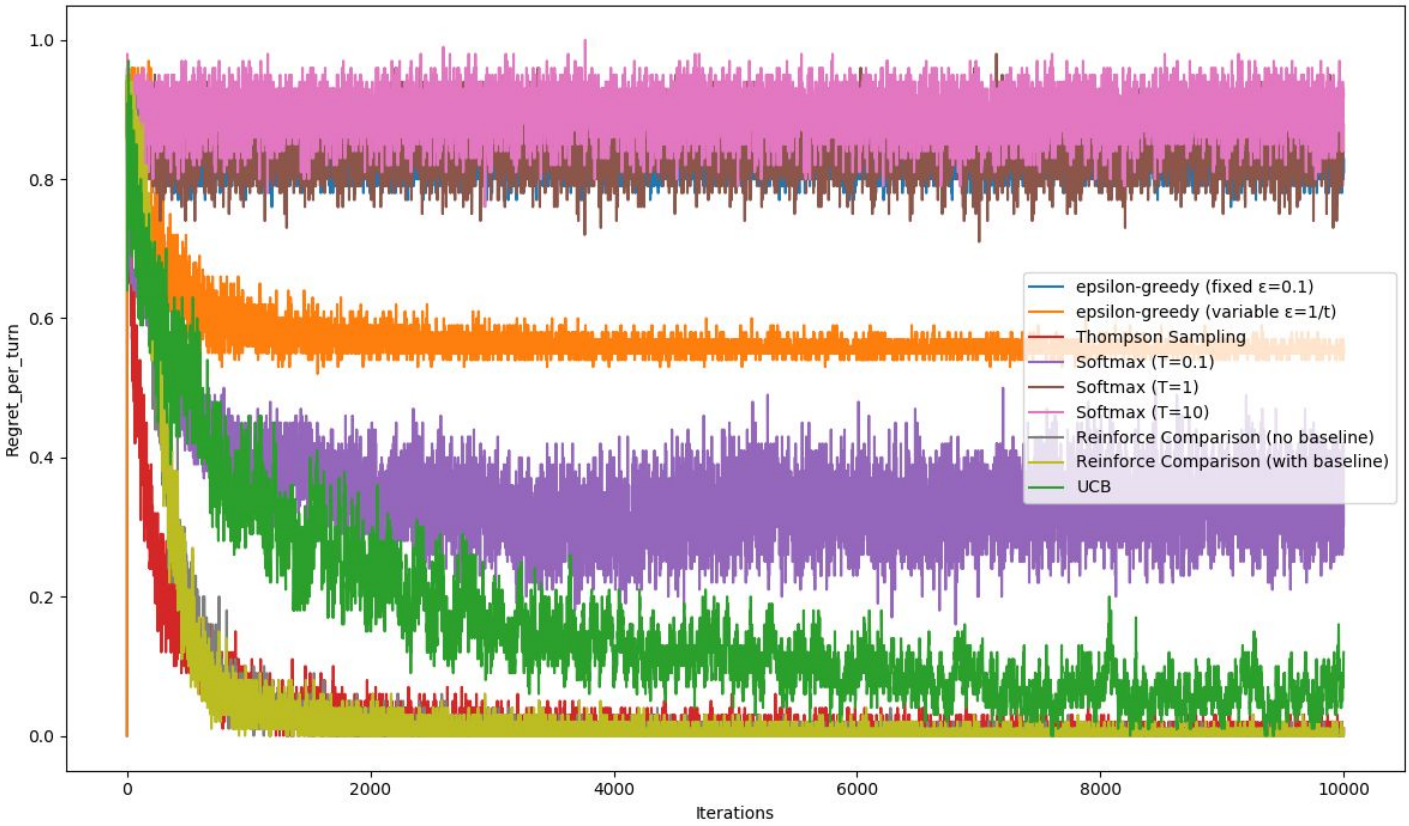
Algorithm	**Total Regret**
Thompson Sampling	18.1
UCB	48.65
Reinforce Comparison (with baseline)	118.4
Reinforce Comparison (no baseline)	127.15
Epsilon-Greedy (Variable,epsilon=1/t)	307.35
Softmax (T = 0.1)	535.4
Epsilon-Greedy (Fixed,epsilon=0.1)	952.75
Softmax (T = 1)	4229.7
Softmax (T = 10)	4939.2

Table 3: The total regret incurred by each algorithm for K=2 arms with P=[0.9, 0.6].

Algorithm	**Total Regret**
UCB	83.2
Thompson Sampling	198.0
Reinforce Comparison (with baseline)	338.2
Reinforce Comparison (no baseline)	342.05
Epsilon-Greedy (Variable,epsilon=1/t)	1243.9
Softmax (T = 0.1)	2514.15
Softmax (T = 1)	4767.75
Softmax (T = 10)	4959.0
Epsilon-Greedy (Fixed,epsilon=0.1)	7693.85

Table 4: The total regret incurred by each algorithm for K=2 arms with P=[0.45, 0.55].


epsilon-greedy (fixed ε=0.1)
epsilon-greedy (variable ε=1/t)
Thompson Sampling
Softmax (T=0.1)
Softmax (T=1)
Softmax (T=10)
Reinforce Comparison (no baseline)
Reinforce Comparison (with baseline)
UCB
total_Regret
Iterations
0
2000
4000
6000
8000
10000


epsilon-greedy (fixed ε=0.1)
epsilon-greedy (variable ε=1/t)
Thompson Sampling
Softmax (T=0.1)
Softmax (T=1)
Softmax (T=10)
Reinforce Comparison (no baseline)
Reinforce Comparison (with baseline)
UCB
Regret_per_turn
Iterations
0.0
0.2
0.4
0.6
0.8
1.0
0
2000
4000
6000
8000
10000

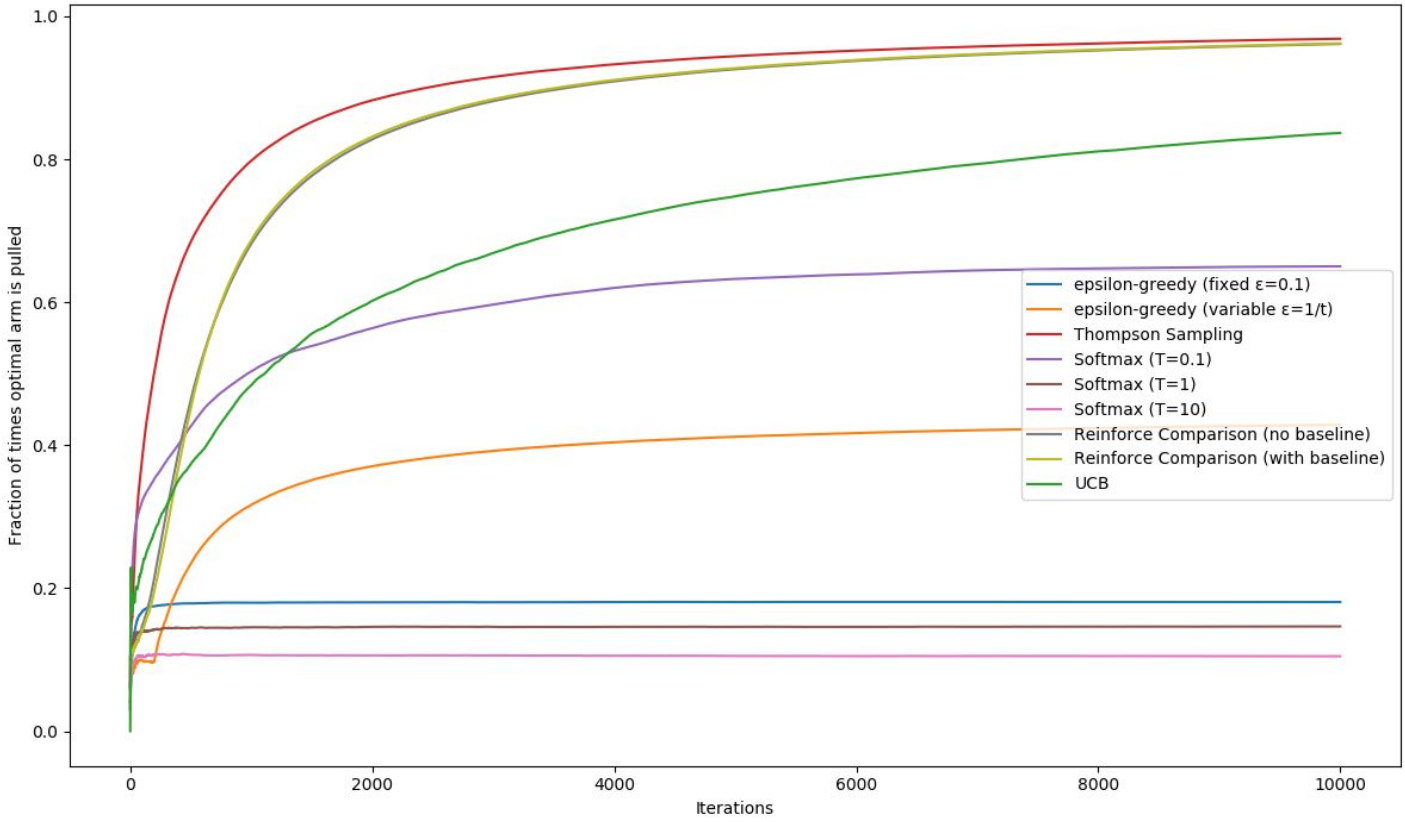
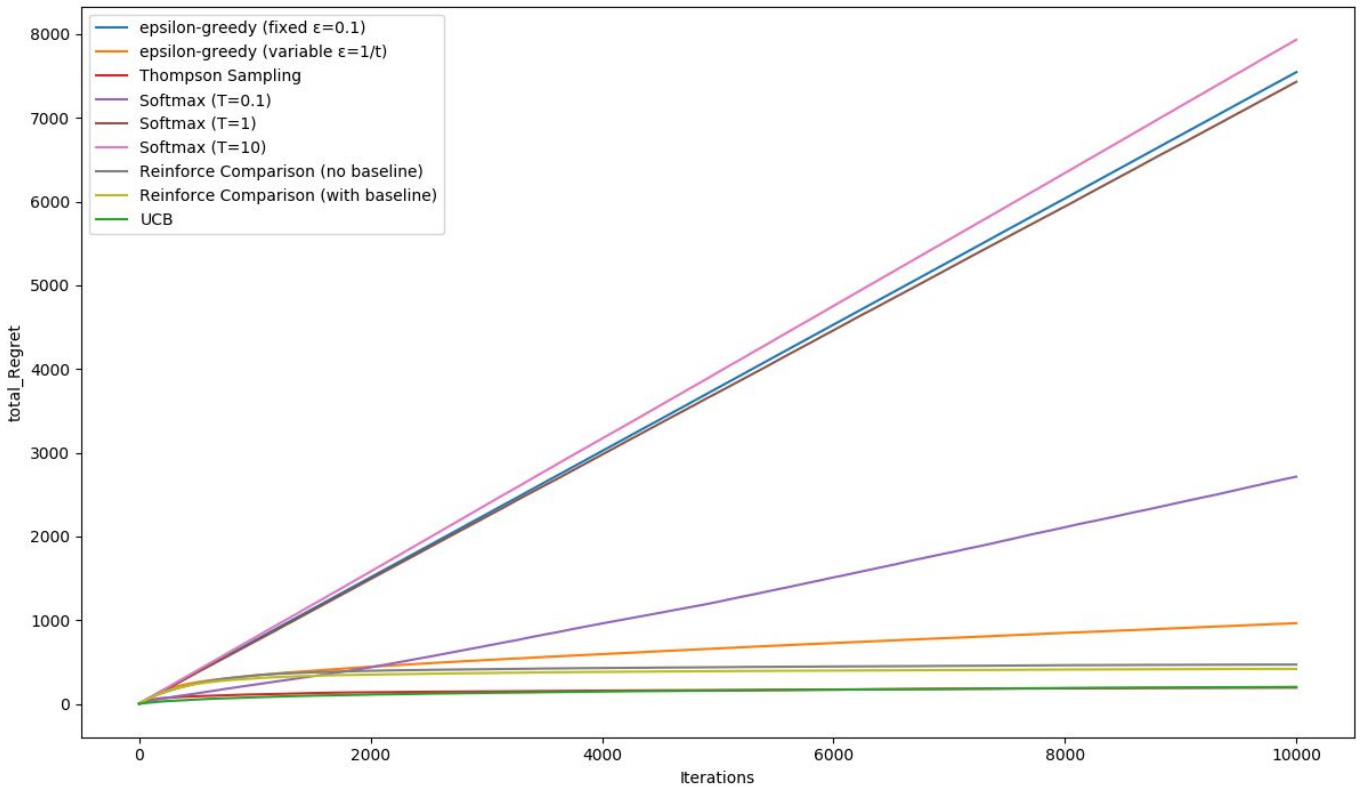


Figure 1: Empirical Results for 10 arms with P = [0.1, 0.2, 0.3, 0.4, 0.003, 0.6, 0.25, 0.8, 0.55, 0.7]



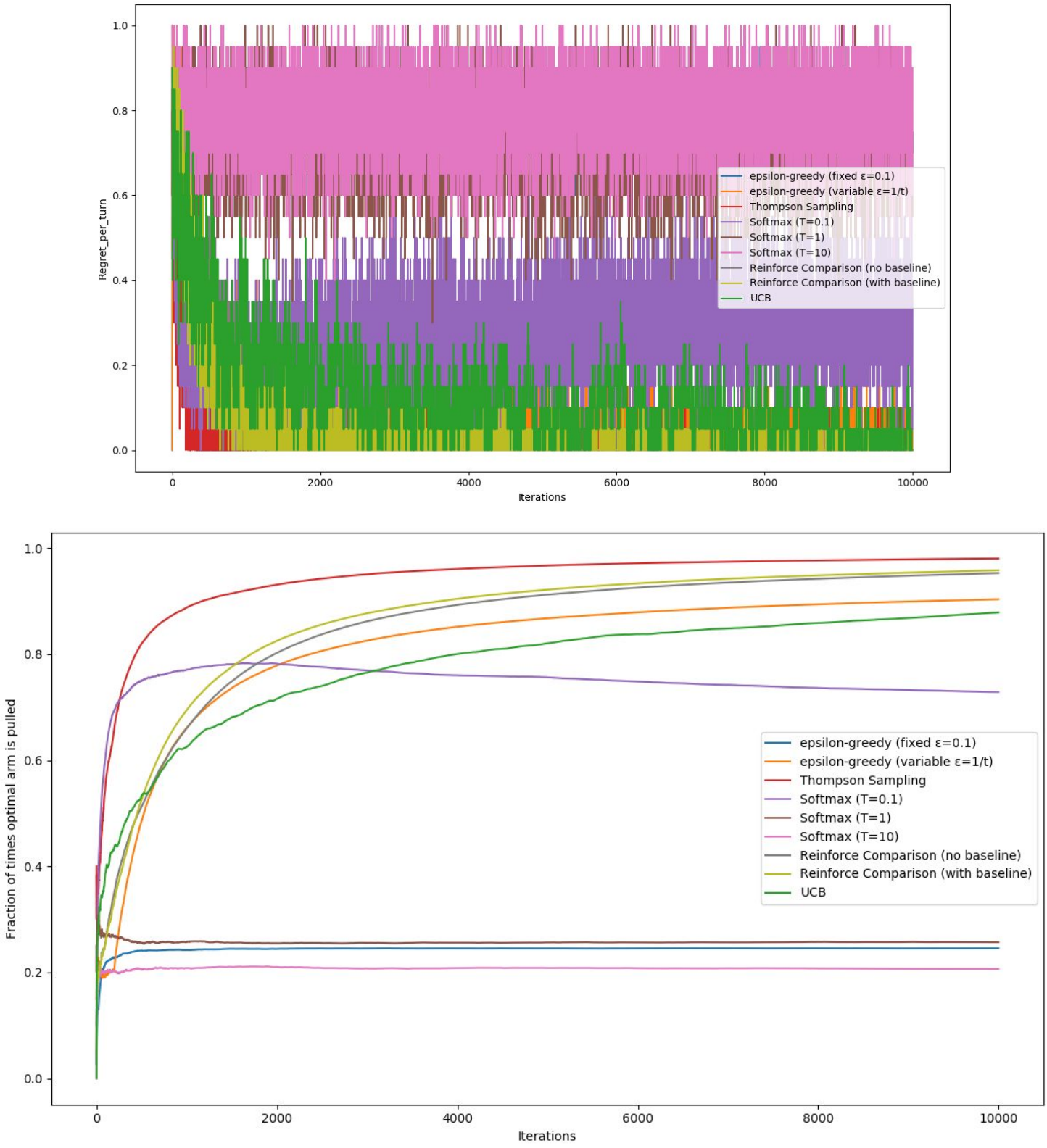
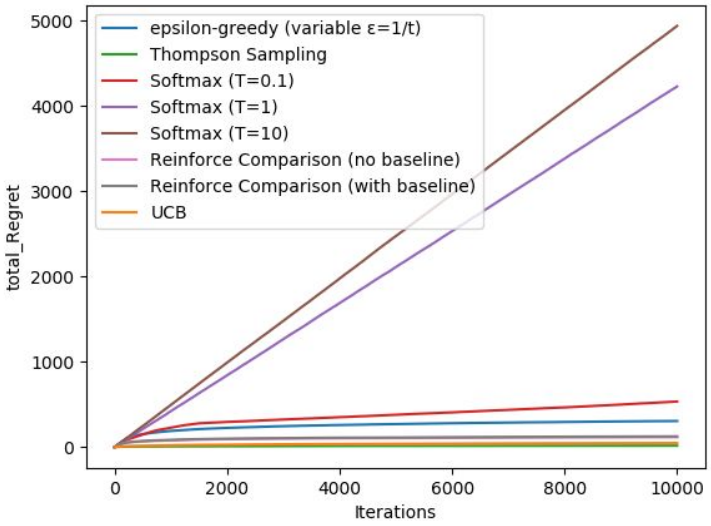
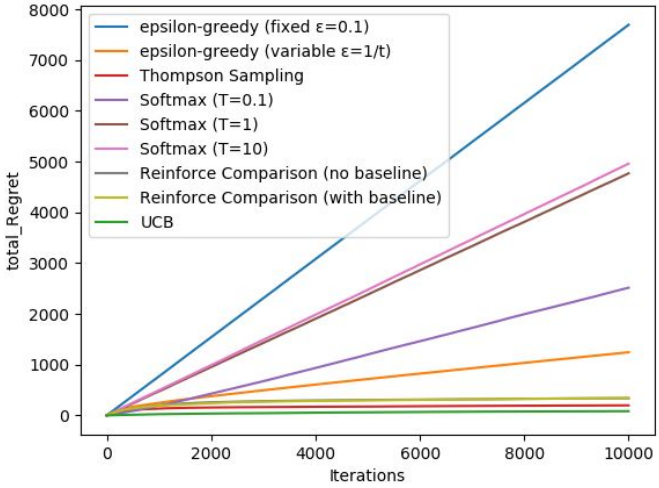
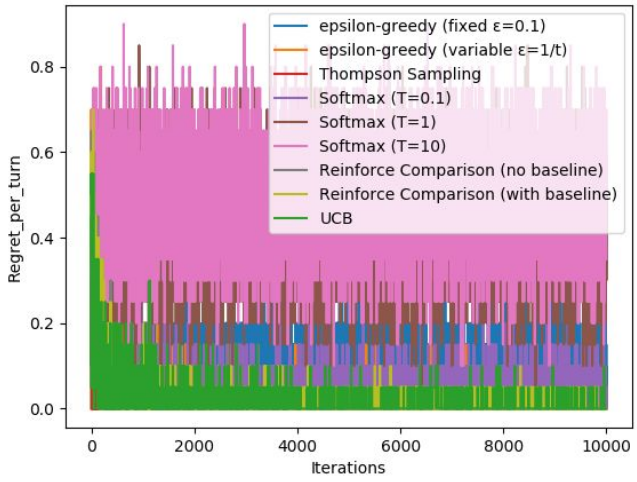
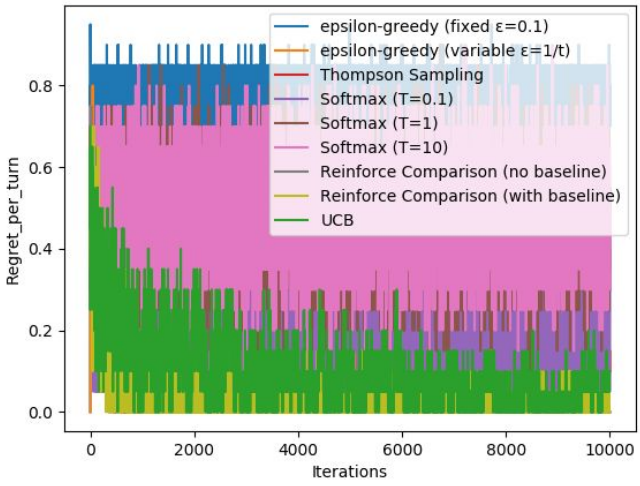


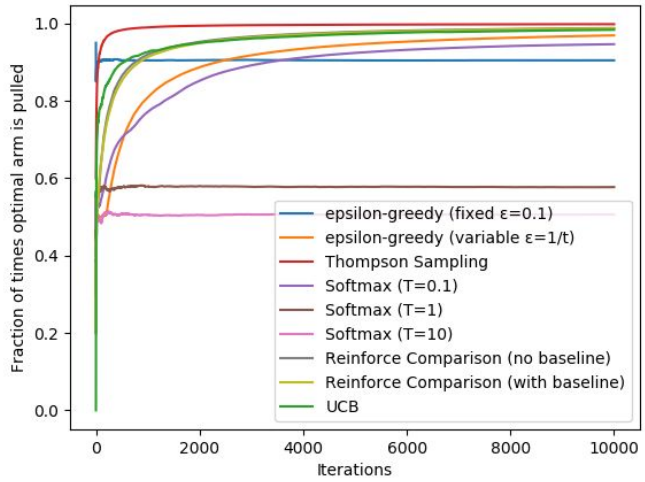
Figure 2: Empirical Results for 5 arms with P = [0.1, 0.4, 0.6, 0.3, 0.7]











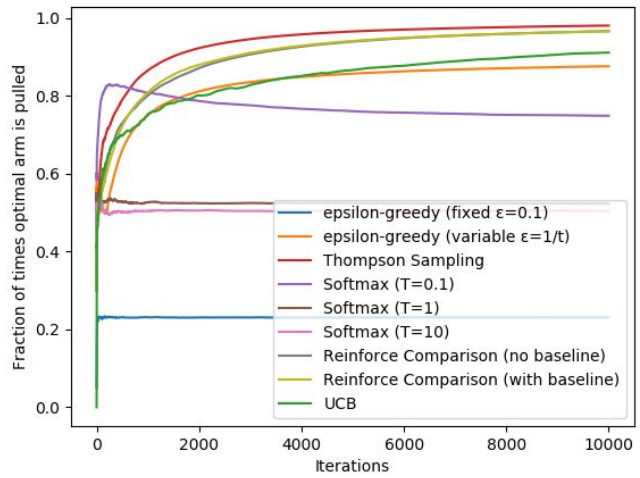


Figure 3: Empirical Results for 2 arms with P = [0.9, 0.6] (Left) and Empirical Results for 2 arms with P = [0.45, 0.55] (Right)

CONCLUSIONS

In general, we observed that the regret increases as the number of arms increases. This is natural since the choices become more at each step. Also, the performance of the algorithms decreases if the expected reward is each arm is approximately close. Thompson sampling performed better compared to others followed by UCB, Reinforce Comparison, epsilon-greedy(0.1), softmax(Temperature=0.1). The performance of the UCB algorithm was comparable or sometimes better than the rest of the algorithms if the number of arms is lesser (K=2,5).

Hence the final ranking is:

Thompson >= UCB > Reinforce Comparison (baseline) > Reinforce Comparison (without baseline) > Softmax >= Epsilon-Greedy (Variable) > Epsilon-Greedy (Fixed)

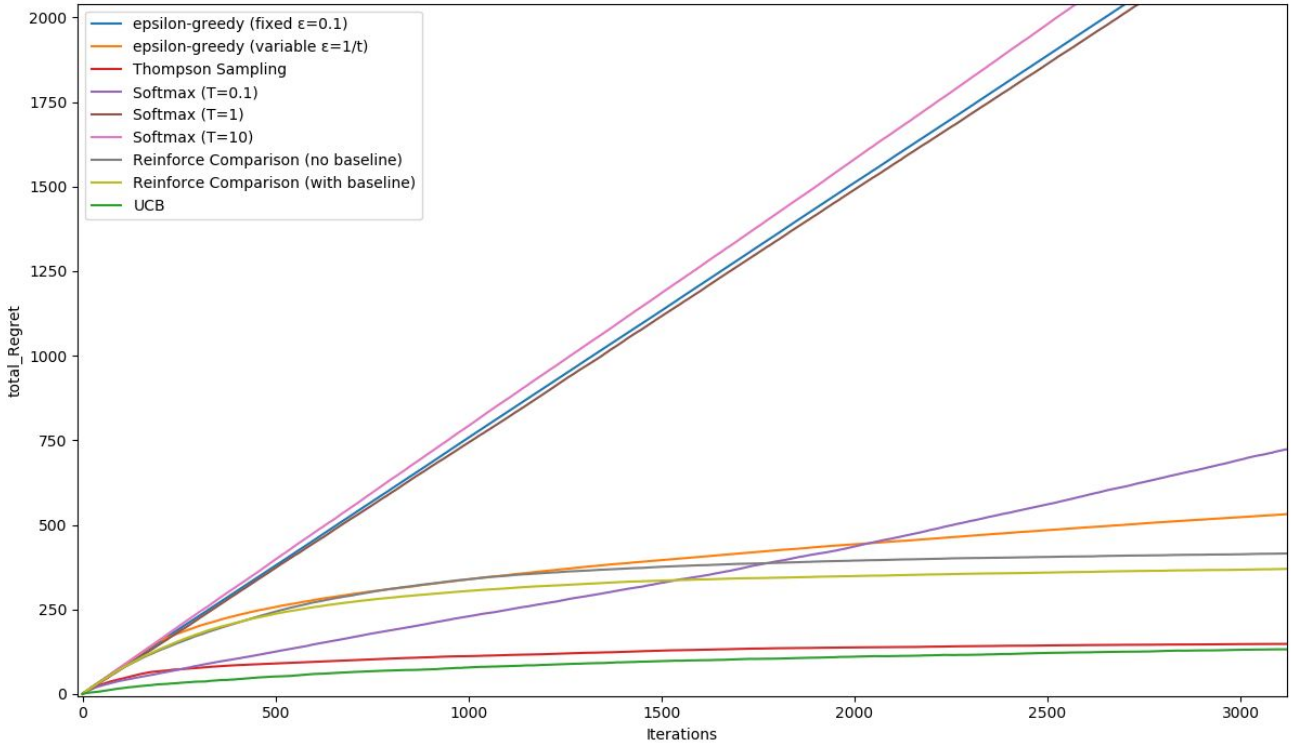


Figure 4: Close-up of the regret as a function of time for the algorithms: Thompson Sampling, UCB and Reinforce give logarithmic regret. Epsilon-Greedy with variable epsilon and Softmax (T=0.1) also give logarithmic regret, thought the constants are higher. Epsilon greedy with fixed epsilon and softmax at higher temperatures gives linear regret.