Master Project Report - Empirical Research on Multi-armed Bandit Problems with Free Pulls

Tiancheng He

April 16, 2020

Abstract

The project tries to model a class of real problems as multi-armed bandits problems with free pulls. The performance of different algorithms with various free pull policies is compared and analyzed based on simulation results. Proving the existence of counter intuitive examples where extra information worsens performance, the project explores certain properties of free pulls.

1 Introduction

Multi-armed bandits problem is a classical reinforcement learning problem with abundant research results. It is applied to various areas such as online advertisement[1], information retrieval[2], clinical trials[3] etc. A number of variants are proposed to model different problems. However, we find bandits with free pulls hasn't been studied yet. This project focuses on empirical research on this variant. The report will first explain the meaning of the new setting and applicable scenarios. Simulation results will be presented and analyzed. The most significant finding is the existence of counter intuitive examples that extra information worsen the performance. We will explain the reasons for those cases and briefly discuss about the methods to utilize extra information.

2 Background and motivation

The typical traditional multi-armed bandits(MAB) problem refers to a problem that a gambler wants to maximize his profits(minimize his regrets) when he has T coins and there are n single-armed bandits(arms) each of which has unknown probability to yield the reward. The essence is the exploration-vs-exploitation question that the gambler must balance the choice to explore unknown probability and the choice to exploit the best known arm.

An interesting question is what if the gambler has options to observe reward signals for free, without cost or real reward, like watching others pull an arm. The application

motivating this project is a temperature scheduler for a public room which can ask users what temperature they prefer. An arm is a specific schedule of temperature for a whole day. The reward is the level of users' satisfaction. A survey of a specific schedule for all users can be considered as a free pull, because the schedule isn't realized and users don't experience any comfort or discomfort. We assume the average satisfaction of all users for a schedule is stochastic. This is reasonable because the population differs every day and a user may not report or know or have the exactly fixed optimal schedule for him. We also assume the cost to take a survey is quite small compared to the real discomfort. To make this assumption more reasonable and the question more feasible to study, we only consider free pulls in a fixed frequency, meaning one free pull every k rounds.

The scheduler can control the policies of both normal pull and free pull in this setting, which this project focuses on. However, the free pull model can be also extended to any MAB settings with shared information where free pull can't be controlled such as observing other gamblers pull arms, or receiving collaborator's exploring results.

3 Simulation

We will empirically study on this problem via simulation. The objective is to explore some properties/effects of free pulls by testing different combination of normal/free pull policies on different reward distributions. The settings will base on the temperature scheduler problem, while several settings are simplified for fast simulation.

3.1 Formulation

To make the setting more clear, we will formulate the free pull MAB problem as follows: given n arms, the reward of an arm A_i is a random variable with expectation μ_i. Define $\mu^* = max_{i=1}^{n}(\mu_i)$. An algorithm can do a normal pull on arm A_i at round t, receiving reward r_t. It can also do a free pull every k rounds, denoted as $r'_{i,t}$ which means pulling an arm A_i after t normal pulls. The regrets after T rounds is defined as $R(n) = \Sigma_{t=1}^{n}\mu^* - r_t$. The target is to minimize $E[R(T)]$ for a given T. Note that the rewards of free pulls don't appear in the definition of regrets because free pulls won't incur real costs or rewards. It just provides more information for an algorithm to estimate μ_i.

3.2 Simulation settings

1. Normal pull policies

3 algorithms are popular applied for a traditional MAB problem, ϵ-greedy, upper confidence bound(UCB) and Thompson sampling.

- An ϵ-greedy algorithm[4] exploits the best known arm with probability $1 - \epsilon$ and explores a random arm with probability ϵ.

- A UCB algorithm[5] will rank each arm by adding its confidence interval to the estimated mean and choose the best one. An arm's confidence interval is a monotonic decreasing function of its pulling times.

- A Thompson sampling algorithm[6] assigns each arm a prior Beta distribution. In each round it samples from all arms and chooses the arm with the largest sample, and updates posterior distribution according to the reward.

Although UCB and Thompson sampling are proved to achieve the optimal regret bound[7][8], these algorithms do differ in performance. Therefore, when simulated with free pulls, we only pay attention to how free pulls change the original algorithm performance instead of comparing the performance of different policy combination directly. A natural result is no need to optimize any algorithm.

The figure 1 shows the performance differences of 3 algorithms on our basic setting. The number in the brackets means the value of ϵ. Random works as the baseline.

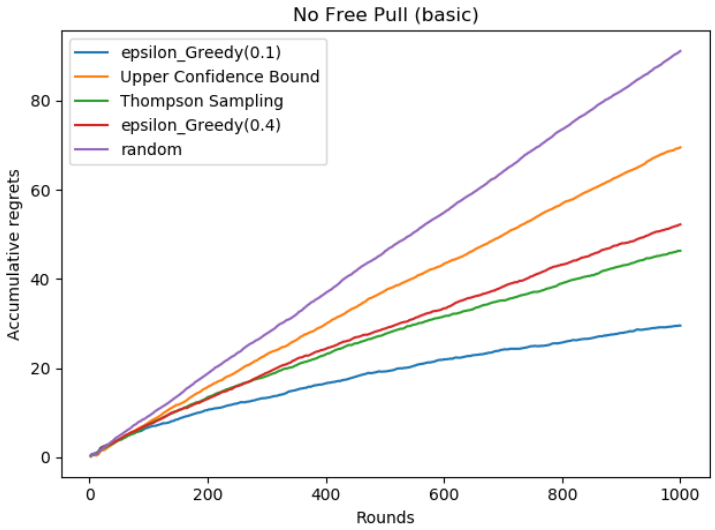

Figure 1: algorithm performance difference

2. Free pull policies

It might be better to focus on exploration because no real reward means it doesn't need to exploit at all. According to such understanding, we consider several free pull policies and predict their performance in advance.

- Same policy as normal pull algorithm. It seems only providing more rounds without utilizing the no real reward property. So it could be a baseline of enhanced performance.

- Pure exploration policy. This pure random exploration is a natural choice for our exploration-oriented understanding. However, it considers all arms equally and may waste pulls on arms that will unlikely be chosen by normal pulls.

- Second best arm policy. The second best arm is ranked by the normal pull algorithm. The intuition is that if the normal pull is allowed to ask one more question, it would like to know how good the most competitive candidate is. So we choose it in free pull. If it's better, the next normal pull will choose it instead, otherwise the next normal pull will choose the best one.
- Worst arm policy. The worst arm is also ranked by the normal pull algorithm. The intuition is that if normal pull choose the worst arm, it will lose the most, so free pull can explore it at first to avoid the loss.
- Least pulled arm policy. It's a variant of worst arm policy. The least pulled arm is considered as the worst arm. However, the intuition is different: the arm needing exploring most is the arm explored least.
- Another normal pull algorithm. Just to see how a different algorithm collaborates with the base one.

Worst arm policy and least pulled arm policy may stick to bad arms which are unlikely chosen by normal pulls, so they are enhanced with successive elimination(SE). SE means an arm bad enough will be eliminated from candidates and "bad enough" means this arm's estimated mean plus its confidence interval is less than the best arm's mean minus the best arm's confidence interval.

3. Reward distribution In the temperature scheduler problem, an arm is a schedule for one day. Our simulation simplifies an arm as a specific temperature integer, because a schedule as an arm or continuous temperature means there are enormous arms. According to the recommended range of room temperature, we choose 16 arms, from 60°F to 76°F. 68

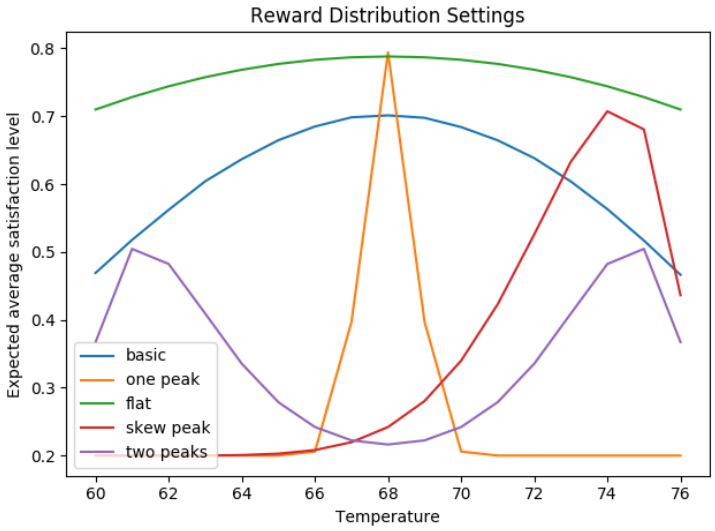

Figure 2: reward settings

Based on some assumptions about people preferences, we assign arms with certain expected rewards, shown as "basic" (blue line) in the figure 2. The x-axis is temperature. The y-axis is expected average satisfaction level, 1 meaning all users are perfectly satisfied, 0 meaning totally unsatisfied. Other settings are some special cases to test how the policies perform on different situations.

- basic. It looks like normal distribution because we assume the preference is sampled from a normal distribution.
- one peak. Only one arm is extremely good. Other arms are similarly quite bad.
- flat. Although there is one best arm, all arms are very similar.
- skew peak. Half of arms are similarly quite bad. The other half is sharply normal.
- two peaks. The best arm and the second best are quite similar.

4. Other settings

- Bernoulli random variable. The basic Thompson sampling algorithm only works for Bernoulli random variables, so we define arms as Bernoulli random variables instead of continuous variables. Although continuous variables are more suitable to our settings, the conclusions should stay similar.
- 500 rounds. 500 rounds should be reasonable for 17 arms.
- 15000 trials or more. We run each experiment for 15000 times. If the results still contain a level of randomness, we will run more trials until the trend in results is clear.
- the frequency of free pull k is 1. To see the effects of free pull more clearly, we allow a free pull following each normal pull.

3.3 Results

The results are figures 3 to 7. The x-axis is rounds. The y-axis is reduced regrets by free pulls. The larger reduced regrets means the stronger enhancement of performance by this free pull policy.

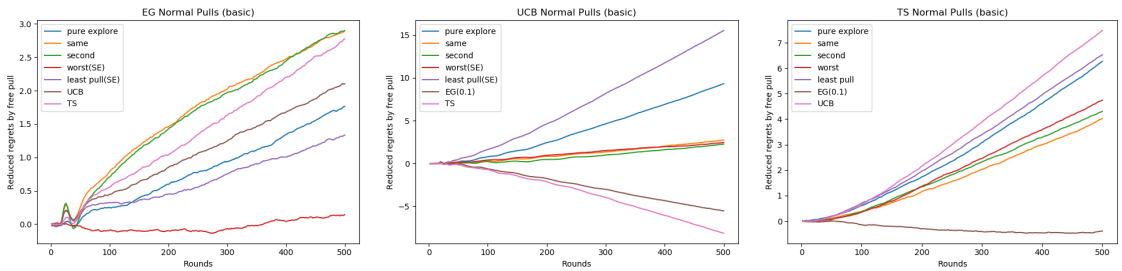

Figure 3: performance difference on basic setting

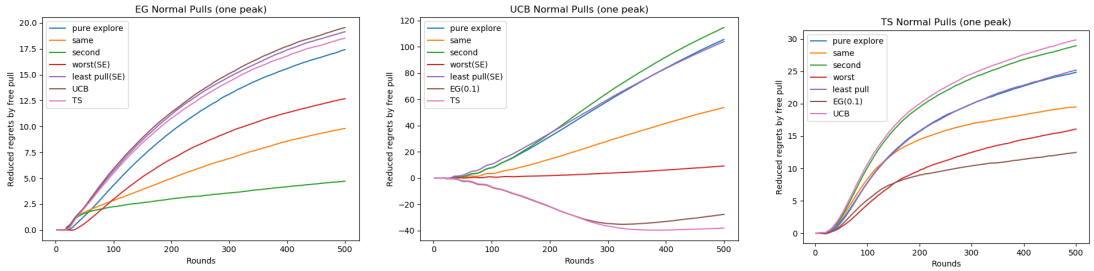

Figure 4: performance difference on one peak setting

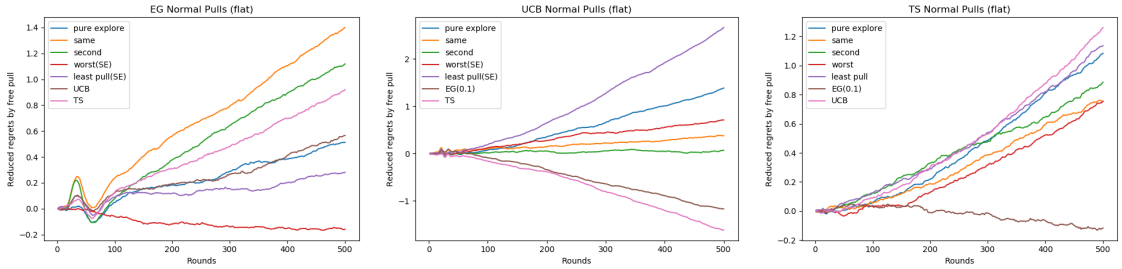

Figure 5: performance difference on flat setting

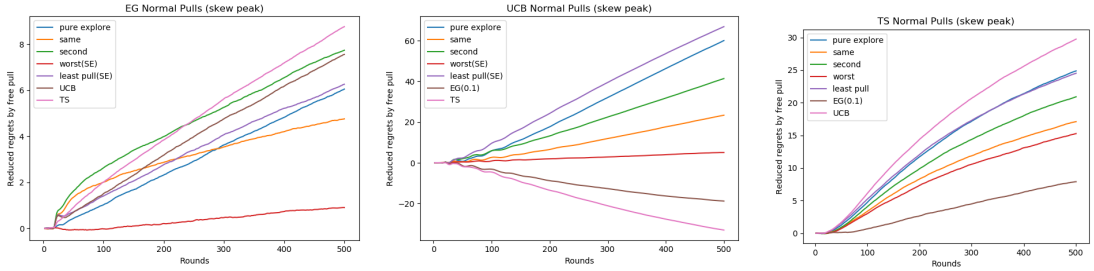

Figure 6: performance difference on skewed peak setting

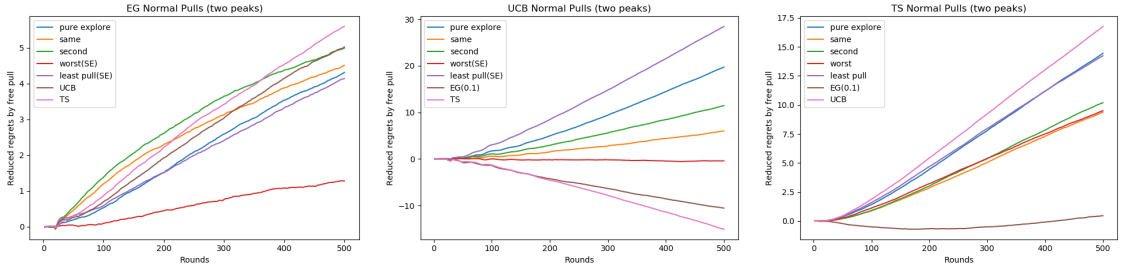

Figure 7: performance difference on 2 peaks setting

4 Analysis

4.1 The existence of bad cases

The first significant observation is that in some settings free pulls can worsen performance. We supposed free pulls will always help because more information should always be better but the results disagree.

To confirm this unusual observation, we construct a much simpler example and do numerical calculation of the expected regrets. Consider only two arms. A always yields reward 0.1. B yields 1 or -1 with probability 0.5, so its expectation is 0. Suppose the normal pull policy is a pure greedy algorithm, always choosing the best known arm, and the free pull policy is always choosing B. This should be the simplest stochastic problem instance with the simplest free pull policy. The simplicity allows us calculate the expected regrets by enumerating all possible events. As shown in figure 8, the regrets of free pull can exceed the regrets without free pull even in 10 rounds. In fact, a pure exploration free pull policy is even worse. We can see from figure 9 the pure exploration policy has more regrets than the always bad arm policy by nearly 1.

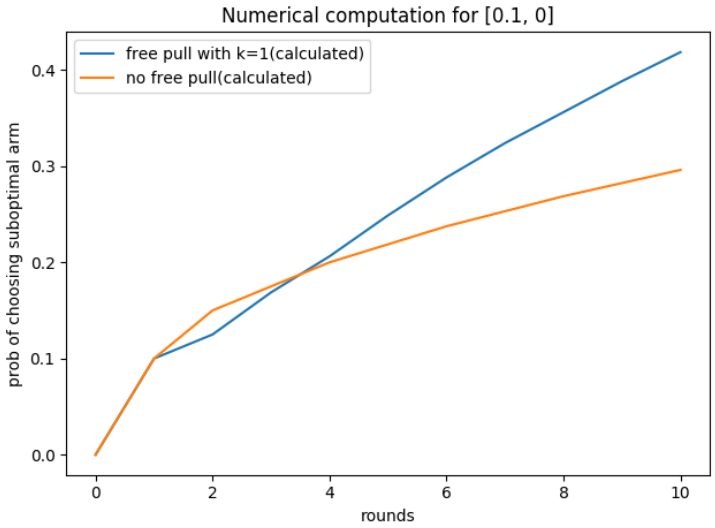

Figure 8: numerical computation for counter example

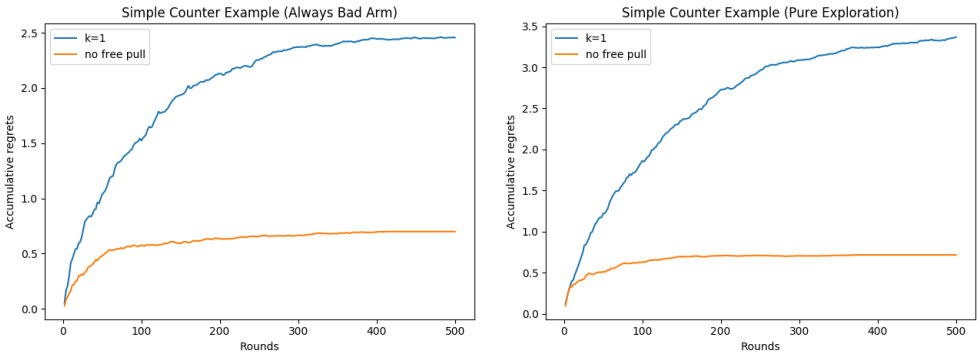

Figure 9: different policies in long run

Our explanation is that free pulls give bad arms chances to be overestimated even if they are abandoned by normal pulls. In this counter example, the best arm gains no benefits from any extra information. It is always 0.1. Without free pulls, if A is pulled, it will be pulled forever. However, with free pulls, as long as the free pull has non-zero probability to choose B, B is possible to be chosen by normal pulls, meaning free pull pulls B and receives 1 enough times to change estimated mean to be larger than 0.1.

This effect decreases when the variance decreases or the normal pull will choose the "abandoned" arm. As shown in figure 10, In the left plot, we adjust the variance of the bad arm from 1 to 0.3, free pulls help eventually. In the right plot, we adjust the normal pull algorithm to ϵ-greedy with $\epsilon = 0.05$. The regret without free pulls increases relatively more.

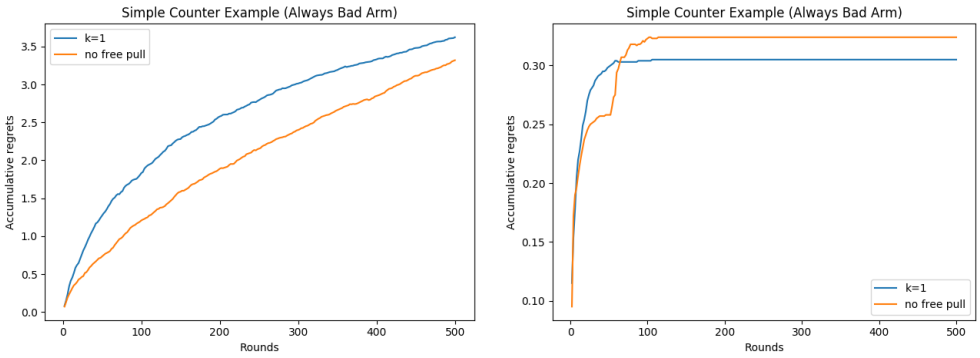

Figure 10: decreasing effect

4.2 A lesson from bad cases

The last example shows the difference of arms' variance matters. In fact it represents arms' different ability to benefit from extra pulls. If a bad arm has relatively larger variance and is chosen by free pulls, it might be overestimated. As a result, we learn an important lesson: a good free pull policy shouldn't let normal pull algorithm overestimate bad arms. That

might be why the worst arm policy works poorly on ϵ-greedy algorithm. However, is it the same reason that Thompson sampling policy worsens UCB normal pulls and ϵ-greedy free pull policy worsens both Thompson sampling and UCB normal pulls?

Our answer is yes, but the mechanism works in an opposite way: it helps underestimating good arms. Both UCB and Thompson sampling rank all arms in each round. Less pulled arms will receive extra scores. If a free pull policy keeps pulling good arms, good arms will receive less extra scores, equivalent to giving bad arms more extra scores. This is clearly seen in the execution of UCB. In the early stage, the confidence interval is the major part of an arm's score. An arm's mean is between 0 and 1 but its confidence interval can be more than 2, although it decreases in the speed of $O(\log n)$. Without free pulls, all arms' confidence intervals decrease in the same rate. However, with free pulls keep pulling good arms, UCB normal pulls will keep pulling bad arms until their confidence interval is close or the confidence interval is small enough.

In sum, in these bad cases, the free pulls take over exploration of good arms so that the normal pulls choose good arms less. A strong proof is the fact that the worsen policies are exactly other normal pull algorithms that perform better in common MAB settings. As shown in figure 1, Thompson sampling is better than UCB, as well as ϵ-greedy(0.1) is better than Thompson sampling and UCB, which exactly corresponds to the bad cases.

4.3 Comparison of free pull policies

Except the bad cases discussed above, most free pull policies improve performance, but there is no policy working universally well. The best free pull policy is different in the different settings. Pure exploration and least pulled arm policies are generally good for UCB and Thompson, but not good for ϵ-greedy. The same algorithm policy and worst arm policy work bad for UCB and Thompson sampling, but can work quite well for ϵ-greedy. The reward distribution also changes the performance of these policies. For example, the same algorithm policy is the best in basic and flat settings, but the second worst in skewed peak settings.

However, the understanding of bad cases can help categorizing what free pull policies fit each normal pull algorithm. ϵ-greedy favors policies more likely to choose good arms. UCB and Thompson sampling favor policies pulling less explored arms.

4.4 Some questions

Is there any policy that always help no matter what normal pull algorithm is? Without knowing the reward variance difference, we can't be sure. If given equal variance, we think pure exploration should always help, because it's a weak version of pulling all arms and pulling all arms should always help with equal variance.

Is there any normal pull algorithm that benefits from any free pull? We suspect the answer is no. At least the tested three popular algorithms can be hurt even with equal variance. However, it might be easy to figure out some defensive ways to utilize free pulls. A simple idea is to set up some standard to use free pulls' signals. For example, normal

pulls will wait for free pulls to pull all arms at least once, and then retrieve equal amount of signals from all arms until any arm has no signal.

5 Conclusion

This project models a new variant of MAB problem that allows free pulls. It is applicable to scenarios like temperature scheduling for a shared room. Based on simulation of 3 popular MAB algorithms combined with several free pull policies, we find in certain circumstances free pulls may hurt the performance. It shows that in the dynamic decision making process, extra information can help reducing uncertainty, but also can break decision assumptions to make original decision progress perform worse. For ϵ-greedy, the assumption is that bad arms will be abandoned and will be chosen in very small probability, so policies often pulling bad arms may hurt. For UCB and Thompson sampling, the assumption is that good arms and bad arms share the similar exploration rate until it's quite confident to distinguish them, so policies often pulling good arms may hurt.

Our empirical results can help making decisions for real applications. If given a normal pull algorithm, we suggest ϵ-greedy using policies more likely to choose good arms, while UCB and Thompson sampling using policies favoring less explored arms. If not given a normal pull algorithm, we suggest using pure exploration. If one can't choose the free pull policy, we suggest to use some defensive standard to ensure free pulls won't hurt the original algorithm.

References

[1] Abe, Naoki. "Learning to optimally schedule internet banner advertisements." In Proc. of 16th Int. Conf. on Machine Learning, pp. 12-21. 1999.

[2] Radlinski, Filip, Robert Kleinberg, and Thorsten Joachims. "Learning diverse rankings with multi-armed bandits." In Proceedings of the 25th international conference on Machine learning, pp. 784-791. 2008.

[3] Villar, Sofía S., Jack Bowden, and James Wason. "Multi-armed bandit models for the optimal design of clinical trials: benefits and challenges." Statistical science: a review journal of the Institute of Mathematical Statistics 30, no. 2 (2015): 199.

[4] Watkins, Christopher John Cornish Hellaby. "Learning from delayed rewards." (1989).

[5] Agrawal, Rajeev. "Sample mean based index policies by o (log n) regret for the multi-armed bandit problem." Advances in Applied Probability 27, no. 4 (1995): 1054-1078.

[6] Thompson, William R. "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples." Biometrika 25, no. 3/4 (1933): 285-294.

[7] Auer, Peter, Nicolo Cesa-Bianchi, and Paul Fischer. "Finite-time analysis of the multi-armed bandit problem." Machine learning 47, no. 2-3 (2002): 235-256.

[8] Agrawal, Shipra, and Navin Goyal. "Further optimal regret bounds for thompson sampling." In Artificial intelligence and statistics, pp. 99-107. 2013.