CS7641 Assignment 4
Markov Decision Processes

Introduction

The two MDPs I picked are Rock Paper Scissors and Stock trading. Although both of them are well-known and common MDPs I found them particularly interesting as they allow me to truly compare Value iteration Policy Iteration and PPO reinforcement learning.

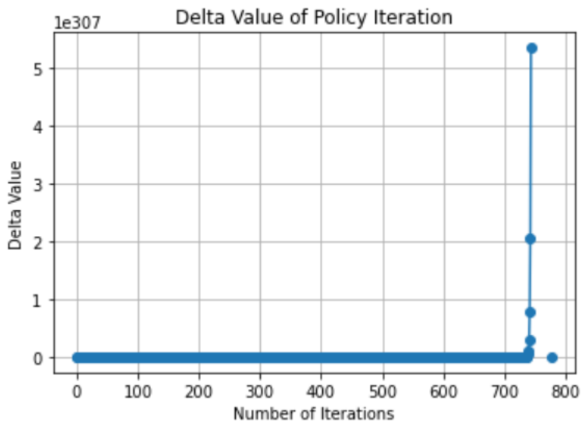
For the Rock-Paper-Scissors Problem, the objective is to find the optimal strategy to play against an opponent who picks moves at random. The goal is to maximize the chances of winning or at least achieving a draw. As MDP, the states include Rock, Paper, and Scissors. The rewards are +1, -1, and 0. The transitions depend on the opponent's move.

For the stock market Problem, the objective is to find the best strategy to get the most profit from stock trading based on weather conditions and stock prices. Essentially the problem is to find the optimal trading strategy that maximizes profits or minimizes losses To scale this to a large problem, I used 10 weather conditions and 100 stock prices. Therefore, there are 10 * 100 = 1000 states. The 3 actions defined for this problem are buy, sell, and hold. The reward for this problem is the increase in value from the previous. If the value increases the reward is +1, if decreases it's -1, and 0 for no change.

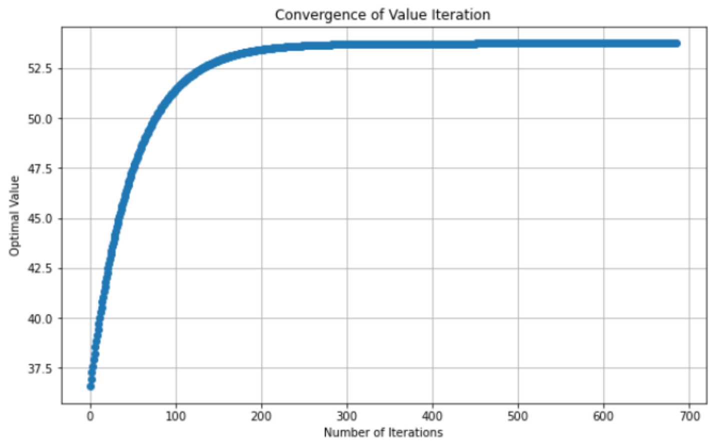
I find both of these problems ideal for this research because they are based on real-world ideas. I think that proved some real meaning to the results computed. Additionally, the two problems are so different in terms of information involved that it forces the experiments to explore the performance of this algorithm under a different variety of problems. While one problem is a game-type problem where the goal is to find an ideal strategy, the other focuses on maximizing profit. Lastly, the huge state space of the stock problem is a great contrast to the simplicity of the rock paper scissor problem. In summary, while Rock-Paper-Scissors is a simple game with a clear objective of winning against an opponent, Stock Trading is a complex and dynamic problem.

Discussion of the experiments: Stock Trading

For the Stock trading problem policy iteration converged after 778 iterations and The final policy matrix suggests that the optimal action for each state is to sell initially then hold for all later states. From out state definition it is clear that the initial state is where the price is low. Therefore, we can conclude that policy iteration suggests a conservative strategy by suggesting to sell when the price in lowest and then hold until the price is highest.

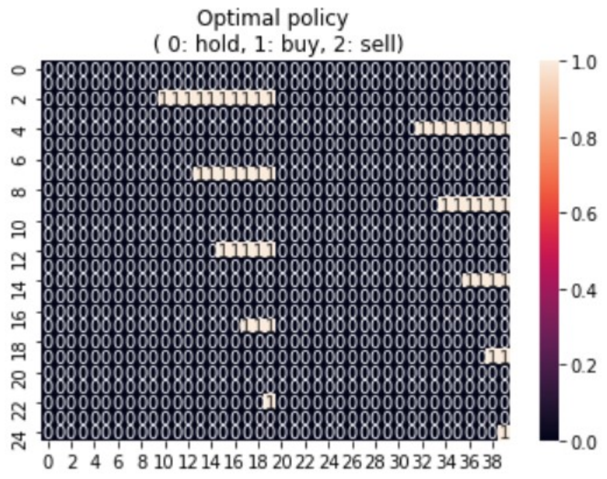


Value Iteration converged after 686 iterations to the same policy. The optimal value function gives the expected total reward starting from each state under the optimal policy. As the stock price increases, the value of being in that state also increases, which makes sense since higher prices lead to higher rewards.

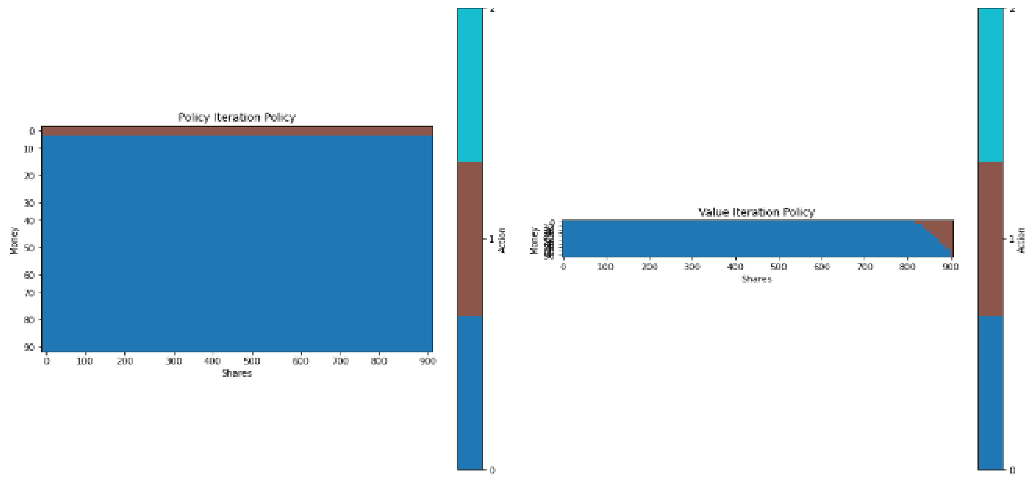


PPO’s results for the Stock problem suggested a very direct strategy form the other algorithms. It mostly chooses to "Buy" and "Hold" alternately. The rewards are mostly negative (-1) for choosing "Buy" and zero (0) for choosing "Hold". Overall , PPO's strategy doesn't seem to be consistently profitable.

When Comparing the methods we can see that Value Iteration converged after 686 iterations, which is fewer than the 778 iterations required by Policy Iteration. This observation was just as expected as value Iteration tends to converge faster because it updates the value function directly without waiting for the policy to stabilize. Although Policy Iteration starts with an initial policy and iteratively improves it, whereas Value Iteration focuses on updating the value function both of them converged to the same policy. Comparing PPO to Policy Iteration and Value Iteration, PPO's results appear less optimal. PPO shows a strategy with mostly negative rewards, indicating that further training and exploration are needed to find a more optimal strategy.



When looking at Computational Efficiency, Value Iteration is computationally more efficient than Policy Iteration in this case, as it requires fewer iterations to converge. However, each iteration of Value Iteration might be more computationally expensive than Policy Iteration due to the maximum operation involved.



In summary, both Policy Iteration and Value Iteration provided similar insights into the optimal stock trading strategy for this problem, with Value Iteration being slightly more computationally efficient in terms of iterations.

Discussion of the experiments: Rock-Paper-Scissor

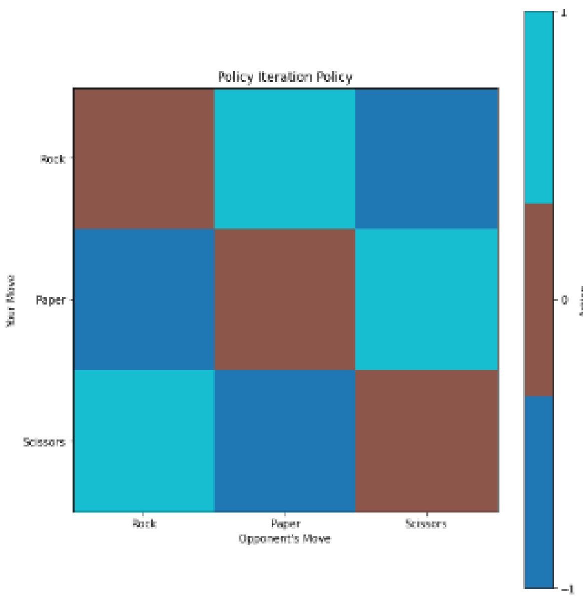
For Rock Paper Scissors problem Policy Iteration Results Converged after 2 iterations. And the Optimal Policy was found to be (rock: 'scissors', paper: 'scissors', scissors: 'scissors') meaning it believes that playing 'scissors' is the optimal move against any opponent's move, including 'scissors' itself.

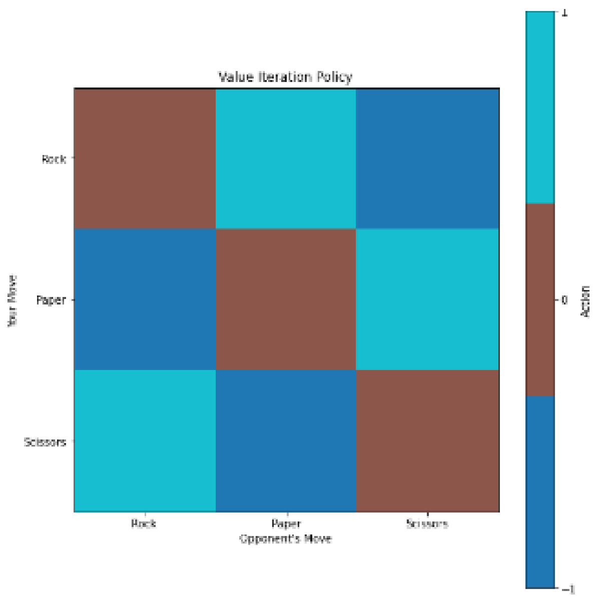
Value Iteration Converged after 2 iterations and suggested Optimal Values to be (rock:0 ,paper: 0,scissors: 0) This indicates that the expected reward (or value) for each action is the same, and none of the actions is better than the others in terms of maximizing the expected reward.

PPO seems to prefer choosing "Paper" and "Scissors" alternately. It picked rewards that are mostly negative (-1) for choosing "Paper" and positive (0) for choosing "Scissors". Overall PPO doesn't seem to have a clear strategy as the rewards are not consistently positive or negative.

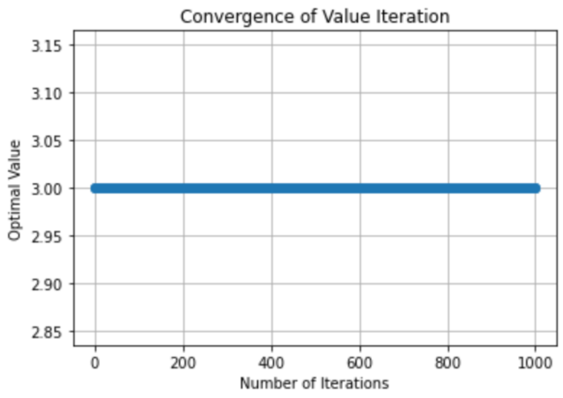
We can see that both algorithms In the context of Rock-Paper-Scissors, this result suggests that the game is fair under the optimal strategy computed by Value Iteration as shown in the figure below. No matter what action you choose, your expected reward (or the probability of winning) is the same, which makes sense for a fair game where each action has an equal chance of winning, losing, or drawing. Similarly PPOs results also seem vague

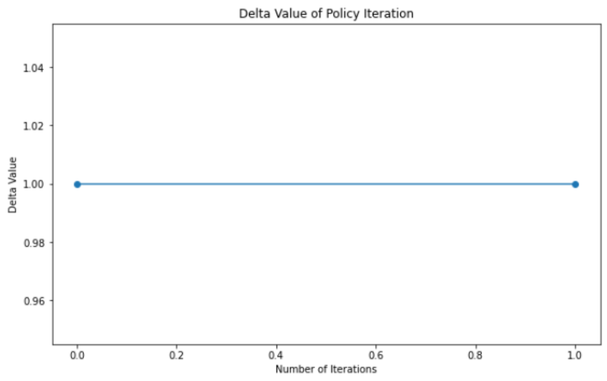
.





When considering Computational expense, it is expected that value Iteration will converge faster because it updates the value function directly without waiting for the policy to stabilize. However, given the scale of this problem is so small there is essentially no difference in the computational outputs of the problem. To show this I have included the convergence plots for the strategies below. Notice how quickly the functions converge.





Discussion of the experiments: Comparing both Problems.

When comparing the results of Rock-Paper-Scissors and Stock Trading Problem through the lens of Policy Iteration we can see that Rock-Paper-Scissors Converged after 2 iterations. While Stock Trading Converged after 778 iterations. Which was well expected as Stock trading problems often require more iterations to converge due to their complexity and real-world nuances. The Optimal Policy picked for RPS was {'rock': 'scissors', 'paper': 'scissors', 'scissors': 'scissors'} and for Sock trading it was to sell initially then hold for all later states. Both of these make sense. Although the rock-paper-scissors result might seem vague, it is important to remember that this just suggest the fairness of the problem itself.

When comparing the results of Rock-Paper-Scissors and Stock Trading Problem through the lens of Value Iteration Results we can see that Rock-Paper-Scissors Converged after 2 iterations and had Optimal Value of {'rock': 0, 'paper': 0, 'scissors': 0}. Similarly Stock Trading Converged after 686 iterations and also suggested to selling initially then holding after that. Again, the stock trading result is more practical and aligned with typical trading strategies.

For PPO the results were indecisive or wrong for both problems. For rock paper scissors the results kept changing, and for the stock problem the output seemed random, suggesting that no strategy was picked.

In conclusion, Rock-Paper-Scissors is a simple game but its optimal strategy is not that obvious. While stock Trading is a more complex problem, it has consistent results. Additionally, Stock trading problems often require more iterations to converge due to their complexity and real-world nuances. Overall, while both problems involve decision-making under uncertainty, the optimal strategies and insights derived from Policy Iteration and Value Iteration vary significantly based on the context.

On the other hand, it was found that PPO is computationally the best option, the optimal solution is not to be determined. There fore For a complex and dynamic environment like stock trading, Policy Iteration and Value Iteration are expected to outperform PPO once optimal policies/values are found.

Recourses and Citations

https://www.kaggle.com/code/charel/learn-by-example-reinforcement-learning-with-gym

https://stackoverflow.com/questions/69873630/modulenotfounderror-no-module-named-stable-baselines3

https://github.com/openai/gym/issues/2809