Proceedings of the 6th International Conference on Computing and Data Science
DOI: 10.54254/2755-2721/68/20241405

Performance variance in Multi-Armed Bandits: In-depth analysis of three core algorithms

Bowen Chen

Hainan International College, Communication University of China, Hainan, China

202229013098N@cuc.edu.cn

Abstract. In real-time strategy games, players grapple with uncertainty regarding the best strategy for victory. This paper delves into multi-armed bandit (MAB) algorithms as potential solutions. The theoretical foundations of MAB are explored, with a focus on the crucial balance between exploration and exploitation. An experimental comparison of the Explore-Then-Commit (ETC), Upper Confidence Bound (UCB), and Thompson Sampling (TS) algorithms is conducted, showcasing their varied performance. Beyond gaming, the paper also examines the broader applications of MAB algorithms in fields such as healthcare, finance, and dynamic pricing in online retail, highlighting their versatility. A significant portion of the study is dedicated to implementing the UCB and TS algorithms in StarCraft, a popular real-time strategy game. The performance of these algorithms is assessed by calculating cumulative regret values, a metric critical to understanding their effectiveness in decision-making contexts. The results indicate that both UCB and TS algorithms substantially improve players' win rates in StarCraft. However, the study acknowledges existing challenges and the need for further research in this area. The use of MAB algorithms in complex, dynamic environments like real-time strategy games presents a rich avenue for exploration and holds significant promise for enhancing decision-making strategies in diverse domains. This research, therefore, not only contributes to the understanding of MAB algorithms in gaming but also underscores their potential in various other sectors.

Keywords: muti-armed bandit, real time strategy game, balance between exploration and exploitation.

1. Introduction

In real-time strategy games, players continually face decision-making challenges, with the game's outcome hinging on their choices. These players aim to adopt the most profitable strategy to maximize success [1]. Despite drawing on personal experience, determining the most effective strategy remains a quandary. Leveraging algorithms for optimal decision-making can significantly enhance a player's likelihood of winning and deepen their understanding of different strategic benefits.

The multi-armed bandit (MAB) approach is an effective solution for the exploration-exploitation dilemma, addressing the challenge of balancing the two for maximum cumulative reward. MAB has found applications in online marketing, medical trials, and website design, assisting in decisions like optimizing advertisement placement for enhanced click-through rates [2]. Similarly, real-time strategy games, akin to these fields, necessitate maximizing rewards for victory. Players must consider a balance

DOI: 10.54254/2755-2721/68/20241405

between exploring new strategies and exploiting known ones. Additionally, the advent of upgrades and downloadable content has simplified dynamic strategy alterations compared to the past.

This paper proposes employing MAB algorithms to tackle decision-making in real-time strategy games, focusing on "Space Craft," where continual decision-making impacts outcomes [3]. Given the complexity of modeling the entire game, the study concentrates on the initial decisions, establishing a simpler model. Classic MAB algorithms such as Explore-Then-Commit (ETC), Upper Confidence Bound (UCB), and Thompson Sampling are tested using a dataset derived from game simulations [4]. These algorithms are evaluated based on cumulative regret, a crucial metric for assessing algorithmic efficiency. The study aims to ascertain the most suitable algorithm for real-time strategy gaming through comparative analysis.

2. Relevant Theories

2.1. Definition and Fundamentals of Multi-Armed Bandits

The multi-armed-bandit (MAB) problems exist in almost every aspect of the real world, from recommender systems in streaming platform to clinic trials for drugs. In the problem, decision-maker will face a series of sequential choices, and the goal is to choose the best choice in every round [5]. At the beginning, the decision-maker choose some of the choices to explore the reward, called exploration phase. Then he can make an optimized choice according to the information that got from the exploration phase, which is called exploitation phase. For example, in the recommender system for advertising, all the advertisement can be posted for a period, and the decision-maker analyzes the click-through rates (CTR). After exploration, one or several of the advertisements with higher CTR are chosen to be posted further to maximize the reward. One important point is that the exploration and exploitation need to be balanced because more exploration causes more loss (e.g. advertisement with less CTR are posted for more times) and more exploitation produces less reward (e.g. a sub-optimal advertisement rather than an optimal one is chosen) [6].

To abstract the problem into a mathematical model, it's assumed that there are k actions, called arms. The reward of each arm is not known before it is selected but follows some kind of probability distribution. In every round, the decision-maker pulls one of these arms to get a random reward x_t. Pulling one arm repeatedly yields random rewards that are independently dispersed over time and unaffected by other arms [7]. The aim is to maximize the cumulative reward in n rounds (n is defined as horizon). The cumulative reward is defined as $\sum_{t=1}^{n} x_t$. Equivalently, the performance of the strategy that determines how to choose the arm can be measured by the regret. The expected total regret for n rounds is defined as:

$R(n) := E[\sum_{t=1}^{n}(\mu^* - \mu_t)] = E[\sum_{i=1}^{k} T_{i,n}\Delta_a]$. Here, μ^* denotes the reward from the best arm, and μ_t represents the reward in round t. Furtherly, $T_{i,n}$ denotes the total number of times arm i is played from round11to n, and $\Delta_a = \mu^* - \mu_t$, which is called the sub-optimal gap. The expectation is considered for both randomness in results, which may alter the algorithm's sequential decisions, and any randomization within the algorithm [8].

2.2. Explore-Then-Commit Algorithm

Explore-Then-Commit Algorithm is a classical algorithm that can be applied to multi-armed-bandit problems. The algorithm has two consecutive phases called exploration and commitment. During the exploration phase, the decision maker randomly investigates each arm. Then in the whole commitment phase, the optimal arm that is determined by the exploration is committed. Given that there are k arms and n rounds will be played in total, the decision maker determines the m value which represents the number of times each arm will be explored. For each round t from 1 to m*k (in the exploration phase), choose arm I_t = (t mod k) + 1 and get a reward X_t. Then the average reward of each arm is calculated to determine the optimal one. The average reward from arm i until round t is defined as $\hat{u}_i(t) =$

DOI: 10.54254/2755-2721/68/20241405

$\frac{\sum_{s=i}^{t} T_{i,t} X_s}{\sum_{s=i}^{t} T_{i,t}}$, $T_{i,t}$ denotes the total number of times arm i is played from round1to t. From round m*k+1 to n (in the commitment phase), choose arm $I_t = \arg\max_i \hat{u}_i(\text{mk})$. As shown in Table 1.

Table 1. Explore-Then-Commit Algorithm.

Algorithm 1 Explore-Then-Commit Algorithm

Input m, n

Exploration phase:

for t = 1 to m*k **do**

play arm I_t = (t mod k) + 1

end for

Calculate $\hat{u}_i(t) = \frac{\sum_{s=i}^{t} T_{i,t} X_s}{\sum_{s=i}^{t} T_{i,t}}$

Commitment phase:

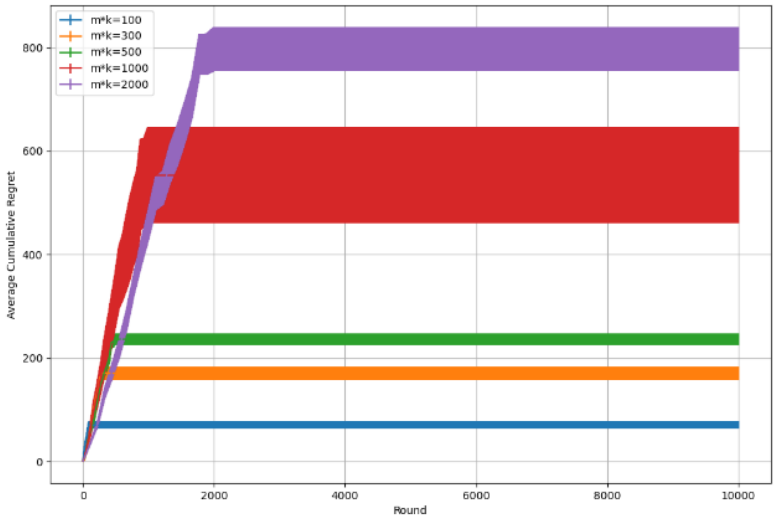
for t = m*k+1 to n **do**

play arm $I_t = \arg\max_i \hat{u}_i(\text{mk})$

end for

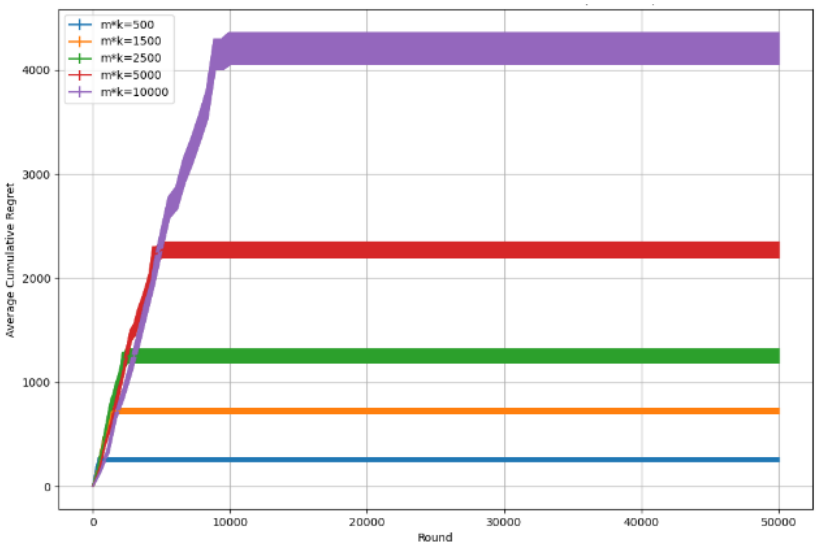
There are two parameters that can influence the performance of ETC algorithm. In theory, it has been proved that when $m = \frac{4\log n}{\min_{i \neq optimal} \Delta_i^2}$, the algorithm has the lowest regret bound [9]. In practical implementation, choose $m = \lceil \frac{4\log n}{\min_{i \neq optimal} \Delta_i^2} \rceil$. However, it is commonly difficult to know Δ_i(the sub-optimal gap) in advance. Meanwhile, the 'best' m value may not perform best in practical problem, because the regret bound only represents the worst case in theory while the regret may not cumulate so much to reach the bound.

In the experiment, the Movielens 1M dataset is used to stimulate the MAB problem. The dataset contains 1 million ratings for 4000 movies from 6000 users, which is divided into three tables: ratings (from 1 to 5), user information and movie information. It is assumed that each genre is an arm. The ratings of all the movies in one specific genre is the reward of this arm. In each round, randomly selecting a movie rating stimulates the action of pulling an arm. The optimal arm is defined as the genre that has the largest mean rating and the optimal reward is the largest mean rating. As shown in Figure 1.

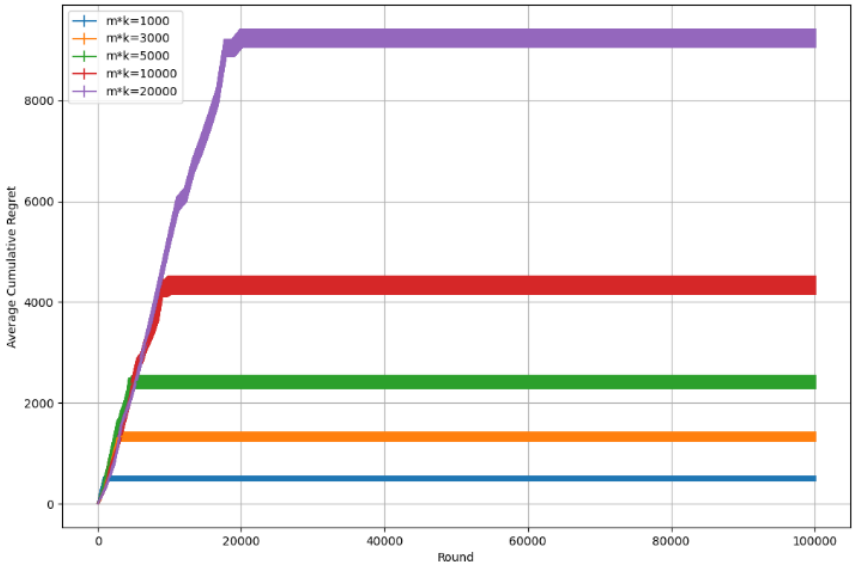


(a)

DOI: 10.54254/2755-2721/68/20241405



(b)



(c)

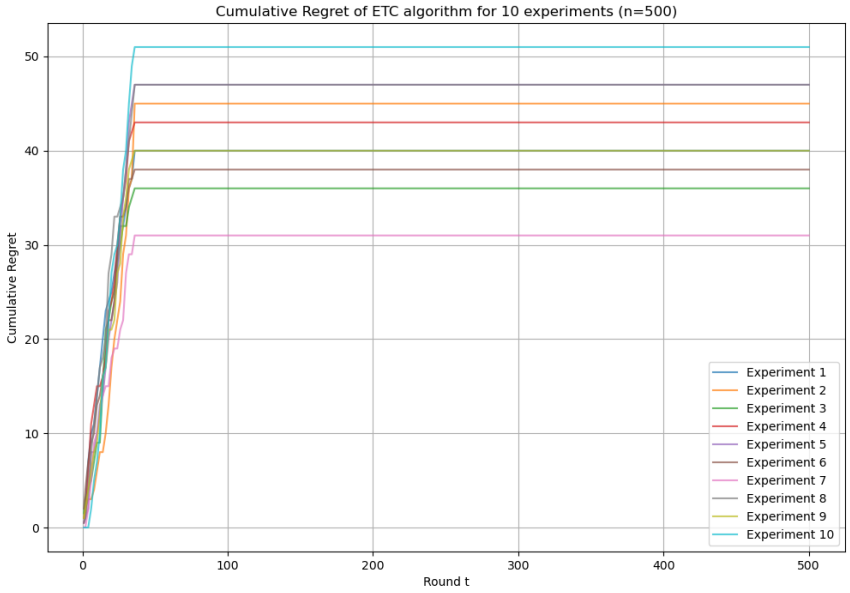
Figure 1. ETC Performance for Different m Values with error bar(n=10000) (Photo/Picture credit: Original).

Here the horizon is set as 10000, 50000, 100000, and the m*k (k equals to 18) values equal to 1%, 3%,5%,10%,20% of the horizon. Each of these three experiments is operated 10 times to calculate the average cumulative regret, making the result more universal and reliable and the error bar is plotted to show the data fluctuation. In the same horizon, as m*k increases, the final cumulative regret grows as well. It makes sense that longer exploration spends more rounds exploring the sub-optimal arm. Focusing on the exploration phase, though it's hard to distinguish, the curve with larger m*k value is flatter than the others, which means in the same round it produces less regret. However, the conclusion doesn't mean that m*k value can increase without limit even regardless of the cost of implementing the exploration because there is no significant and effective improvement in the situation that m*k equals to 10% and 20%, particularly when n is large (the curve is closer when n is 100000 compared with

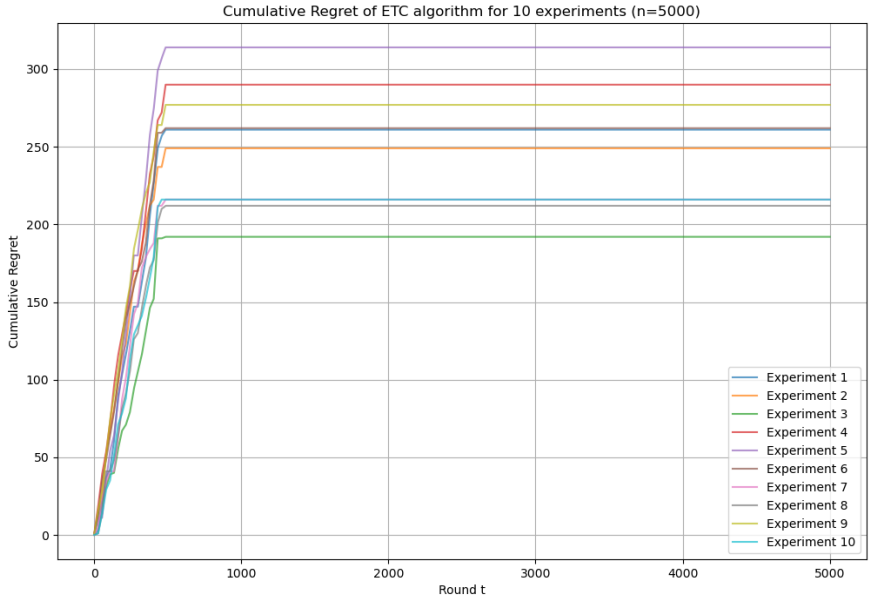
DOI: 10.54254/2755-2721/68/20241405

10000 and 50000). One confusing point is that the regret will not grow in the commitment phase which means the reward the agent get in the commitment phase is constant. The reason is that regret is defined as the expected value of the reward we lose by taking sub-optimal decisions. If the optimal arm is selected, there is no need to stimulate selecting a movie rating of the best genre, because the goal of the experiment is to evaluate the performance of algorithm but not to implement it into practical problem. However, one thing must be guaranteed is that every arm can be explored in the exploration otherwise the regret can continue to grow because it is unknown whether the unselected arm is the optimal arm. In this case, k is 18, and the shortest exploration phase is 100, guarantying the precondition.

Another parameter that affects the performance of ETC algorithm is the horizon. It has been proved that the standard ETC algorithm is not an anytime algorithm, and the regret boundincreases with the horizon [10]. Therefore, the horizon needs to be known before it is implemented. As shown in Figure 2.

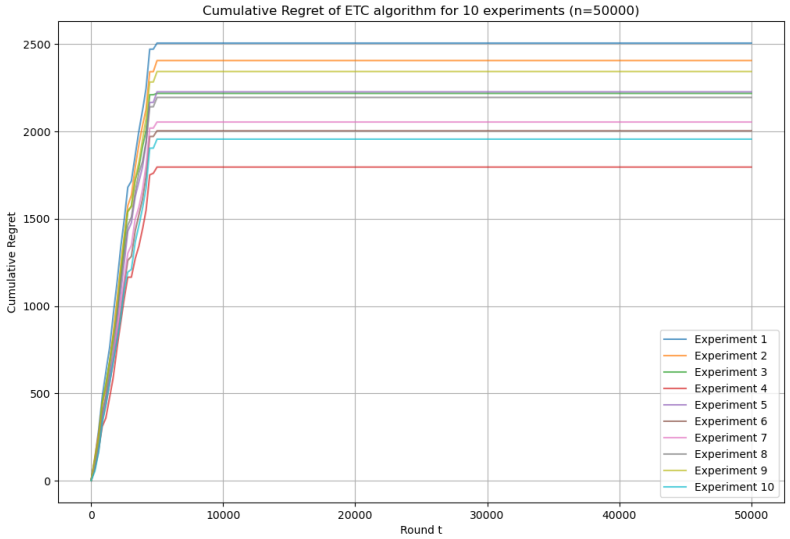


(a)

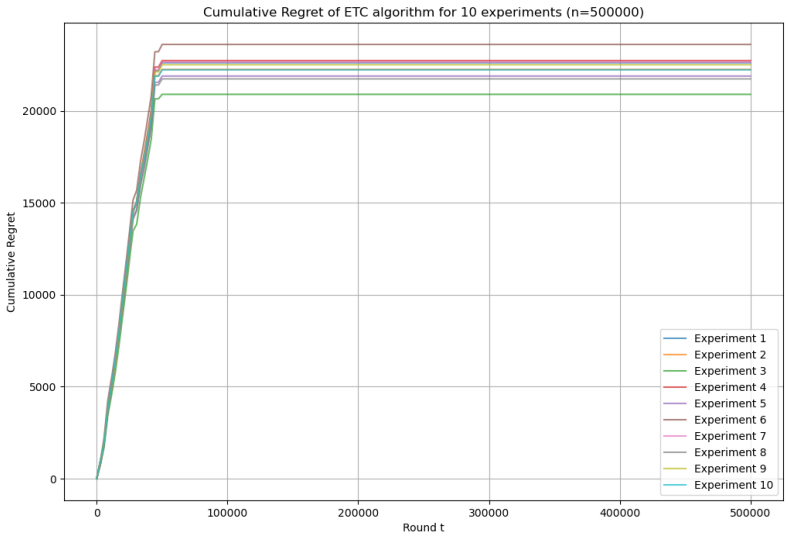


(b)

DOI: 10.54254/2755-2721/68/20241405



(c)



(d)

Figure 2. Cumulative Regret of ETC algorithm for 10 experiments (n=500000) (Photo/Picture credit: Original).

In this experiment, the horizon is defined as 500,5000,50000 and 500000. The m*k value equals to 10% of the horizon. In each experiment, the algorithm is implemented 10 times. From the result, the performance is not improved as the horizon increases, because the regret cumulated per round does not change much in different horizons. As a matter of course, the larger horizon contains more rounds to cumulative regret, so the final cumulative regret is more. While still the algorithm has not reached its regret bound and it cannot prove how the horizon influences the regret bound and the performance when the exploration is not sufficient.

2.3. Upper Confidence Bound Algorithm

The Upper Confidence Bound (UCB) Algorithm is a strategy that can solve MAB problem as well. Unlike ETC algorithm, UCB will not select only one arm in the exploitation phase but gives chance to

DOI: 10.54254/2755-2721/68/20241405

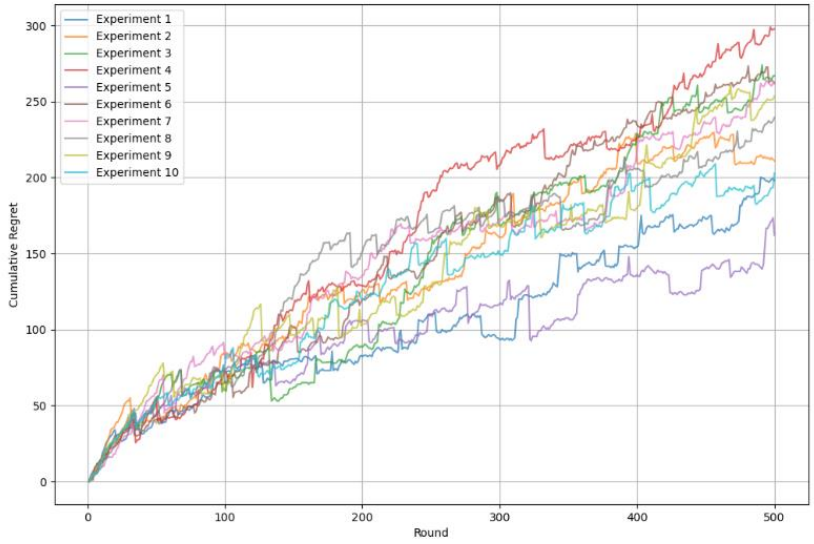
different arms to explore them further. It is regarded as exploratory adjustments to the empirical mean. Such adjustment is important because the rewards are basically noisy. Using previous data to estimate an arm's worth always creates noise. Larger historical data sets can reduce estimation noise and enhance confidence intervals. In each round, the arm with highest UCB value is selected. The UCB value of arm i is defined as $\mathrm{UCB_i(t)} = \hat{\mu}_\mathrm{i}(\mathrm{t}) + \sqrt{\frac{\alpha \log\ \mathrm{n}}{\mathrm{T_i(t)}}}$. Here n is the horizon. $\hat{\mu}_\mathrm{i}(\mathrm{t})$ calculates the average reward of arm i until round t. $\mathrm{T_i(t)}$ calculates the number of times arm i is selected until round t. $\alpha > 0$ .The first part of the formula is simply the current average reward. The second part determines the size of the one-sided confidence interval for the average reward, within which the genuine expected reward falls with extra probability. It is the extra probability that produces a dynamic selection. As shown in Table 2.

Table 2. UCB.

Algorithm 2 UCB:
for t = 1 to k **do**
Play arm I_t = t
end for
for t = k+1 to n **do**
For each arm i = 1 to k, calculate $UCB_i(t) = \hat{\mu}_i(t) + \sqrt{\frac{\alpha \log\ n}{T_i(t)}}$
Play arm I_t = $\arg_i \max\ UCB_i(t)$
end for

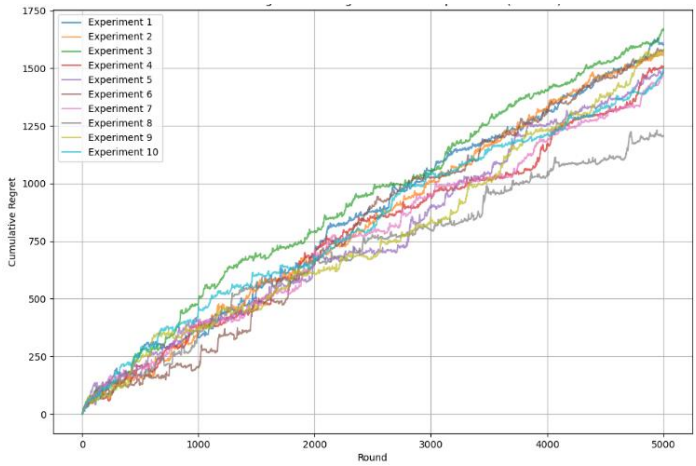
From the UCB formula, it can be noticed that the horizon n determines the confidence interval, and according to some prove it will achieve a logarithmic regret uniformly over n. Meanwhile, the larger is the α the more will the algorithm explore, while with smaller it will more aggressively exploit. Furtherly, the UCB value can redefined as $UCB_i(t) = \hat{\mu}_i(t) + \sqrt{\frac{\alpha \log\ f(t)}{T_i(t)}}$. The advantage of the asymptotically optimal UCB is that for the un-selected arm, the UCB value can increase, leading the un-selected arm have more probability to be selected.

In experiment 1, the Movielens 1M dataset is used to explore the performance of UCB algorithm in different horizon as 500, 5000, 50000, and 500000. Considering that the reward in each round range from 1 to 5, α is defined as 8. As shown in Figure 3.

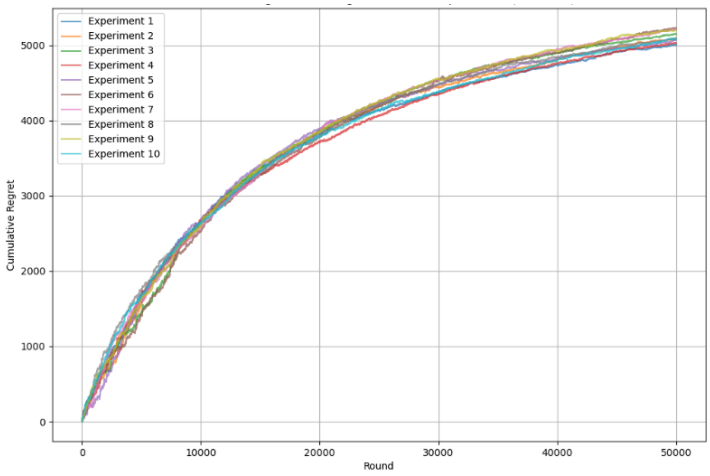


(a)

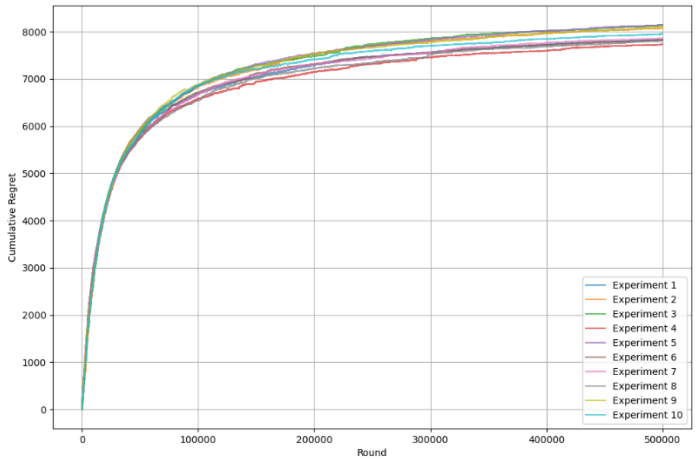
DOI: 10.54254/2755-2721/68/20241405



(b)



(c)



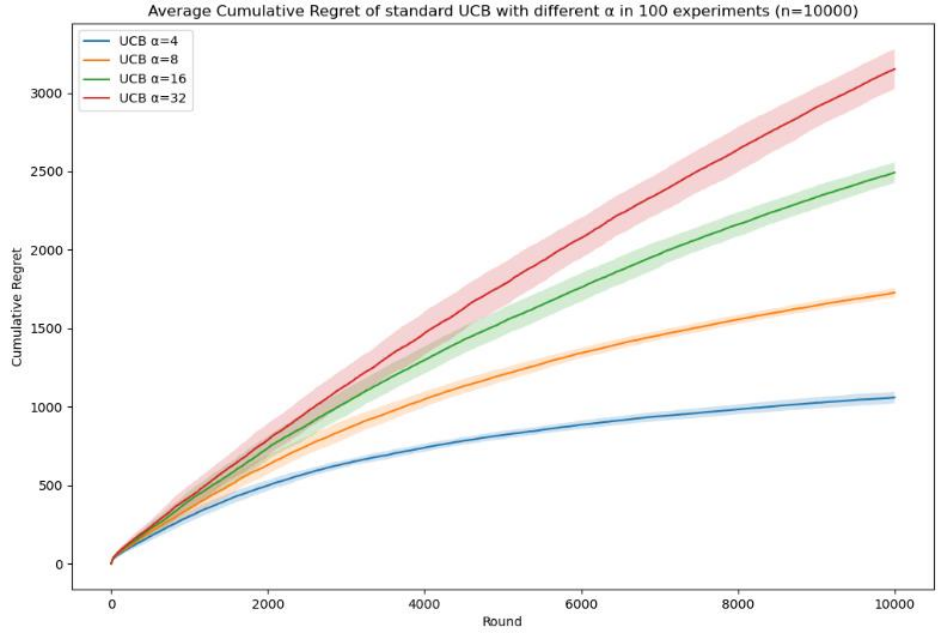
(d)

Figure 3. Cumulative regret of UCB algorithm for 10 experiments(n=500) (Photo/Picture credit: Original).

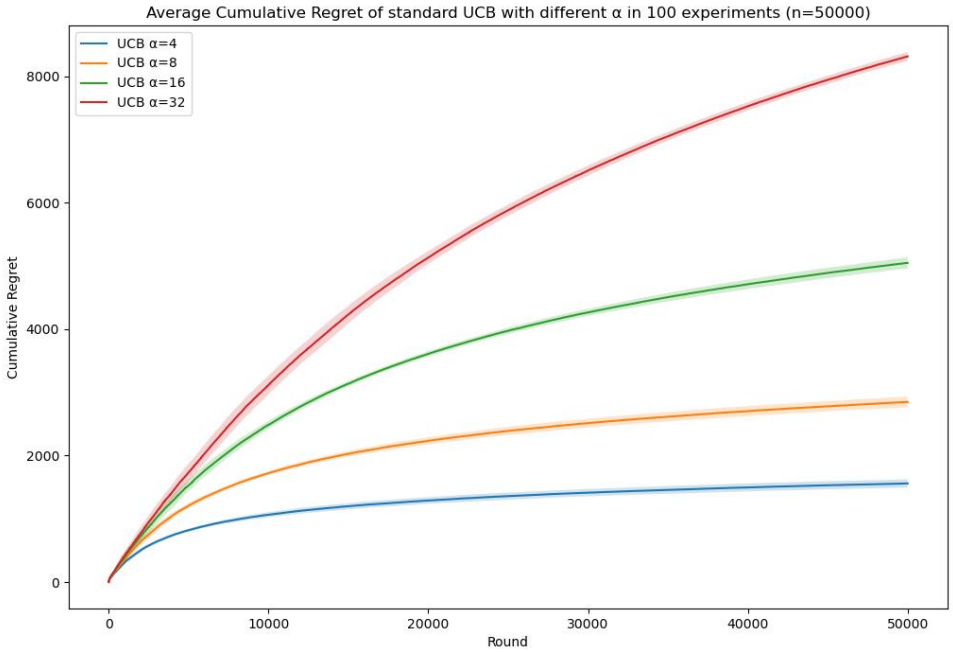
DOI: 10.54254/2755-2721/68/20241405

When n is larger than 50000, the growth rate of regret gets slower and slower as the number of rounds goes to a certain degree. This feature conforms to its theoretically logarithmic regret. Anytime algorithms provide a trade-off between execution time and solution quality. While UCB is an anytime algorithm, as the time increases the algorithm performs better. When n is small, it can be clearly seen that the growth of regret is more unordered, compared with ETC. This represents the strategy of selecting is smarter but not sequential.

In experiment 2, α is defined as 4, 8, 16, and 32. The aim is to explore the effect of α in different horizon, whether it has the same feature. The horizon is defined as 10000, 50000, 100000. In each horizon, the algorithm is implemented 100 times with error bar plotted. As shown in Figure 4.

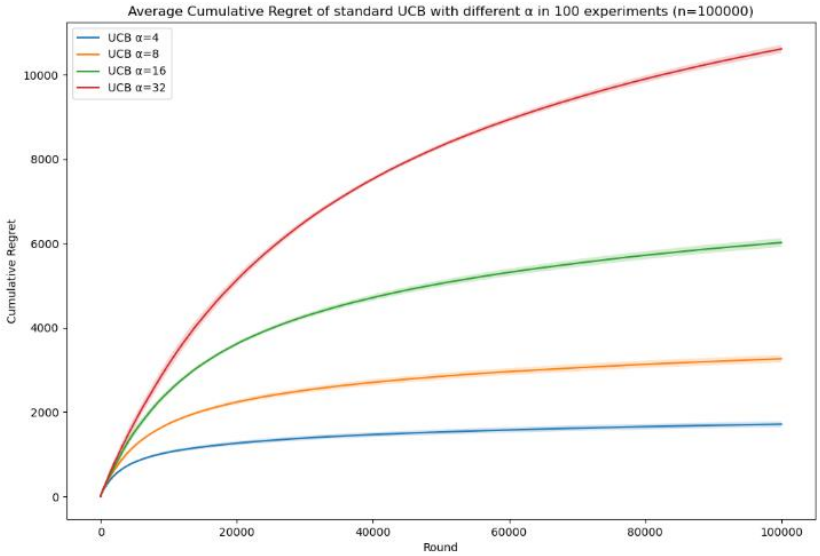


(a)



(b)

DOI: 10.54254/2755-2721/68/20241405

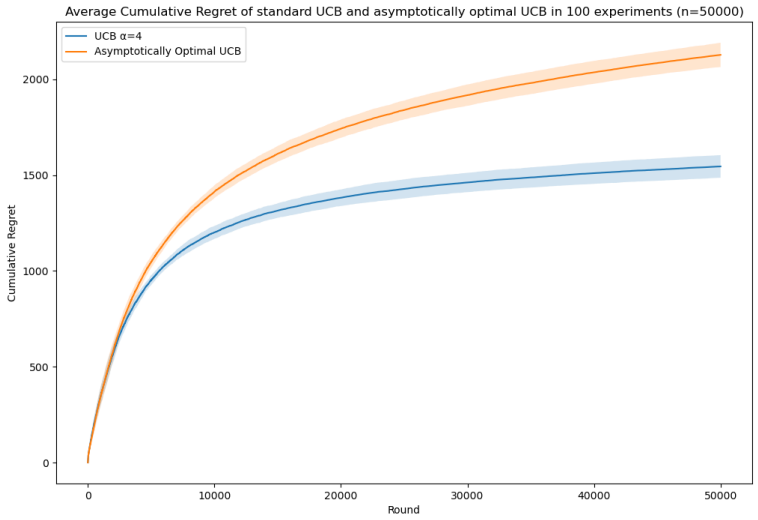


(c)

Figure 4. Average Cumulative Regret of standard UCB with different α in 100 experiments (n=100000) (Photo/Picture credit: Original).

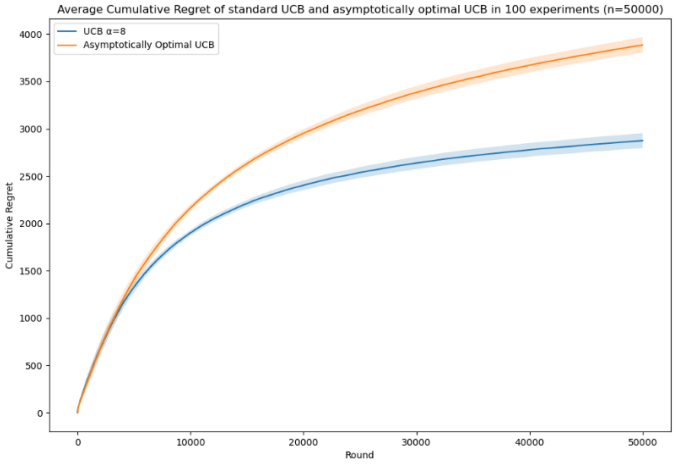
In all the horizons, it remains the same that the larger α is, the more regret will cumulative. According to the formula of UCB value, it makes sense that the upper confidence bound will be higher if α gets larger, leading those 'bad' arms with lower sampled mean reward have more chances to be selected. Though longer exploration cumulates more regret, it brings advantage that the upper confidence is closer to its true mean value. Given that one arm has low reward distribution, but the sampled mean value is luckily high because of the insufficient exploration, it will probably cumulative more regret compared with the case that it is well explored. Therefore, the agent must balance the value of α in practical terms.

In experiment 3, it remains to explore the performance of asymptotically optimal UCB. The function t is defined as $f(t) = 1 + t(\log t)^2$. The α remains the same in each experiment as 4, 8, 16 and 32. The n equals to 50000. At each α, both the standard UCB and asymptotically optimal UCB are implemented 100 times with error bar plotted. As shown in Figure 5.

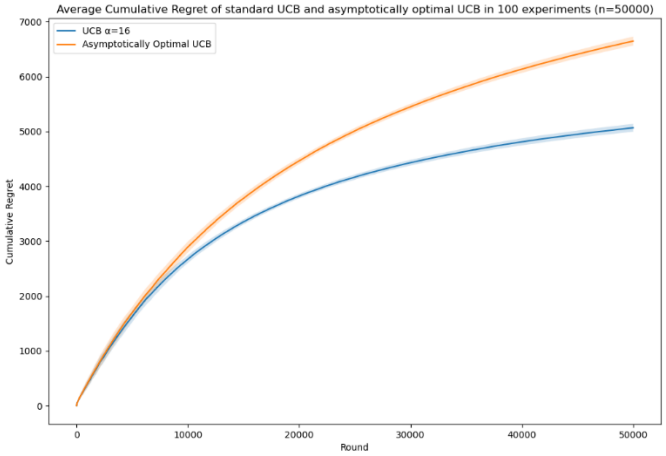


(a)

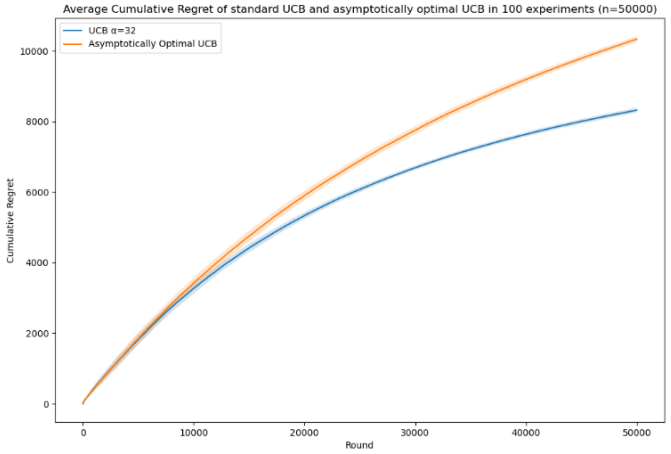
DOI: 10.54254/2755-2721/68/20241405



(b)



(c)



(d)

Figure 5. Average Cumulative Regret of standard UcB and asymptotically optimal UCB in 100 experiments (n=50000) (Photo/Picture credit: Original).

DOI: 10.54254/2755-2721/68/20241405

In all these experiments, asymptotically optimal UCB performs worse than standard UCB, which is not unexpected because the function t is a monotone increasing function. The function can change the UCB value of unselected arms as well as the selected arm in each round, making them have more probability to be explored, though more regret will cumulate. Like the conclusion in experiment 2, more comprehensive exploration provides some advantages, which is more obvious when the reward from each arm is close to each other.

2.4. Thompson Sampling Algorithm

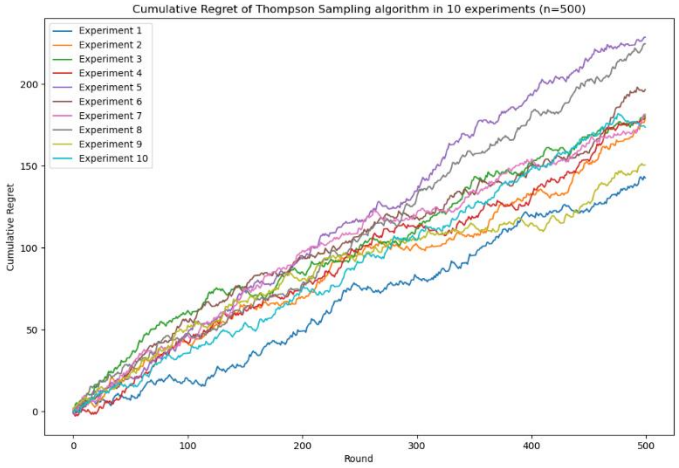
Thompson sampling algorithm is a method based on Bayesian inference, first introduced by William R. Thompson in 1933. TS-type algorithms are gaining popularity for MAB problems due to their ease of implementation and improved efficiency. Unlike UCB-type algorithms, which need compute upper confidence bounds for unknown mean reward of arms to determine the arm index, TS-type algorithms do the selection according to the posterior distribution, and continuously update the distribution. At the beginning, the Ts sampling algorithm assumes that every arm i has the prior $F_i(t)$ for the mean rewards (initially round t =1). In each round t, sample $\theta_i(t)$ from distribution $F_i(t-1)$ independently for each arm i, and choose arm $I_t = \arg\max_i \theta_i(t)$ to get the reward X_t. Then update the distribution of the selected arm. For the unselected arms, $F_i(t) = F_i(t+1)$. In most implementations of the algorithm, we use Gaussian update, which means that the $F_i(t) \sim \mathcal{N}_i\left(\hat{\mu}_i(t), \frac{\sigma^2}{T_i(t)}\right)$ if the reward distributions are σ-sub-Gaussian, $\hat{\mu}_i(t)$ denotes the mean reward from arm i until round t, and $T_i(t)$ denotes the number of times arm i is selected until round t.

Table 3. Thompson sampling algorithm.

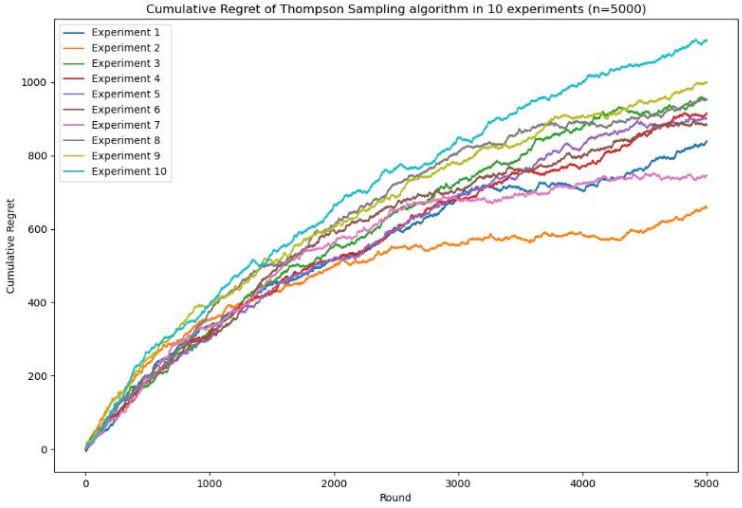
Algorithm 3 Thompson sampling algorithm:
for t = 1 to k **do**
Play arm I_t = t
end for
for t = k+1 to n **do**
For each arm i = 1 to k, sample $\theta_i(t)$ from $\mathcal{N}_i\left(\hat{\mu}_i(t-1), \frac{\sigma^2}{T_i(t-1)}\right)$ distribution.
Play arm $I_t = \arg\max_i \theta_i(t)$ and observe reward X_t.
Update the distribution of arm I_t with $\mathcal{N}_i\left(\hat{\mu}_i(t), \frac{\sigma^2}{T_i(t)}\right)$.
end for

Previous work has proved that Ts sampling algorithm is an anytime algorithm, and ultimately show a logarithmic regret. In the experiment, the Movielens 1M dataset is still used to explore the performance in different horizons as 500, 5000, 50000, 500000 and 5000000. For the ratings ranges from 1 to 5, σ is defined as 2.5 (the mean rating in theory). As shown in Figure 6.

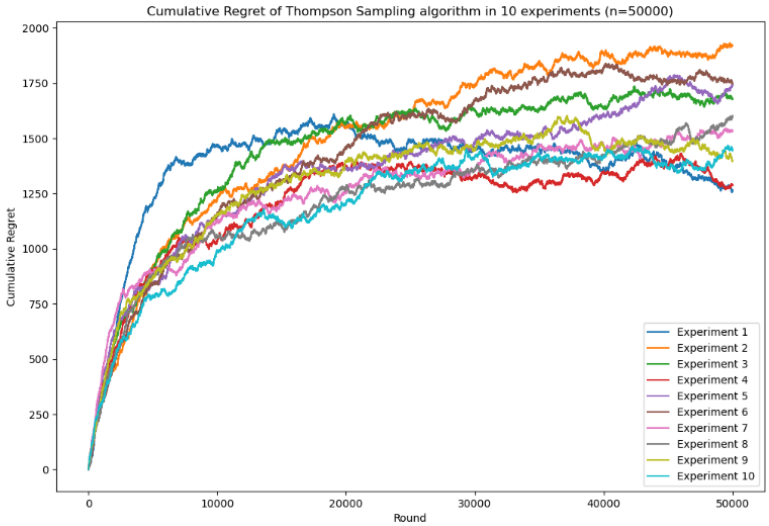
DOI: 10.54254/2755-2721/68/20241405



(a)

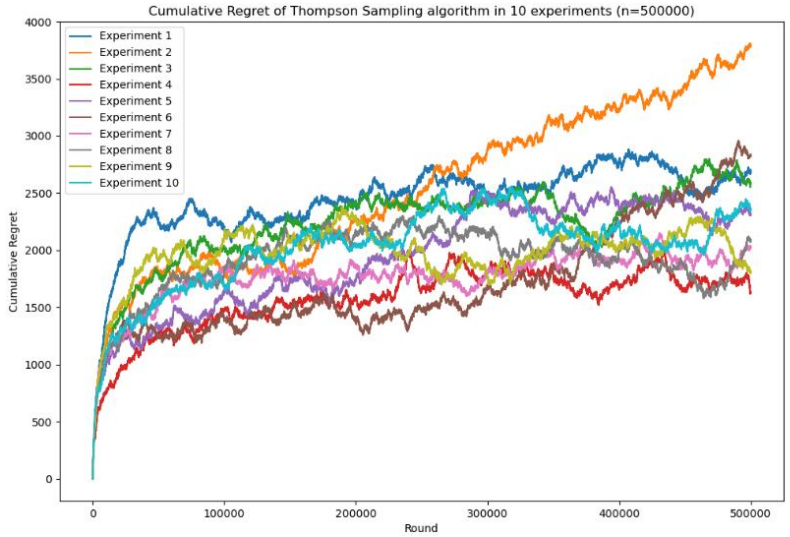


(b)

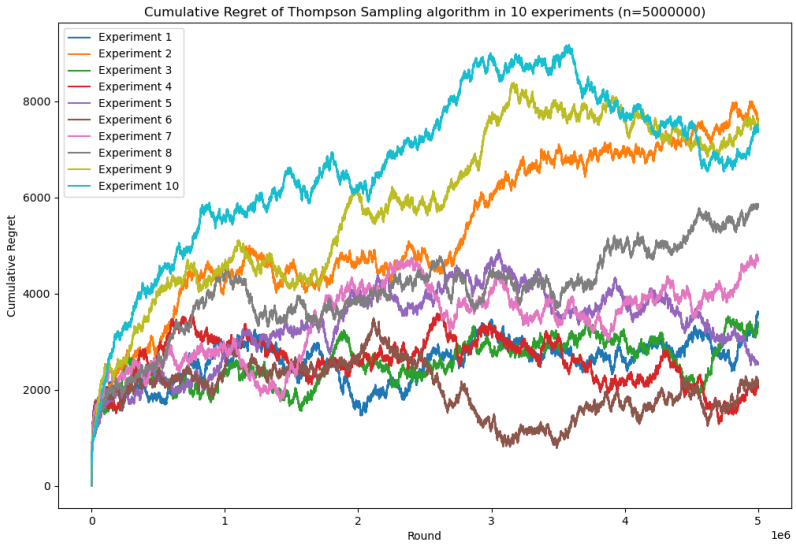


(c)

DOI: 10.54254/2755-2721/68/20241405



(d)

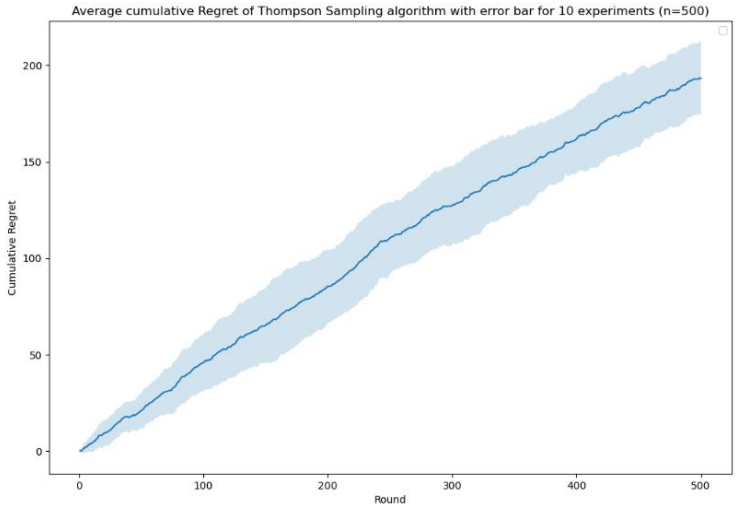


(e)

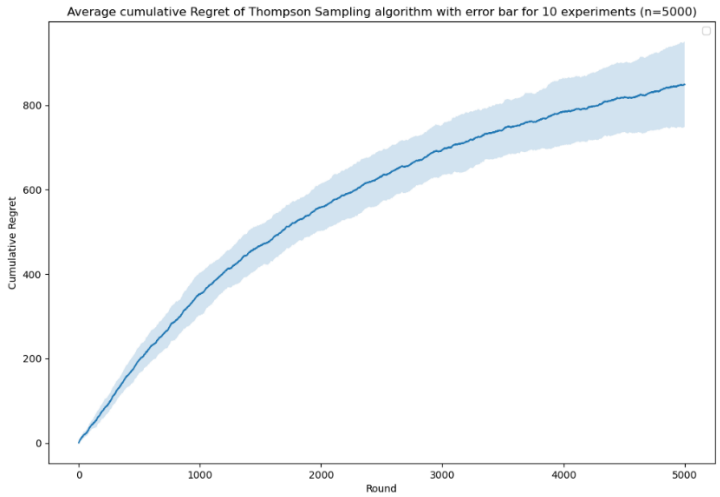
Figure 6. Cumulative Regret of Thompson Sampling algorithm in 10 experiments (n=5000) (Photo/Picture credit: Original).

As is portrayed in the results, the regret has a significant fluctuation in each experiment. In addition, it varies differently from each other even when n is large. Compared with ETC and UCB algorithms, Ts sampling selects the arm more randomly, while it performs well for the regret declining in some stages. However, it brings disadvantage that in practical implementation the result may be unsatisfactory because the agent cannot play many times considering the cost. Moreover, it is hard to conclude that the expect regret reach a logarithmic growth in final. Therefore, the average regret with error bar is plotted below. As shown in Figure 7.

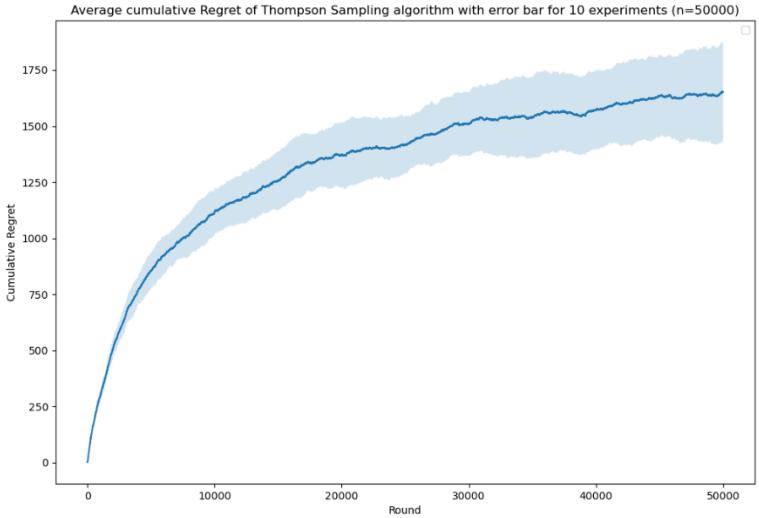
DOI: 10.54254/2755-2721/68/20241405



(a)

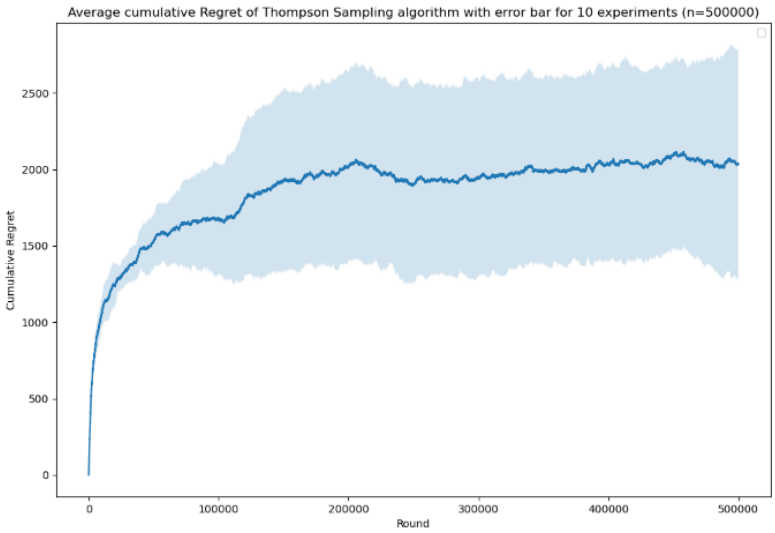


(b)

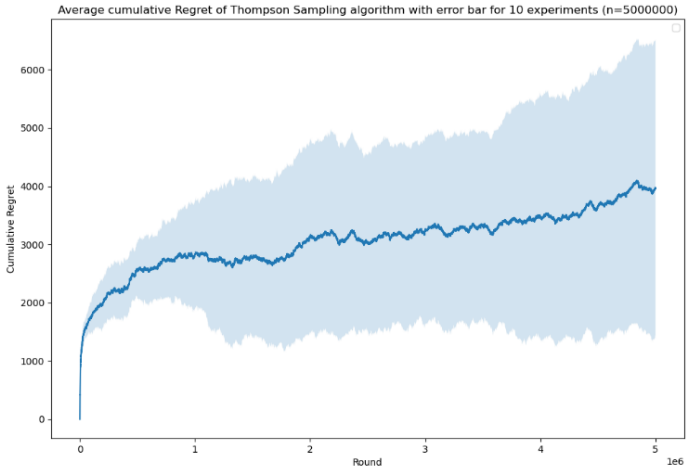


(c)

DOI: 10.54254/2755-2721/68/20241405



(d)



(e)

Figure 7. Average cumulative Regret of Thompson Sampling algorithm with error bar for 10 experiments (n=5000000) (Photo/Picture credit: Original).

The results show the logarithmic growth with the average regret. Interestingly, the error bar is much wider than those of ETC and UCB and vibrates more fiercely with the round goes. This indicates that every round selection is a randomization and make a big difference in a big scale. It is the randomization that enables the algorithm to perform better than ETC and UCB, while it has not been well explained the reason yet.

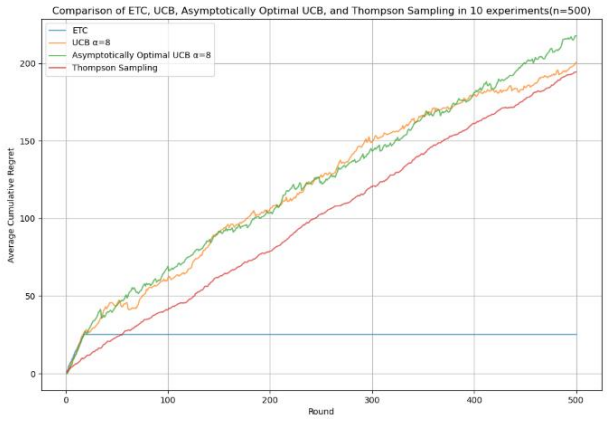
3. Comparative Performance Analysis

3.1. Performance Across Different Horizons

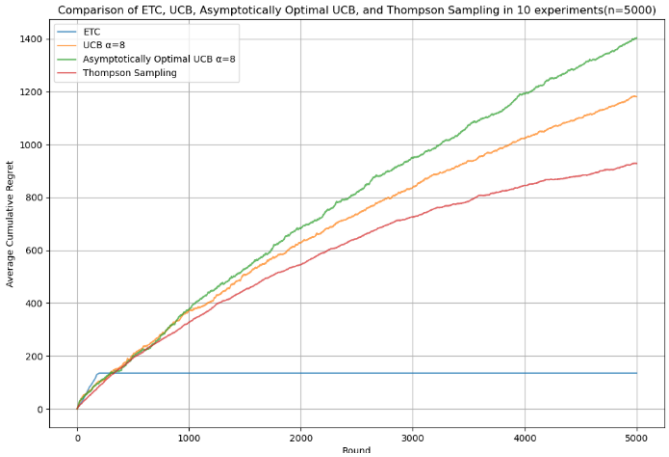
In this section, experiments aim to make an analysis of comparative performance of the four algorithms in different horizons (n=500, 5000, 50000, 500000, and 5000000), including the asymptotically optimal UCB. For ETC algorithm, m*k value is set as 10% of the horizon. In standard UCB algorithm, α is defined as 8, as well as the asymptotically optimal UCB, while the function t in asymptotically optimal UCB is defined as $f(t) = 1 + t(\log t)^2$. In Thompson sampling algorithm, it is believed that the

DOI: 10.54254/2755-2721/68/20241405

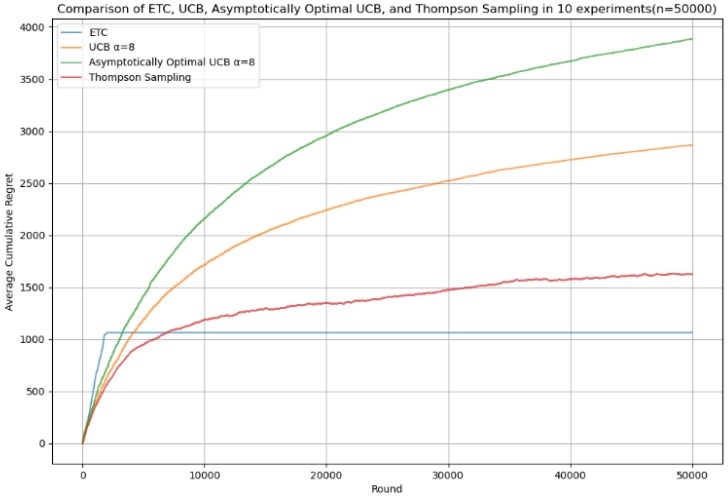
posterior distribution of the reward $F_i(t) \sim \mathcal{N}_i\left(\hat{\mu}_i(t), \frac{\sigma^2}{T_i(t)}\right)$. For each horizon, every algorithm is implemented 10 times to average the cumulative reward. The Movielens 1M dataset is used to stimulate the multi-armed-bandits problem. As shown in Figure 8.



(a)

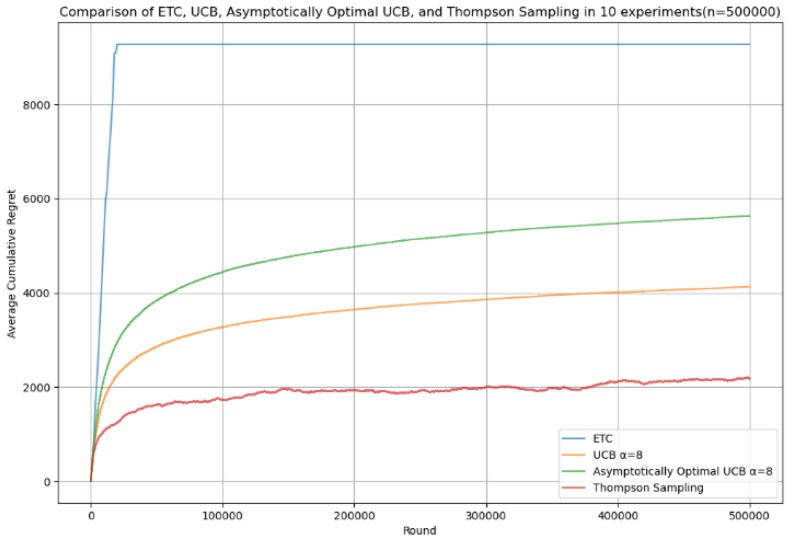


(b)

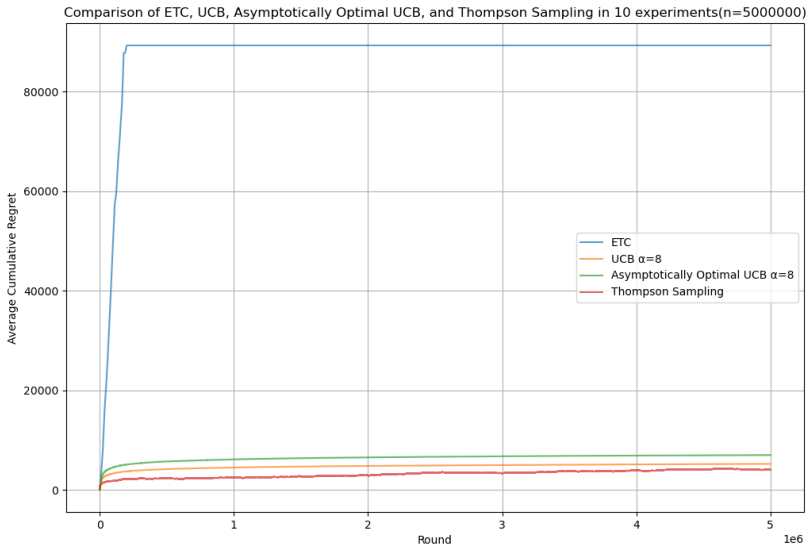


(c)

DOI: 10.54254/2755-2721/68/20241405



(d)



(e)

Figure 8. Comparison of ETC, UCB, Asymptotically Optimal UcB, and Thompson Sampling in 10 experiments(n=50000) (Photo/Picture credit: Original).

In the case that n is small (500, 5000, and 50000), ETC algorithm performs much better than UCB and Ts sampling in the global view. While concentrating on the exploration phase, the growth rate of ETC is larger than the other, which means that in each round it cumulates more regret. It is the shorter exploration phase that makes the regret lowest (precisely, there is no clear boundary between exploration and exploitation in UCB and Ts sampling). Then Ts sampling is better than two UCBs in all rounds. In previous conclusion, it is said that Ts sampling performs better than UCB in most cases, which is consistent with the results of the experiments. Compare the two UCB algorithms, in all the experiments standard UCB has less regret than asymptotically optimal UCB, but the more sufficient exploration in asymptotically optimal UCB make the mean reward of each arm more precise and can distinguish the best arm with more reliability.

DOI: 10.54254/2755-2721/68/20241405

When n gets larger enough (500000 and 5000000), the effectiveness of TS algorithm continuously increases, ranks the first in all rounds. Then two UCB algorithms have more regret than Ts, samely, standard UCB is better. They three all show a logarithmic regret behavior. However, the regret of ETC algorithm surges as the horizon increases, much more than the remaining, which show a linear growth in total.

Therefore, when the horizon is small to a certain extent, ETC algorithm is the optimal option. Nevertheless, in the era of highly developed electronic technology, the arms can be played by billions of net citizens, e.g., in the recommender system for advertising. Considering the lower regret and easier implementation, Ts sampling algorithm may come to the first place. Limitations exist that the parameters are set as constants, and the performance may change in some different settings (particularly in UCB and Ts sampling). In addition, universality of the conclusion is not guaranteed because environment remains the same (only one dataset is used).

3.2. Performance on Different Datasets

In this section, the three algorithms are implemented on three different datasets. Different datasets mainly vary the number of arms and the distribution of each arm's reward. The three used datasets are: Streaming Application Reviews dataset, Amazon Musical Instruments Reviews dataset and Beer Reviews dataset. The number of arms in each dataset are 100, 900 and 3844. In addition, the standard deviation of each arm's average reward is calculated to show the difference among all the arms, 0.19, 0.47 and 0.54 in order. The reward in each round rank from 1 to 5. The horizon is set as 10000000. All the other settings are the same as those in section 3.1. Each experiment is implemented 10 times to calculate the average cumulative reward. As shown in Figure 9-10

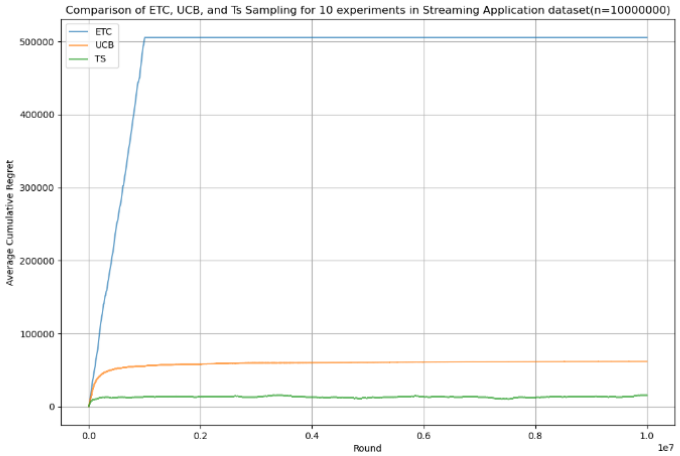


Figure 9. Comparison of ETC, UCB, and Ts Sampling for 10 experiments in Streaming Application dataset(n=10000000) (Photo/Picture credit: Original).

DOI: 10.54254/2755-2721/68/20241405

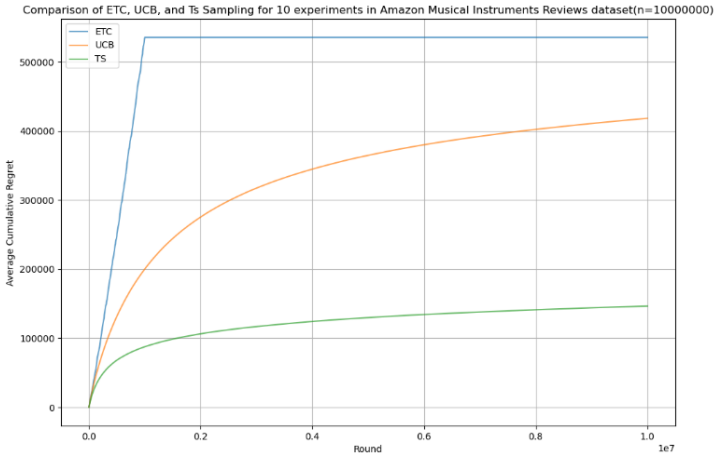


Figure 10. Comparison of ETc, UcB, and Ts Sampling for 10 experiments in Amazon Musical instruments Reviews dataset(n=10000000) (Photo/Picture credit: Original).

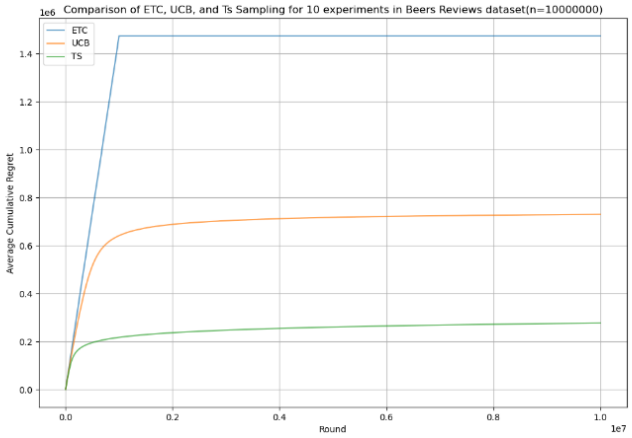


Figure 11. Comparison of ETc, UCB, and Ts Sampling for 10 experiments in Beers Reviews dataset(n=10000000) (Photo/Picture credit: Original).

In all the experiments, Ts sampling has the lowest regret, then UCB ranks the second, and ETC gets the largest regret, which is the same as section 3.1. Within the same horizon, there is no doubt that the regret is higher when the number of arms and the gaps among each arm are larger, making the algorithm harder to select the optimal arm. In experiment 1 (with lowest standard deviation), UCB and Ts perform far better than ETC, because the cost of 'randomly' exploring will decrease with arms that have similar reward. In experiment 2 and 3, the gap between ETC and UCB is not such big as experiment 1. The large number of arms make UCB selecting arms more difficult.

Therefore, ETC algorithm has good stability for different environments, though it has the largest regret, while in some cases the gap is not much big. UCB algorithm is sensitive to the gaps among each arm and number of arms, but can perform extremely well when the gap is small compared to ETC. Ts sampling is the best in all cases, with lowest regret and least sensitivity to the environment, while it costs more space and time computing in practical implementations.

4. Challenges

Though multi-armed-bandit problems are investigated for tens of years, there are some challenges remaining that the environment can greatly impact the difficulty of solving the problem, particularly when the environment is non-static. For ETC algorithm, it may make decisions too early and miss out on better options and cannot adapt to the dynamic environments. Similarly, UCB is also sensitive to

DOI: 10.54254/2755-2721/68/20241405

changes in the environment, leading to unreliability to the upper confidence bound. In Ts sampling algorithm, more sampling may be required to get an accurate posterior distribution, resulting in increased computational costs. For high-dimensional action spaces, it may face the challenge of computational complexity. In theory, the reason of the high efficiency of Ts sampling is not well known yet. Although these algorithms perform well in solving bandit problems, they still have some problems that need to be further studied and solved to improve the performance and applicability of the algorithms.

5. Conclusion

This study utilizes four datasets to simulate multi-armed bandit scenarios, assessing the performance of various algorithms in differing contexts and identifying optimal settings with theoretical justifications. A comparative analysis of three algorithms is conducted, revealing that the Explore-Then-Commit (ETC) algorithm is effective in scenarios with a smaller horizon, but its efficacy decreases significantly as the horizon expands. The performance gap between the Upper Confidence Bound (UCB) and Thompson Sampling (TS) is minimal, with TS demonstrating a slight edge. Furthermore, when applied to dynamic environments, both UCB and TS display robust generalization capabilities. Practically, TS is favored for its high efficiency and ease of implementation. In summary, the bandit framework stands as a vibrant and promising area of research, brimming with opportunities for further exploration. This paper aims to provide readers with a foundational understanding of the three algorithms and offer insights into approaching multi-armed bandit problems effectively. Looking ahead, there is a pressing need for further research in dynamic, or non-static, settings. Extending existing algorithms to these environments poses a significant challenge. Moreover, even within static contexts, the development and exploration of more advanced derivatives of these algorithms remain crucial areas for future investigation.

References

[1] Bouneffouf, D., & Rish, I. (2019). A survey on practical applications of multi-armed and contextual bandits. preprint:1904.10040.

[2] Agrawal, S. (2019). Recent advances in multiarmed bandits for sequential decision making. Operations Research & Management Science in the Age of Analytics, 167-188.

[3] Garivier, A., Lattimore, T., & Kaufmann, E. (2016). On explore-then-commit strategies. Advances in Neural Information Processing Systems, 29.

[4] Auer, P., Cesa-Bianchi, N., & Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. Machine learning, 47, 235-256.

[5] Zhu, X., Xu, H., Zhao, Z., & others. (2021). An Environmental Intrusion Detection Technology Based on WiFi. Wireless Personal Communications, 119(2), 1425-1436.

[6] Nguyen-Thanh, N., Marinca, D., Khawam, K., Rohde, D., Vasile, F., Lohan, E. S., ... & Quadri, D. (2019). Recommendation system-based upper confidence bound for online advertising. preprint :1909.04190.

[7] Jesus, A. D., Liefooghe, A., Derbel, B., & Paquete, L. (2020, June). Algorithm selection of anytime algorithms. In Proceedings of the 2020 genetic and evolutionary computation conference (pp. 850-858).

[8] Kong, F., Yang, Y., Chen, W., & Li, S. (2021). The hardness analysis of thompson sampling for combinatorial semi-bandits with greedy oracle. Advances in Neural Information Processing Systems, 34, 26701-26713.

[9] Kong, F., Yin, J., & Li, S. (2022). Thompson sampling for bandit learning in matching markets. preprint:2204.12048.

[10] Baudry, D. (2022). Non-parametric algorithms for multi-armed bandits (Doctoral dissertation, Université de Lille).