Multi-Armed Bandit final report

Yujian Liu[1], Qichen Fu[2], and Isaac Skinner[3]

ABSTRACT

Reinforcement learning is inspired by the way infants learn from their experiences. It uses the feedback from its interactions with its environment to learn how to make better decisions. Here we focus on the Multi-Armed Bandit (MAB) model, inspired from the gambling world, a multi-armed bandit is a slot machine with multiple levers. The model assumes each arm has its own probability of winning the jackpot. The goal of learning in the model is to discover which arm has the highest probability of success, and choosing it repeatedly. We take a look at different algorithms for optimizing the model, and how adjustments in the models assumptions can affect which algorithm is best suited for learning. We show each algorithm's performance compared to its theoretical performance [Sutton-Barto] in a series of experiments, and demonstrate the trade-offs between the various methods for solving the problem.

Introduction

Related Work

Reinforcement learning is a broadly studied area in machine learning. It formulates and studies various decision making problems in real life. It consists of two parts: agent and environment. The agent can be in a set of possible states, and it will perform some actions based on the state it is currently in. The environment will then provide the feedback to the agent in the form of reward. Based on the interacting experience with the environment, the agent will try to learn the environment and a set of rules of how to behave in the environment in order to get maximum reward. Reinforcement learning is different from regular MAB in the way that it chooses actions based on states (contextual MAB is able to do that as well), and it also considers the benefit from a series of actions rather than simply the sum of several separate actions. While reinforcement learning is able to simulate more complex problems in real life, MAB can be suitable for some cases, and it has grounded theoretical support. As an introduction project, we will focus further on MAB.

Multi-Armed Bandit Problem

We consider the *stochastic bandits*. The model is parameterized by K the number of arms, and T the number of rounds that an agent can pick an arm. In each round, the agent choose an action and observe the reward from the chosen action. The goal of the agent is to maximize the cumulative reward over T rounds.

In the model, we assume we only observe the reward from the action that we choose, and nothing else. We also assume each action has its reward distribution, and the reward is drawn *iid.* from the distribution every time.

The regret at round t is then defined as the difference between the optimal cumulative reward and the observed cumulative reward till round t.

$$R(t) = \mu^* \cdot t - \sum_{s=1}^{t} \mu(a_s) \tag{1}$$

where μ^* is the expectation of reward of the optimal action, a_s is the action that the agent takes at time step s, and $\mu(a_s)$ is the expectation of reward of action a_s.

The goal of each algorithms is to balance exploring the possible arms while exploiting the expected best arm in order to minimized Regret.

Experiment Design

Experiment model

This paper simulates 4 different methods {Naive, Epsilon-Greedy, Upper Confidence Bound, Thompson Sampling}.

In the experiment, all the bandits give binary output, which is either 0 or 1. The reward distribution is parameterized by bernoulli distribution with probability p. The simulation consists of K arms, T rounds, and E repeated experiments. The parameters are able to be modified by users. The parameters corresponding to the results given is indicated in the figures.

The simulation algorithm is shown below. Input is K, E, T.

Algorithm 1 MAB simulation

```
P ← random((K, 1))
for i in range(E) do
    Observation ← zeros
    Cumulate_Regret ← Zeros
    Generate_Reward ← rand((T, K), ref = P)
    for i in range(T) do
        Arm ← algo(Observation)
        update Observation based on Arm and
        Generate_Reward
        update Cumulate_Regret based on Arm and
        Generate_Reward
    end for
end for
Cumulate_Regret ← Cumulate_Regret/E
return Cumulate_Regret
```

The random method selects a random bandit in each trail. From the observation, the cumulative expected regret is increasing linearly. The Naive algorithm first selecte each bandit for several rounds (exploration phase), then always chooses the bandit with best observed performance. The Epsilon-Greedy algorithm has probability ϵ to choose the bandit with best observed performance so far and probability $1 - \epsilon$ to randomly pick a bandit. The UCB algorithm uses the upper and lower confidence bound to decide which arm is better than others. Thompson Sampling uses the beta

[1] yujianl@umich.edu
[2] fuqichen@umich.edu
[3] jonisask@umich.edu

distribution updated by the observation to decide which arm is better.

In the work by Aleksandrs, it is proven that for the two arm MAB model there are theoretical bounds for regret in each Algorithm. We designed simulations to demonstrate the convergence of each algorithm to its theoretical bound. Each theoretical bound gives a maximum regret that cumulative expected regret will be under. In these experiments we define the cumulative regret to be the total regret up to the point of time t, and the expectation is the average cumulative regret at each time interval t averaged over E experiments. Therefore, when E is small cumulative expected regret is much more random and unlikely to have converged to its theoretical bound, but when E is large the cumulative expected regret with be a smooth function that increases monotonically to its bound.

Results

The results backup the theoretical bounds that were derived in the book.

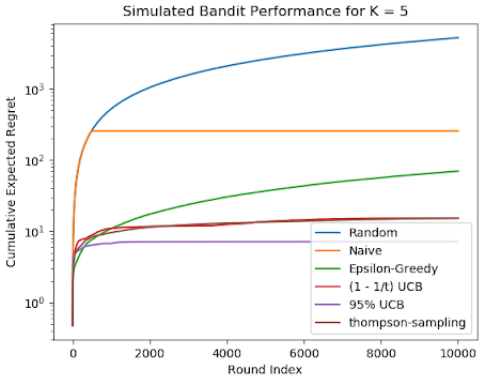

Figure 1. Average Cumulative Regret with parameters {K = 5, T = 10000, E = 1000}.

Note these simulations do not apply to our theoretical bounds since K ≠ 2; however, it demonstrates the relative ability of each algorithm to converge to the optimal arm. Also note that in this simulation in the Naive Algorithm, N (number of rounds to explore) must have been set to a very high number relative to K such that the Algorithm found the optimal Arm with probability approaching 1. This figure also shows the characteristics of each algorithm. For example, Epsilon-Greedy algorithm keeps exploring at all time, so its regret keeps increasing even in later phase. As a result, we can see UCB and Thompson Sampling perform better than other algorithms.

Convergence to Expectation

Due to the inherent randomness of reward at a time interval t, when an individual experiment is run an Algorithm's cumulative regret can deviate greatly from its expectation. This is especially obvious in the case of the Naive algorithm where it finds the correct arm early on, and cumulative regret has a slope of zero for the duration of the experiment. As the number of experiments is increased we see that the Cumulative Expected Regret (We define Cumulative Expected Regret as the average cumulative regret at each time interval t over all the experiments E) converges to its theoretical expectation. As anticipated the regret achieved by each algorithm converged to its expected bound when ran over a large number of experiments; this is most obvious in the case of the Naive Algorithm

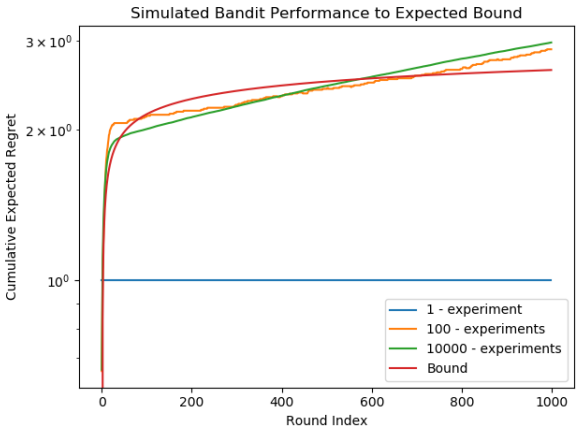

Figure 2. Convergence of Cumulative Expected Regret of Naive Algorithm, due to varying numbers of Experiments {1, 100, 10000}.

Fatal Flaw of UCB

The UCB algorithm has an obvious flaw, there exists a case in which the algorithm will not converge: if the Lower Confidence Bound of any arm is never greater than one or more other arm's Upper Confidence bound. The algorithm is left oscillating between the remaining arms and accumulating more regret. The damage to regret caused by this phenomenon can be seen in figure 4 where the non-converging UCB accumulates about ten times more Regret.

1 Alternate two arms until $\mathrm{UCB}_t(a) < \mathrm{LCB}_t(a')$ after some even round t;
2 Then abandon arm a, and use arm a' forever since.

Figure 3. Pseudo code for UCB Algorithm.

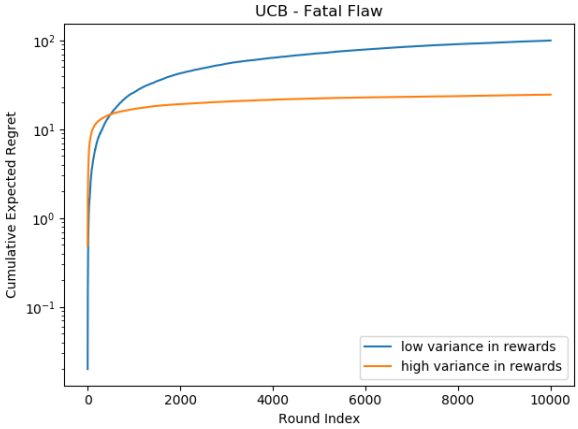

Figure 4. Cumulative Expected Regret of UCB algorithm in both converging and non-converging scenarios.

Alternate Regret Definition

The definition of regret from equation (1) is from the book. While understanding this definition, it is reasonable to think

of another alternate definition of regret: instead of using expectation of reward, we use the observed reward. That brings us the definition

$$R(t) = \sum_{s=1}^{t} (r(a^*) - r(a_s)) \tag{2}$$

where $r(a^*)$ is the generated reward of the optimal action if we choose that, and $r(a_s)$ is the observed reward of the actual action we chose.

This definition incorporates the randomness from generating reward. Since reward is generated from some distribution, we expect this regret to be more unstable. The result is shown below. Note that in both figures, the regret of naive

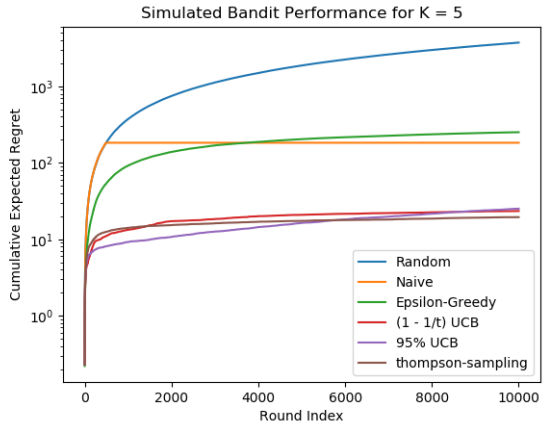

Figure 5. Average Cumulative Regret with parameters {K = 5, T = 10000, E = 100}.

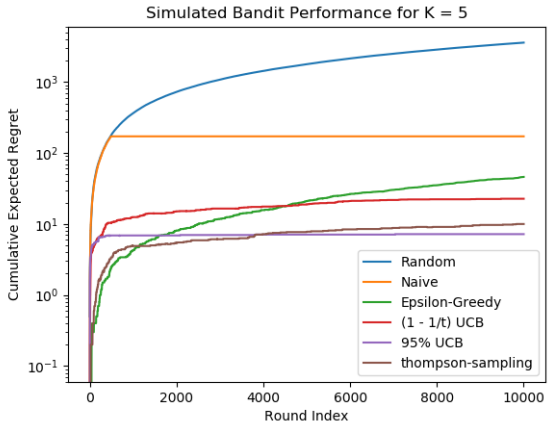

Figure 6. Average Cumulative Regret with parameters {K = 5, T = 10000, E = 10}.

algorithm stops increasing after exploration phase. That is because the number of rounds to explore for each arm is large enough so that we indeed find the optimal arm. Then in exploitation phase, the regret is always 0. Also note that for 10 experiments, the regret is more unstable. Particularly for epsilon-greedy algorithm, there might be less exploring rounds in 10 experiments, which results in less regret in Figure 6.

Conclusion and Future Work

We simulate 4 algorithms for MAB model to plot their cumulative regret. The regret plot conforms to the characteristics of each algorithm. We also show the trend of convergence of cumulative regret to the expected cumulative regret as the number of experiments increases. Furthermore, we study how each of our assumption of our model can affect the result. Particularly, we research on how the reward distribution of K arms will affect different algorithms. Finally, we propose an alternate definition of regret, and simulate to see how this definition differs from that in the book.

Acknowledgements

Sindhu Kutty

References

Aleksandrs Slivkins (2019). *Introduction to Multi-Armed Bandits*. Chapter 1.

Richard S. Sutton and Andrew G. Barto (2018). *Reinforcement Learning: An Introduction*, second edition. The MIT Press Cambridge, Massachusetts.