Two-Sided Bandit

An empirical comparison of different learning methods for solving the two-sided bandit problem.

ZIJUN XU

zxu26@gmu.edu

This paper aims to empirically compare the performance of 3 algorithms for solving the two-sided bandit problem. Performance is measured with 1) the probability of getting a stable matching and 2) the time takes to convergence. The core of solving the two-sided bandit problem is to find a balance between exploration and exploitation. I simulated the two-sided bandit problems in both global or arbitrary ranking systems and discovered that Thompson turns out to have the best performance, while ε-Greedy and CA-UCB have their own pros and cons over each other.

1 INTRODUCTION

A well-known problem in the field of machine learning is the multi-armed bandit. In this problem, there is one player and multiple slot machines. Each slot machine has an arbitrary probability of paying off a $1 reward when pulled. In each turn, the player can choose to pull one of the slot machines – the player's goal is to maximize their total reward within a time duration by discovering the slot machine with the highest probability of paying off.

A more complex version of the multi-armed bandit is the two-sided bandit problem [Das, Kamenica, 2005]. Instead of a single player, there are multiple players playing simultaneously. Each player has preference over the arms and each arm also has preference over the players. The introduction of more players brings new dynamics to the table – when multiple players pull the same arm, conflict arises. Only the most preferred player wins the competition and gets rewarded. Furthermore, the winner's reward varies, depending on how much the player likes the arm he/she pulled. Intuitively, it is insufficient for each player to pull only their most preferred arm, as this often leads to conflicts, thus ending up with some players getting no rewards. Our goal in this case, is then to find a stable matching, such that it maximizes the cumulative reward of all players.

Ultimately, the core of the two-sided bandit problem is for each player to re-evaluate their initial preference list – sometimes the most preferred arm is not the best choice, since there could be a stronger competitor, who is more preferred by that arm. Being unmatched (no rewards) is strictly the worst situation and should be avoided. Solving the classic exploration-exploitation tradeoff is the key to a successful learning method for solving the two-sided bandit problem. Would the player stay with the safe choice (the arm he/she can constantly win) or try to pull some unknown arm with potential higher rewards, risking being unmatched?

In this paper, I aim to empirically compare the performance of three learning methods for solving the two-sided bandit problem. Performance is measured by the probability of convergence to stable matchings (reliability) and the time of convergence (efficiency). The methods will be tested on various settings: global ranking and arbitrary ranking. For example, global ranked players indicate all arms have the same preference on players. I will also discover whether the number of arms affects the performance of these methods.

2 THE MODEL

Consider a two-sided bandit game with N players and L arms ($N \leq L$), where $N = \{p_1, p_2, p_3, \ldots, p_N\}$, $L = \{a_1, a_2, a_3, \ldots, a_L\}$. Each player/arm has a strict preference ordering of the opposite side. For example, in the following preference ranking, p_1 prefers a_1 the most. a_2 prefers p_2 the most. We denote the rank of arm_j of $player_i$ as $ARank_{ij}$. Similarly, the rank of $player_i$ of arm_j as $PRank_{ji}$. The rank of a_1 to p_1, $ARank_{11} = 1$.

$$p_1\text{: } a_1 \succ a_2 \succ a_3$$

$$a_2\text{: } p_2 \succ p_3 \succ p_1$$

The game runs at a fixed T time step. At each time step, each player p_i attempts to pull one of the arms a_j. At the end of each time step, each arm has a candidate list containing all the players that have attempted to pull it. If an arm has an empty candidate list, we say the arm is unmatched for this round. For arms with a non-empty candidate list, the winner is chosen based on the arm's preference list (the most preferred player wins). We say the winning player and the arm is matched for this round and the losing players are unmatched. The winner's reward, is decided by the rank of the pulled arm in his/her preference list:

$$\text{Reward} = 1 / \text{ARank}_{ij}$$

Simply speaking, a player gets a reward of 1 if the arm pulled is his/her most preferred arm, 1/2 if the arm is the second preferred, 1/3 if the arm is the third preferred, etc.

Next, we formally define the concepts of global ranking and arbitrary ranking. The players/arms have individual setting of global/arbitrary ranking, which implies we have four possible combinations:

1) Global ranking on both players/arms
2) Global ranking on players, arbitrary ranking on arms
3) Arbitrary ranking on players, global ranking on arms.
4) Arbitrary ranking on both players/arms.

Finally, we introduce an algorithm called Gale-Shapley [Gale and Shapley, 1962], which is guaranteed to find asymptotically stable matchings for the stable marriage problem. We use output of the man-optimal version (in our case, player-optimal) as the standard to check the performance of the learning methods to be tested.

3 THE LEARNING METHODS

We introduce 3 learning methods for solving the two-sided bandit problem, along with the parameter settings for each algorithm.

3.1 E-Greedy

At each time step, player_i attempt to pull a random arm with probability ε, otherwise pull arm_j with the highest expected utility u_j. which is defined as:

$$u_j = \frac{\#\ of\ successful\ pulls\ on\ arm\ j}{\#\ of\ pulls\ on\ arm\ j} * \text{Reward of arm}_j$$

In other words, the expected utility of arm_j for a particular player is the probability that this player can successfully pull the arm, based on the evidence so far, multiplying the reward this arm would give this player if successfully pulled. It is worth noting that ε-Greedy is initialized by every player pulling all the arms at the first round.

It is essential that ε must decline as the iteration proceeds, otherwise it will be difficult to judge convergence since the agents could choose random actions with probability ε at any time step. Therefore, we define the declining ε schedule:

$$\varepsilon = \varepsilon^{t/100} \text{ where t is the current time}$$

The initial value of ε also has a great impact on the performance of ε-Greedy. It was reported that the probability of convergence to stable matching increases as ε increases in the range [0.1, 0.9] [Das, Kamenica, 2005]. Thus, we choose ε = 0.9 for the experiments. With the above decline ε schedule, ε is decreased to 0.001 at iteration 6500.

3.2 CA-UCB [Liu, Ruan, Mania, Jordan, 2020]

At each time step, player_i attempt to pull the same arm as the last round with probability λ, otherwise pull arm_j from the plausible arm list with the highest ucb (upper confidence bound) value. An arm is considered plausible if player_i was able to successfully pull it in the last round. In other words, an arm is not plausible if it was pulled by some other player_j in the last round, who is ranked higher than player_i by this arm. Clearly, the CA-UCB algorithm requires information sharing among players: for each arm, the player successfully pulled it in the last round is known to all players.

Additionally, every player knows that whether they are ranked higher or not with respect to the last round's winner of each arm, by this arm.

It was reported that CA-UCB guarantees stable regret for each player when the players are globally ranked. The value of λ is recommended to be in the range (0, 0.25) [Liu, Ruan, Mania, Jordan, 2020]. Therefore, we choose λ = 0.2 for all cases.

3.3 Thompson [Thompson, 1933]

At each time step, $player_i$ sample a random beta distribution of (a, b) of each arm_j, where a is the count of $player_i$ successfully pulled arm_j and b is the count of failed pull. Then the arm with the highest utility is chosen:

$$u_j = \text{beta}(a, b) * \text{Reward of } arm_j$$

Unlike ε-Greedy and CA-UCB where a random variable is involved in each iteration, Thompson's decision rule is only dependent on the expected utility.

4 RESULTS

For each scenario of global/arbitrary players/arms mentioned in section 2, 500 instances are generated. The initial preferences are randomly generated with respect to the scenario rule. We will also investigate the effect of the number of arms by comparing the results of 5 arbitrary players/arms vs 5 arbitrary players/10 arbitrary arms. Each instance rans 10,000 iterations. All results are averaged.

The left graph in figure 1 shows the reliability of each algorithm with 5 players/arms. Notice that Thompson performed exceptionally well, with a 1.00 probability to stable matching in all scenarios except a 0.99 probability under arbitrary both sides. E-Greedy comes second, with significant worse performance on the arbitrary both sides case compared to Thompson. Finally, CA-UCB kept up with other two algorithms only in the case of global ranked players/arbitrary ranked arms and is relatively less reliable in other cases. Overall, the graph indicates if both the players and arms are arbitrarily ranked, the probability of getting a stable matching is lower.

The right graph in figure 1 shows that the performance of all algorithms improves as we increase the number of arms from 5 to 10. It is worth noting that CA-UCB got the biggest bump in accuracy from 0.88 to 1.0 (rounded). Intuitively, as we increase the number of arms, players have less chance to prefer/choose the same arm, thus less conflicts and a higher probability of getting a stable matching.

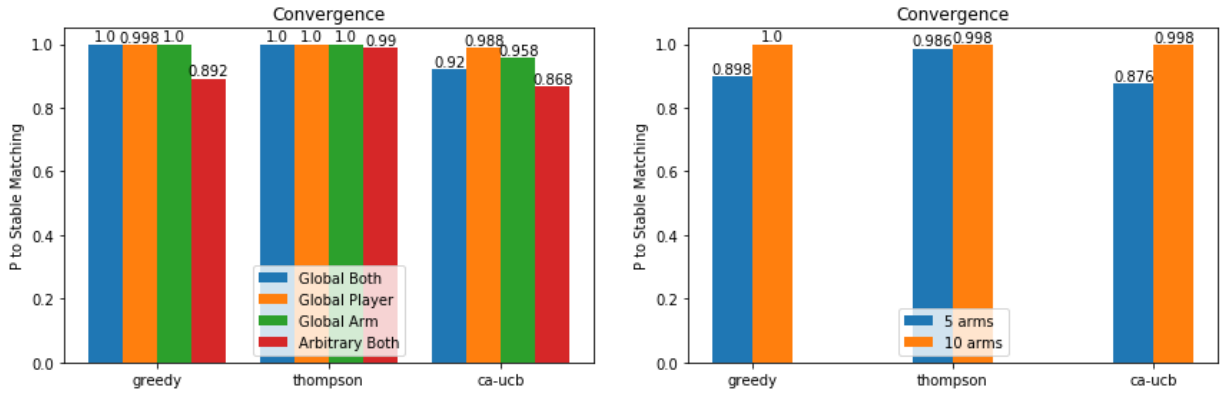

Figure 1: Probability of stable matching under different scenarios

Figure 2 shows the total rank of the assignments as a function of time. Either global or arbitrary rank, Thompson converges in the first few iterations. CA-UCB can generally find a solution close to the optimal matching fast but takes a while to settle down. Lastly, ε-Greedy converges around 5000 steps for global on both sides and converges slower as we introduce arbitrary ranks.

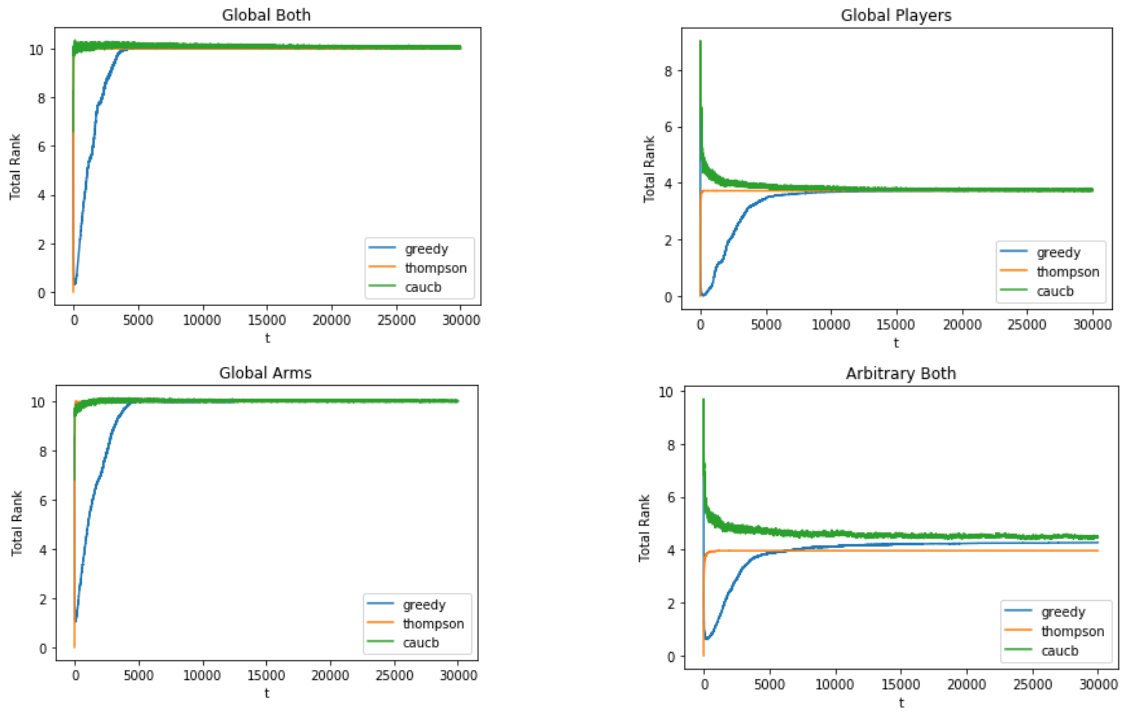

Figure 2: Total rank as a function of iteration t under different scenarios

Finally, figure 3 shows that as we increase the number of arms from 5 to 10, both ε-Greedy and CA-UCB converge faster to the optimal solution, while Thompson maintains a high efficiency in all cases.

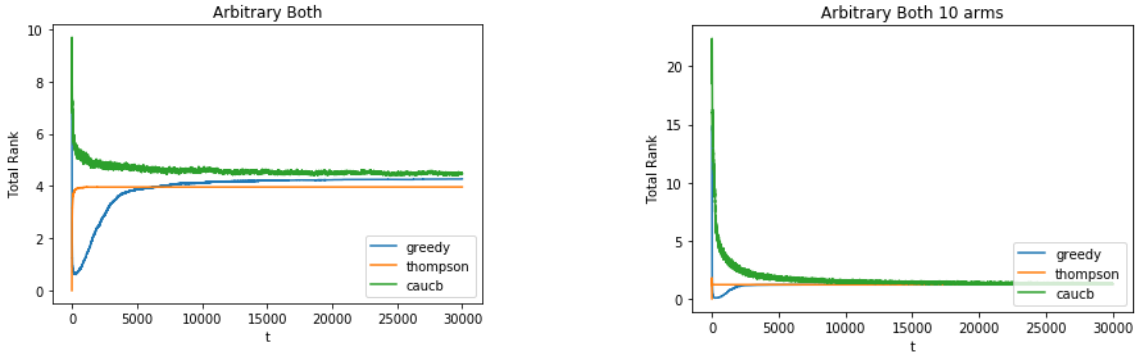

Figure 3: Total rank as a function of iteration t under different number of arms

5 CONCLUSIONS

Thompson performed exceptionally well in all cases, with a nearly 1.0 probability of getting a stable matching and converges to stable matching within a small number of iterations. ε-Greedy converges slower than CA-UCB but has a higher probability of getting a stable match, while CA-UCB converges to a sub-optimal matching more often. In conclusion, Thompson is my choice of the best algorithm for solving the two-sided bandit problem thanks to its excellent performance and simple structure.

This paper only focused on the player-optimal solution – the performance of these algorithms is unclear in terms of optimizing the rank on both sides. Additionally, we also restricted that players and arms cannot have indifference rank over the opposite side. Indifference rank is often presented in problems such as the marriage market. Both limitations lead to great research topics for my future work.

REFERENCES

[1] [Gale and Shapley, 1962] D. Gale and L. S. Shapley. College admissions and the stability of marriage. The American Mathematical Monthly, 69(1):9–15, 1962.

[2] [Das, Kamenica, 2005] S. Das and E. Kamenica. Two-sided bandits and the dating market. https://www.ijcai.org/Proceedings/05/Papers/0334.pdf

[3] [Liu, Ruan, Mania, Jordan, 2020] L. T. Liu, F. Ruan, H. Mania and M. I. Jordan. Bandit learning in decentralized matching markets. https://arxiv.org/pdf/2012.07348.pdf

[4] [Thompson, 1933] W. R. Thompson On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. Biometrika, 25(3–4):285–294

PRESENTATION LINK

https://youtu.be/Ke6ItN245rU