Exploring Reinforcement Learning: Techniques in Decision Making with Markov Decision Processes

Mar Tejedor Ninou
CS7641 Spring 2024

I. Introduction

Reinforcement learning is a fundamental concept in artificial intelligence and machine learning, enabling agents to learn optimal behaviors through interaction with environments. In this study, we focus on two MDP problems: Frozen Lake, challenging agents to navigate stochastic environments, and Forest Management, involving long-term planning and risk mitigation. Motivated by understanding RL algorithm effectiveness, we investigate Policy Iteration, Value Iteration, and Free Model QLearning in solving these problems.

II. Markov Decision Processes

In our exploration of Markov Decision Processes (MDPs), we have chosen two distinct problems to illustrate various aspects of reinforcement learning algorithms: Frozen Lake and Forest Management. These problems vary in complexity, with Frozen Lake representing a small-scale grid world scenario and Forest Management presenting a larger and more nuanced environment.

A. Frozen Lake

Frozen Lake provides a classic grid world problem, in our case as a modest 4x4 grid comprising 16 states. In this scenario, our agent traverses a frozen lake, aiming to reach a designated goal cell while evading treacherous holes that lead to certain failure. With actions limited to moving up, down, left, and right, the agent's objective is clear: navigate from the starting cell to the goal cell while maximizing cumulative rewards and minimizing step count. Despite its straightforward appearance, Frozen Lake introduces complexity through its stochastic environment, making the floor slippery, and providing uncertain outcomes. This simplicity coupled with inherent challenges makes it an ideal introductory problem for grasping fundamental MDP concepts and reinforcement learning algorithms. Also, Frozen Lake compact size facilitates rapid experimentation and comparison of different approaches, making it a staple in algorithmic testing and validation.

B. Forest Management

Forest Management presents a more expansive and intricate scenario, where our agent must decide when to harvest trees in a forest to maximize long-term rewards while mitigating the risk of wildfires. In this conceptualization, the forest's states represent the age of trees, and actions entail waiting for trees to mature or cutting them down for immediate gain. I our problem, aiming to make it bigger, the states will consist of 500 different ones. Unlike the short-term focus of Frozen Lake, Forest Management necessitates strategic long-term planning due to its stochastic complexity. Balancing immediate gains with future risks, the agent must navigate the uncertainty introduced by wildfire probabilities, adding layers of complexity to decision-making. A reward of 4 is granted when the forest is in its oldest state and the agent chooses to 'Wait'. However, if the 'Cut' action is taken in the oldest state, the reward reduces to 2. The probability of Wildfire Occurrence is 0.1.

III. Methodology

In our investigation of the Frozen Lake and Forest Management problems, we adopt a methodical approach centered on applying various reinforcement learning algorithms.

A. Policy Iteration

Policy iteration is an iterative algorithm that systematically evaluates and refines a policy until convergence. It comprises two main steps: policy evaluation, where the value function of the current policy is computed, and policy improvement, where the policy is updated to be greedy with respect to the current value function. By iteratively repeating these steps, policy iteration converges to an optimal policy.

B. Value Iteration

Value iteration shares similarities with policy iteration but differs in its approach to finding the optimal value function directly. Instead of explicitly maintaining a policy, value iteration iteratively computes the optimal value function of an MDP until convergence. By repeatedly updating the value function towards the optimal values, value iteration ultimately converges to the optimal policy.

C. Free model QLearning

QLearning is a model-free reinforcement learning algorithm that learns the optimal action-value function without requiring explicit knowledge of the transition probabilities. QLearning operates through exploration and exploitation of state-action pairs, updating Q-values based on observed rewards and transitions. By iteratively updating Q-values according to a specified learning rate and discount factor, QLearning converges to the optimal action-value function.

IV. Frozen Lake

A. Policy Iteration

In this section, we delve into the performance of the Policy Iteration algorithm applied to the Frozen Lake 4x4 environment. We analyzed several key metrics to understand how the algorithm behaves under different conditions, including time to convergence, average rewards, and the number of iterations required for convergence.

The time taken to reach convergence exhibits notable fluctuations based on the chosen discount factor. Particularly, when the discount factor is set to 0.85 , the computation time is considerably higher compared to other values, Figure 1. This suggests that higher discount factors may lead to longer computation times due to the increased emphasis on future rewards.

We observe an intriguing trend in average rewards concerning the discount factor, Figure 2. As the discount factor increases, there is an exponential growth in average rewards, with the highest rewards achieved when the discount factor is set to 0.95. This phenomenon indicates that higher discount factors incentivize the agent to prioritize long-term rewards, resulting in overall better performance.

The relationship between the discount factor and the number of iterations required for convergence reveals interesting insights, Figure

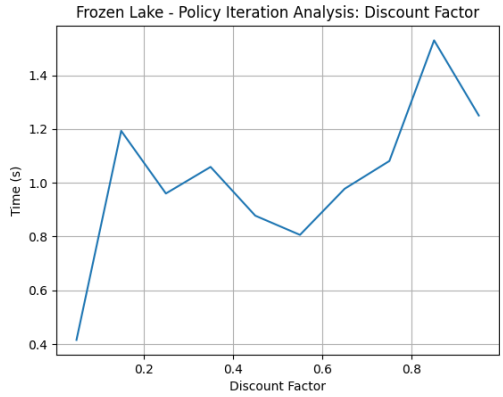

Fig. 1. Frozen Lake Policy Iteration Discount Factor vs Time.

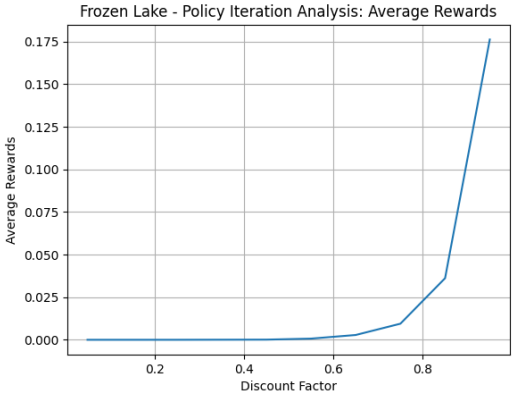

Fig. 2. Frozen Lake Policy Iteration Discount Factor vs Average Rewards.

3. Notably, there are distinct peaks in the number of iterations needed to converge, observed at discount factor 0.85, where 6 iterations are required. Conversely, setting the discount factor to 0.95 leads to faster convergence, requiring only 3 iterations. This implies that higher discount factors may facilitate quicker convergence due to the agent's increased focus on future rewards.

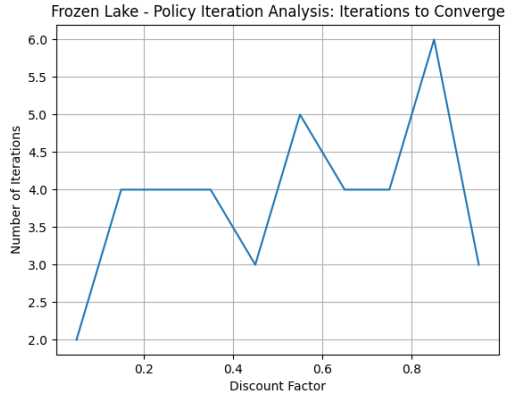

Fig. 3. Frozen Lake Policy Iteration Discount Factor vs Number of Iterations to converge.

In Policy Iteration, convergence is reached when the policy stabilizes, maximizing the agent's performance with consistent average rewards. It consistently converges to the same optimal policy across different discount factors, showcasing robustness. Despite the stochastic nature of the environment, and not making it to the goal some times, Table I, Policy Iteration achieves commendable average rewards, making it a reliable tool for navigating the Frozen Lake 4x4 environment. The best optimal policy obtained from Policy Iteration is characterized by a representation where the first state corresponds to "wait to cut the tree" (0), while all subsequent states indicate "cut the tree" (1), except for the last 13 states, which return to "wait to cut the tree" (0). This policy signifies that initially, it's advantageous to delay cutting trees to minimize risks, followed by a period of aggressive tree cutting to capitalize on the favorable conditions, and finally, reverting to a cautious approach to mitigate potential hazards toward the end of the task.

B. Value Iteration

In this section, we analyze the performance of the Value Iteration algorithm applied to the Frozen Lake 4x4 environment. The time taken to achieve convergence varies significantly depending on the chosen discount factor, Figure 4. Notably, higher discount factors of 0.85 and 0.95 lead to considerably longer computation times compared to other values. This suggests that prioritizing future rewards to a greater extent may require more computational resources and time.

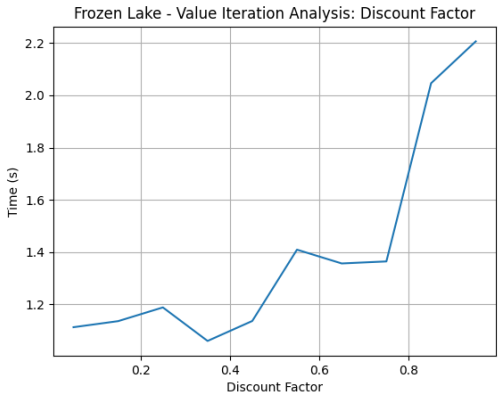

Fig. 4. Frozen Lake Value Iteration Discount Factor vs Time.

There is an exponential increase in average rewards as the discount factor increases, Figure 5, with the highest rewards observed when the discount factor is set to 0.95, reaching 0.19. This trend indicates that higher discount factors encourage the agent to focus more on long-term rewards, resulting in improved overall performance.

The relationship between the discount factor and the number of iterations needed for convergence reveals an exponential increase in the number of iterations as the discount factor approaches 0.95. In particular, setting the discount factor to 0.95 necessitates up to 420 iterations to converge as seen in Figure 6. This suggests that higher discount factors may require more iterations to achieve convergence due to the increased emphasis on future rewards.

In Value Iteration, convergence occurs when the value function stabilizes, accurately estimating cumulative rewards from each state. Despite the stochastic environment, it achieves good average rewards, demonstrating its effectiveness. Interestingly, the best policy obtained from Value Iteration mirrors that of Policy Iteration: the optimal

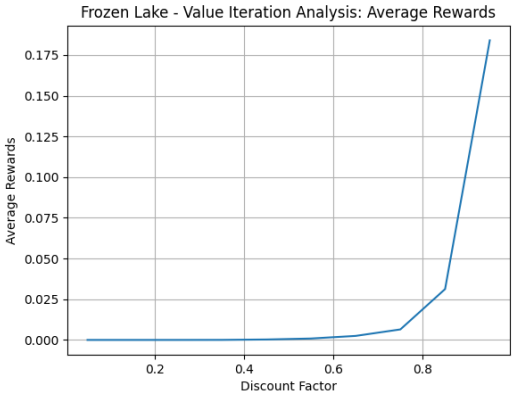

Fig. 5. Frozen Lake Value Iteration Discount Factor vs Average Rewards.

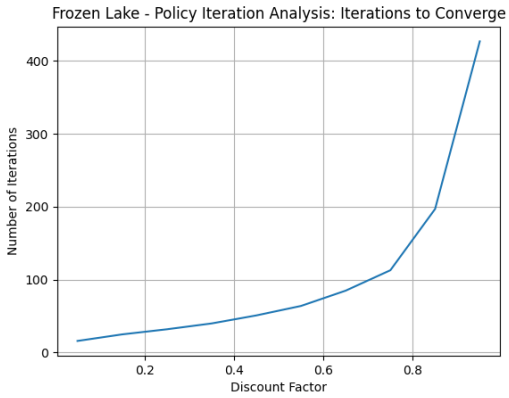

Fig. 6. Frozen Lake Value Iteration Discount Factor vs Number of Iterations to converge.

policy features an initial state of "wait to cut the tree" (0), followed by all subsequent states indicating "cut the tree" (1), except for the last 13 states, which return to "wait to cut the tree" (0). This consistency in optimal policy across both algorithms is noteworthy and suggests a robust convergence towards a specific solution despite different computational approaches.

C. QLearning

In this section, we explore the application of the QLearning algorithm to the Frozen Lake environment, a classic problem in reinforcement learning. QLearning is a model-free reinforcement learning algorithm that learns directly from interactions with the environment without requiring a model of the environment's dynamics. We analyze various aspects of QLearning performance and discuss the significance of selecting optimal parameters.

Epsilon regulates the trade-off between exploration and exploitation in QLearning. Higher epsilon values promote more exploration, while lower values prioritize exploiting learned knowledge. Through experimentation, we observed that epsilon values around 0.5 resulted in brighter colors in the Q-array plot, Figure 7, indicating higher Q-values across more states. This suggests that a moderate level of exploration facilitated a better understanding of the state-action space. Additionally, the average rewards versus episodes graph revealed that epsilon values of 0.5 yielded higher average rewards over time, demonstrating the effectiveness of balanced exploration and exploitation, Figure 8.

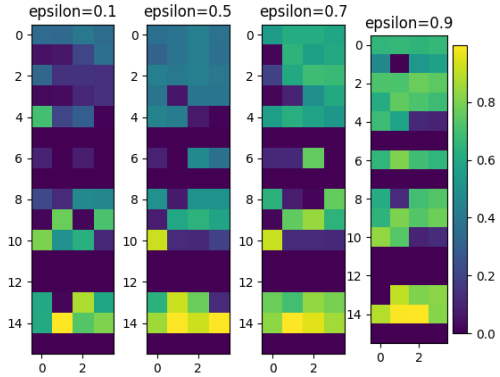

Fig. 7. Frozen Lake QLearning Epsilon Analysis using Q array heat map.

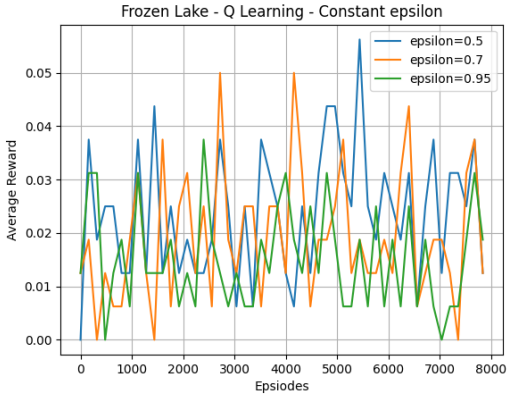

Fig. 8. Frozen Lake QLearning Epsilon Analysis using average rewards vs number of episodes.

Gamma, or the discount factor, influences the significance of future rewards in QLearning. Higher gamma values prioritize long-term rewards over immediate gains. Our analysis showed that gamma values around 0.9 resulted in brighter colors in the Q-array plot, Figure 9, indicating higher Q-values across more states. However, the impact of gamma on performance was not as pronounced as other parameters. Furthermore, the average rewards versus episodes graph indicated that gamma values of 0.95 and 0.7 led to higher average rewards, with faster convergence achieved at gamma 0.95, Figure 10.

Alpha, known as the learning rate, determines the degree to which new information overrides existing Q-values. Higher alpha values facilitate faster learning but may lead to instability. Experimentation revealed that alpha values around 0.7 resulted in brighter colors in the Q-array plot, indicating higher Q-values across more states. However, the impact of alpha on performance was less significant. Additionally, the average rewards versus episodes graph demonstrated that an alpha value of 0.5 led to higher average rewards, suggesting that a moderate learning rate facilitated better learning over time.

As evidenced by the graphical representation in Figure 14 it is apparent that we did not achieve the goal of successful navigation in the Frozen Lake environment. This outcome can be attributed to the stochastic nature of the problem, exacerbated by setting

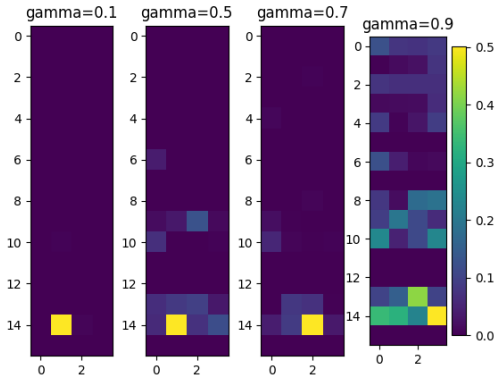

Fig. 9. Frozen Lake QLearning Gamma Analysis using Q array heat map.

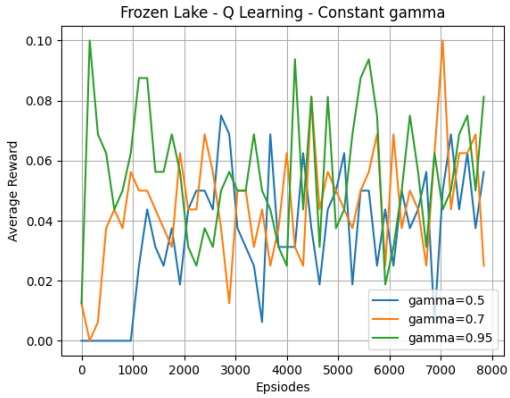

Fig. 10. Frozen Lake QLearning Gamma Analysis using average rewards vs number of episodes.

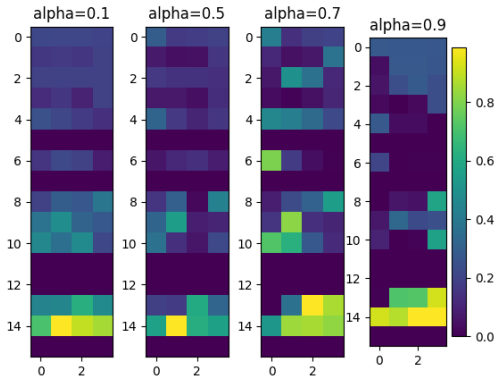

Fig. 11. Frozen Lake QLearning Alpha Analysis using Q array heat map.

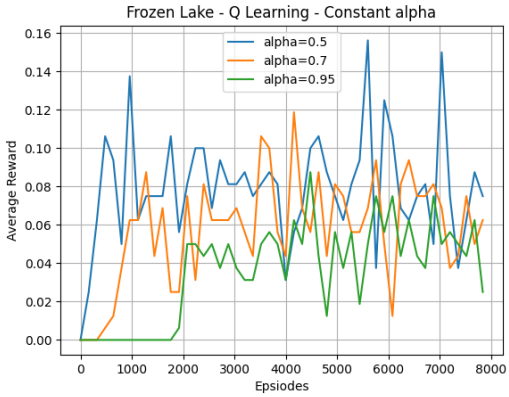

Fig. 12. Frozen Lake QLearning Alpha Analysis using average rewards vs number of episodes.

'is_slippery=True', which introduces unpredictability into the agent's movements.

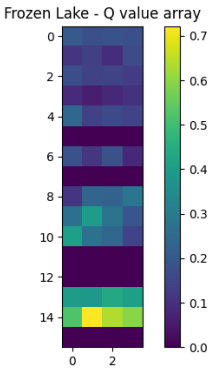

Fig. 13. Frozen Lake QLearning Best Policy Q Array.

Despite our efforts to optimize the QLearning algorithm, Figure 15, with parameters such as Alpha = 0.15, Gamma = 0.95, Epsilon = 0.3, and conducting 12,000 episodes, the stochasticity of the environment posed challenges to achieving the desired outcome. The Q-array plot indicates effective learning, Figure 13, showcasing the agent's attempts to understand the environment, but higher average rewards and successful navigation were not consistently attained.

In conclusion, while QLearning remains a potent tool for tackling reinforcement learning problems, including the Frozen Lake scenario, its performance can be hindered by the inherent stochasticity of the environment, as well for Policy and Value iteration. Further exploration and refinement of algorithms and parameters may be necessary to overcome these challenges and achieve desired objectives in similar scenarios.

D. Analysis and Comparison

In this section, we delve into a comprehensive analysis and comparison of three reinforcement learning algorithms - Policy Iteration, Value Iteration, and QLearning - applied to the Frozen Lake problem. Through this examination, we aim to discern their performance across various metrics and elucidate the strengths and weaknesses inherent in each approach. The key findings derived from executing each

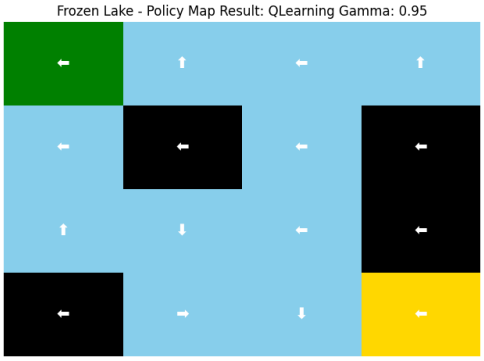

Fig. 14. Frozen Lake QLearning Best Policy 2D Representation.

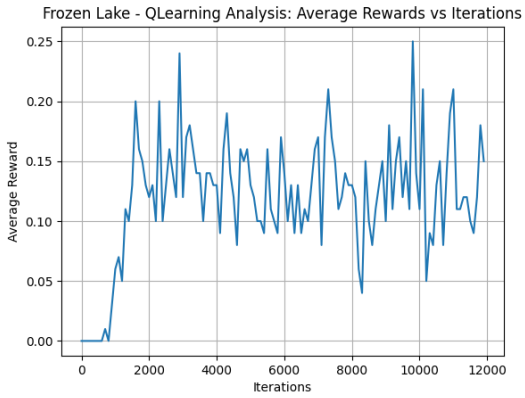

Fig. 15. Frozen Lake QLearning Average Rewards vs Iterations.

algorithm on the Frozen Lake environment are succinctly summarized in Table I below.

TABLE I
ALGORITHMS RESULTS FOR FROZEN LAKE 4X4 (IS_SLIPPERY=TRUE)

	Policy Iteration	**Value Iteration**	**QLearning**
Final Score	0.183	0.175	0.157
Average Steps	46	41	30
Fall in Hole %	23.10%	21.5%	53.9%
Time	1.369	1.631	5.532

The final scores achieved by the algorithms provide insight into their overall success in navigating the Frozen Lake environment. Policy Iteration attained the highest final score of 0.183, closely followed by Value Iteration with a score of 0.175. QLearning yielded a slightly lower final score of 0.157, indicating differences in their performance in trying to reach the goal state.

In terms of efficiency, QLearning emerged as the most effective algorithm, requiring only 30 steps on average to reach the goal state. In contrast, Policy Iteration and Value Iteration necessitated 46 and 41 steps, respectively. This discrepancy suggests that QLearning managed to discover more optimal paths to the goal more efficiently than the other algorithms.

The percentage of falls into the hole provides insight into the exploration-exploitation trade-off employed by the algorithms. QLearning exhibited the highest percentage of falls, indicating a greater propensity for exploration compared to Policy Iteration and Value Iteration. This behavior aligns with the nature of QLearning, which explores uncertain actions to uncover superior policies.

In terms of computational performance, Policy Iteration demonstrated the fastest execution, accounting for only 1.369 seconds. Value Iteration followed closely behind, consuming 1.631 seconds. In contrast, QLearning proved to be the most time-intensive, requiring 5.532 seconds. This difference can be attributed to the iterative nature of Policy and Value Iteration, contrasting with the trial-and-error learning approach employed by QLearning.

In conclusion, each algorithm exhibits distinct characteristics and trade-offs when applied to the Frozen Lake problem. Policy Iteration excels in accuracy and relatively fast convergence but may incur higher computational costs. Value Iteration strikes a balance between accuracy and efficiency. QLearning, while slower and more exploratory, shines in discovering optimal policies through trial-and-error learning. The selection of the most suitable algorithm in this case would be Policy Iteration, however, the best algorithm always hinges on the specific requirements of the problem and the desired trade-offs between accuracy, efficiency, and exploration.

V. FOREST MANAGEMENT

A. Policy Iteration

In this section, we analyze the results obtained from solving the Forest Management problem using policy iteration. We focus on three key aspects: the impact of the discount factor on convergence time, the average rewards obtained with different discount factors, and the iterations required for convergence across various discount factors.

In figure 16, we observed that the time taken for policy iteration increases significantly when the discount factor exceeds 0.8. This trend is reasonable because higher discount factors require more iterations to converge, leading to longer computation times.

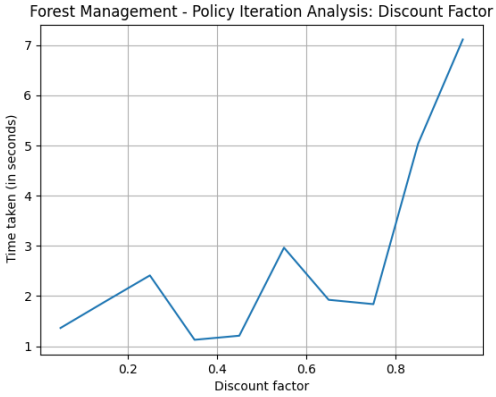

Fig. 16. Forest Management Policy Iteration Discount Factor vs Time.

When examining the relationship between the discount factor and average rewards, we found an exponential growth in average rewards with higher discount factors, particularly at 0.95, Figure 17. This phenomenon suggests that higher discount factors prioritize long-term rewards, resulting in policies that yield higher cumulative rewards over time.

The iterations required for convergence increase as the discount factor rises as seen in Figure 18. Specifically, we observed that the number of iterations to converge is around 13 when the discount factor is higher. This trend implies that higher discount factors necessitate more iterations to achieve convergence, indicating a trade-off between long-term planning and computational efficiency.

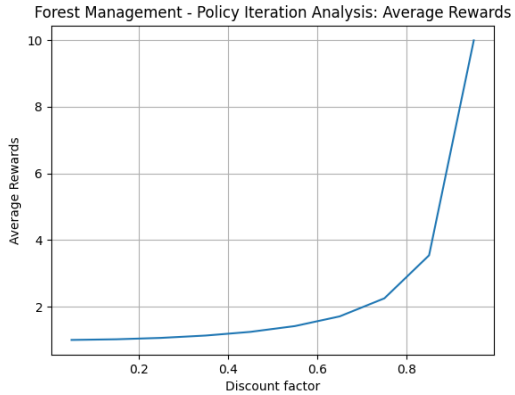

Fig. 17. Forest Management Policy Iteration Discount Factor vs Average Rewards.

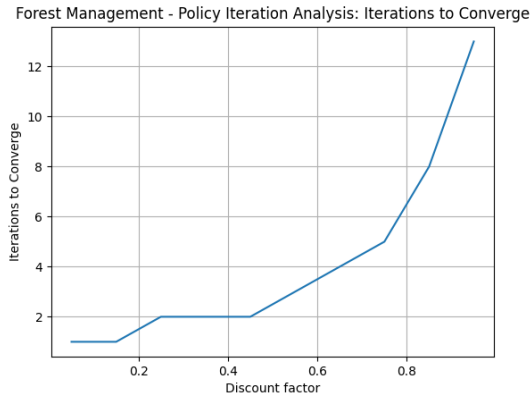

Fig. 18. Forest Management Policy Iteration Discount Factor vs Number of Iterations to converge.

For optimal results, we recommend selecting a discount factor that balances computational efficiency with long-term rewards. A value around 0.95 often strikes this balance, offering efficient convergence while maximizing average rewards. However, the ideal discount factor depends on specific problem objectives and constraints. Lower discount factors converge faster but prioritize short-term rewards, whereas higher ones favor long-term rewards but require more iterations. In our analysis, a discount factor of 0.95, with 500 states and a wildfire probability of 0.1, yielded favorable results.

B. Value Iteration

Now it is time to analyze the results obtained from solving the Forest Management problem using value iteration. Unlike policy iteration, the relationship between the discount factor and convergence time in value iteration is highly irregular, Figure 19. We observed higher computational times when the discount factor is 0.5, which may be attributed to the algorithm's behavior in prioritizing short-term rewards over long-term planning at lower discount factors.

Similar to policy iteration, the average rewards exhibit exponential growth with higher discount factors as seen in Figure 20, particularly at 0.95. This trend suggests that higher discount factors lead to policies that prioritize long-term rewards, resulting in higher cumulative rewards over time.

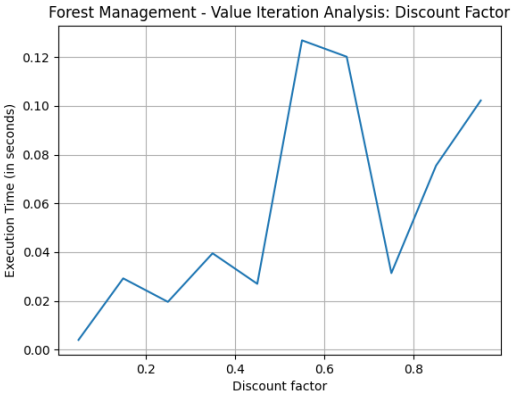

Fig. 19. Forest Management Value Iteration Discount Factor vs Time.

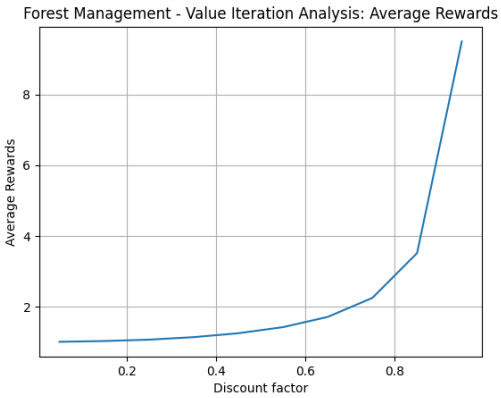

Fig. 20. Forest Management Value Iteration Discount Factor vs Average Rewards.

The iterations required for convergence exhibit an exponential growth pattern as the discount factor increases, Figure 21. Specifically, we found that it takes around 58 iterations to converge when the discount factor is higher. This trend underscores the trade-off between computational complexity and long-term planning inherent in value iteration.

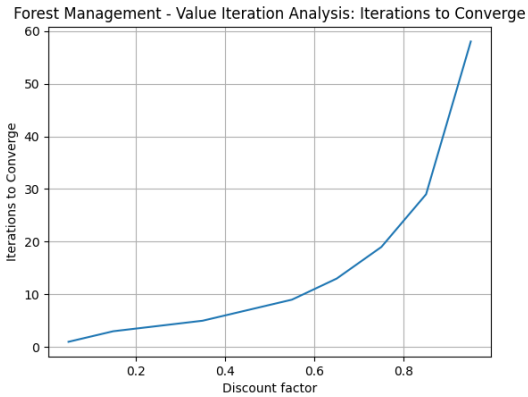

Fig. 21. Forest Management Value Iteration Discount Factor vs Number of Iterations to converge.

Compared to policy iteration, value iteration shows irregular convergence times across various discount factors, possibly due to its balancing of short-term rewards and long-term planning. Despite this, the exponential growth pattern between the discount factor and average rewards remains consistent. Higher discount factors prioritize long-term rewards, resulting in higher average rewards over time. Value iteration's convergence speed generally lags behind policy iteration, especially with higher discount factors, likely due to its iterative nature and the need for multiple value function updates per iteration. Despite differences, both algorithms converge to the same optimal policies and value functions for the forest management problem. Larger state spaces may increase convergence time. Overall, while both methods are effective, policy iteration tends to converge faster and yield higher average rewards, making it preferable for scenarios prioritizing long-term reward maximization.

C. QLearning

In this section, we explore the application of QLearning, a model-free reinforcement learning algorithm, for solving the Forest Management problem. QLearning offers several advantages, particularly in scenarios where the transition dynamics and reward structure of the environment are unknown or difficult to model accurately. We analyze the impact of different hyperparameters on the performance of QLearning and discuss the rationale behind selecting optimal parameter values.

The parameter epsilon controls the balance between exploration and exploitation in QLearning, Figure 22. We observe three distinct peaks in the average value of states, with the highest value achieved when epsilon is set to 0.7. This indicates that a moderate level of exploration leads to better rewards. Additionally, setting epsilon to 0.7 results in significantly less time consumption compared to other epsilon values. This is because moderate exploration allows the agent to discover optimal policies efficiently without excessively exploring suboptimal actions.

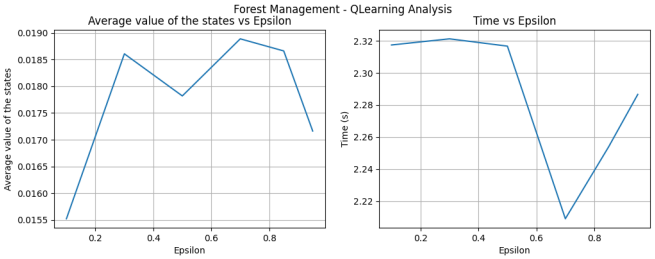

Fig. 22. Forest Management QLearning Analysis Epsilon vs Average val of states and Time.

Gamma, the discount factor, determines the importance of future rewards in QLearning, Figure 23. We observe an exponentially increasing trend in the average value of states, with higher values achieved when gamma is set to 0.95. Interestingly, setting gamma to 0.85 and 0.95 results in less time consumption compared to other gamma values. This is because higher gamma values prioritize long-term rewards, leading to faster convergence to optimal policies.

The learning rate (alpha) controls the rate at which the Q-values are updated in QLearning. We observe in Figure 24 a linear increasing trend in the average value of states with a slight decay towards the end. Higher values of alpha, particularly 0.85, result in higher average rewards. Moreover, choosing a learning rate of 0.8 leads to much less time consumption compared to a learning rate of 0.5 when there is a peak in average value. This is because a higher learning rate

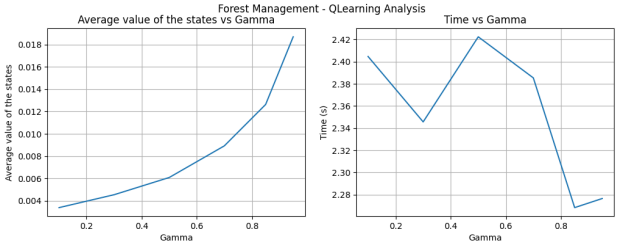

Fig. 23. Forest Management QLearning Analysis Gamma vs Average value of states and Time.

accelerates the learning process, allowing the agent to update Q-values more quickly.

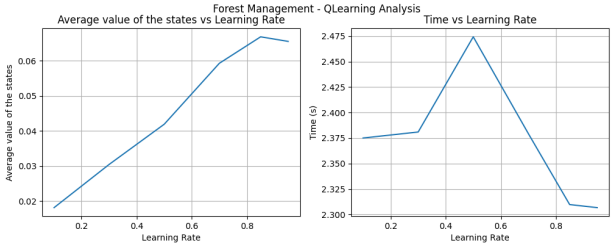

Fig. 24. Forest Management QLearning Analysis Alpha vs Average value of states and Time.

The number of states in the forest management problem directly impacts, as seen in Figure 25 the complexity of the environment. We observe an exponentially decreasing trend in the average value of states as the number of states increases. Therefore, we selected a problem with 500 states for large forest management scenarios, as it strikes a balance between complexity and computational efficiency. Additionally, the time required for QLearning increases linearly as the number of states increases, reflecting the increased computational burden.

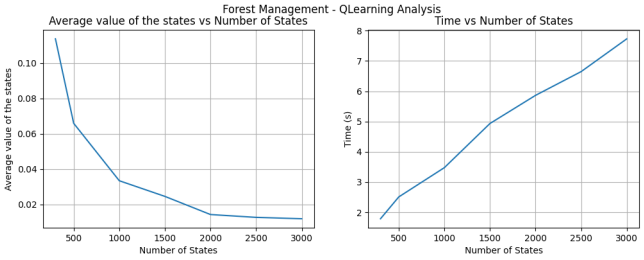

Fig. 25. Forest Management QLearning Analysis Number of States vs Average value of states and Time.

The fire probability parameter represents the likelihood of a fire occurring in the forest, Figure 26. We observe a linear decreasing trend in the average value of states, with a peak at a fire probability of 0.05. Moreover, setting the fire probability to 0.05 results in higher time consumption compared to other probabilities. This is because a higher fire probability introduces uncertainty and volatility into the environment, making it more challenging for the agent to learn optimal policies.

In summary, QLearning offers a flexible and robust approach to solving the Forest Management problem, allowing for efficient policy learning in complex and uncertain environments. By carefully tuning the hyperparameters such as epsilon at 0.7, gamma at 0.95, learning

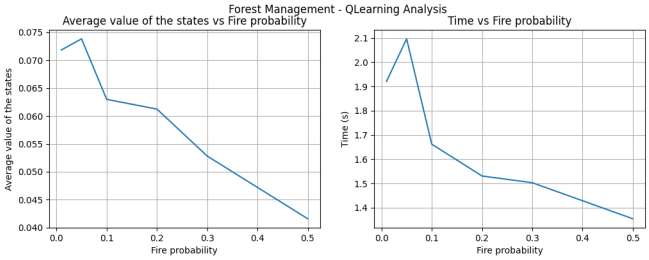

Fig. 26. Forest Management QLearning Analysis Fire Probability vs Average value of states and Time.

rate at 0.95, and considering the characteristics of the environment (500 states and 0.1 wildfire probability), we can effectively train an agent to make informed decisions and maximize long-term rewards in forest management scenarios.

The Q-array plot (Figure 27) visually represents the learned Q-values for each state-action pair in the Forest Management problem. Brighter colors indicate higher Q-values, representing the expected cumulative reward for taking a specific action in a given state. Since the colors are only bright on the top part of the Q-array plot, it suggests that we achieved a decent policy, with the agent having learned the some optimal actions for the initial state-action pairs in the environment, but not for the following ones. This visualization provides insights into the learned policy and helps evaluate the effectiveness, however, the plot illustrates that QLearning has not yielded satisfactory results in this scenario, indicating its limitations for this particular problem.

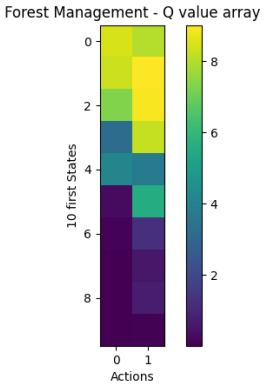

Fig. 27. Forest Management QLearning Analysis First 10 States Q value Array

D. Analysis and Comparison

In this section, we compare the performance of three different algorithms—Policy Iteration, Value Iteration, and QLearning—for solving the Forest Management problem.

While Policy Iteration and Value Iteration demonstrated superior performance in terms of both final score and convergence speed, it's essential to delve deeper into the underlying mechanisms driving their success, Table II. Policy Iteration, renowned for its ability to converge rapidly, leverages a systematic approach to iteratively refine policies, maximizing cumulative rewards efficiently. Its deterministic exploration strategy allows it to navigate the environment with precision, exploiting known dynamics to achieve optimal results

TABLE II
ALGORITHMS RESULTS FOR FOREST MANAGEMENT (S=500, P=0.1)

	Policy Iteration	**Value Iteration**	**QLearning**
Best Score	9.99	9.506	0.103
Number Iterations	13	58	10000
Best Time	4.27	0.088	10.432

Both Policy and Value Iteration converge to the same best policy.

swiftly. Similarly, Value Iteration exhibits remarkable efficiency in converging to optimal value functions, despite requiring slightly more iterations than Policy Iteration. By iteratively updating value estimates, Value Iteration strikes a balance between exploration and exploitation, effectively leveraging environment dynamics to drive policy improvement. For this problem we would use any of these ones as our algorithms due to their good results.

In contrast, QLearning's performance in the Forest Management problem was hindered by its reliance on an epsilon-greedy exploration strategy having the worst results compared to the other two algorithms. While QLearning offers flexibility in adapting to uncertain or complex environments, its efficacy diminishes in scenarios with clearly defined dynamics. Despite its challenges, QLearning remains a valuable tool for tackling reinforcement learning problems, particularly those characterized by ambiguity and volatility.

VI. CONCLUSIONS

In this research, we explored three reinforcement learning algorithms—Policy Iteration, Value Iteration, and QLearning—in two distinct environments: Frozen Lake and Forest Management. Each algorithm was analyzed based on various metrics such as convergence time, average rewards, and computational efficiency to understand their performance characteristics and suitability for different scenarios.

For the Frozen Lake environment, Policy Iteration demonstrated robustness and efficiency, achieving the highest final score of 0.183 with relatively fast convergence in 13 iterations. Value Iteration, although slightly slower, also performed well with a final score of 0.175. QLearning, while exhibiting exploration capabilities, faced challenges due to the stochastic nature of the environment, achieving a lower final score of 0.157. However, QLearning demonstrated efficiency in terms of steps required to reach the goal state.

In the Forest Management problem, both Policy Iteration and Value Iteration showcased their effectiveness in leveraging known environment dynamics, achieving high scores of 9.99 and 9.506, respectively. QLearning, despite its flexibility, struggled with convergence and achieved a significantly lower score of 0.103. Policy Iteration and Value Iteration outperformed QLearning in terms of convergence time and computational efficiency, highlighting the importance of deterministic exploration strategies in well-defined environments.

Overall, Policy Iteration and Value Iteration algorithms emerged as superior choices for environments with known dynamics, offering reliable convergence and high performance. QLearning, while valuable in uncertain or complex environments, requires careful hyperparameter tuning and may face challenges in achieving optimal performance in well-defined scenarios. The selection of the most suitable algorithm depends on the specific characteristics of the environment and the desired trade-offs between exploration, exploitation, and computational complexity. Further research could focus on refining QLearning parameters and exploring hybrid approaches to leverage the strengths of different algorithms in diverse environments.