CS 7641 Machine Learning: Markov Decision Processes

Jilong Cui

jcui@gatech.edu

1 OVERVIEW

This report presents an analysis of two Markov Decision Problems (MDPs), Frozen Lake and Forest Management, using three reinforcement learning algorithms: Value Iteration, Policy Iteration, and Q-Learning. The Frozen Lake environment is evaluated in both 4x4 and 16x16 grid sizes, whereas the Forest Management scenario is investigated in a complex model with 20 and 500 states. The algorithms' parameters were optimized to ensure optimal performance within these distinct environments. The behaviors and outcomes of the algorithms within various environments were observed and analyzed.

1.1 Frozen Lake

The Frozen Lake environment is a grid-world where the agent's goal is to move from the top-left to the bottom-right grid cell. The environment includes safe cells and "holes." Falling into a hole ends the episode with no reward, while reaching the goal yields a reward of 1. Each cell in the grid represents a distinct state. For a 4x4 grid, there are 16 states; for a 16x16 grid, there are 256 states. The agent can move UP, LEFT, DOWN, or RIGHT. However, due to the environment's stochastic nature, the actual movement outcome is probabilistic: each action has a 0.33 probability of moving in the intended direction and 0.33 probabilities for the two perpendicular directions. The agent receives a reward of 0 for falling into a hole and 1 for reaching the goal. The optimal policy should maximize the probability of reaching the goal while minimizing the risk of falling into holes. This involves calculating the value of each action at each state, considering the stochastic movement. Convergence in this problem would mean finding a stable policy where the expected rewards from each state under the policy do not change with further iterations of policy evaluation and improvement.

1.2 Forest Management

This is a non-grid world environment involving forest growth management. The agent must decide between cutting down the forest for immediate rewards or waiting for potentially larger future rewards. Each state represents the growth level of the forest, with the forest growing to a maximum size. The agent has two actions:

- CUT (1): Harvests the forest, yielding a reward of R1 and resetting the forest to its initial growth state, forfeiting future larger rewards.
- WAIT (0): Offers a chance for the forest to grow to the next stage with a probability p, stacking a reward of R2, potentially leading to larger future rewards. However, with probability 1−p, a forest fire occurs, resetting the forest to state-0 and negating any future rewards.

The choice of CUT at any state guarantees a smaller, immediate reward but foregoes the chance of larger future rewards. The optimal policy involves calculating long-term returns, factoring in the probabilities of forest growth and fire, to decide between CUT and WAIT at each state. Convergence in Forest Management is reached when the policy stabilizes, with the expected long-term returns from each state remaining consistent upon further evaluations.

1.3 Summary of Problem

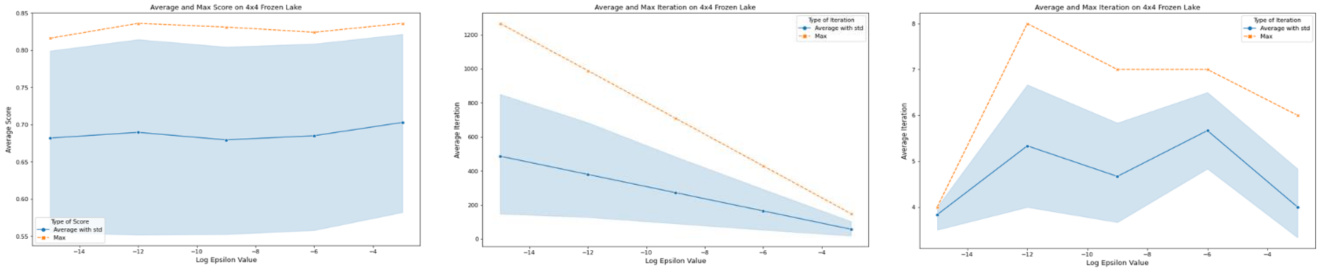
Both the Frozen Lake and Forest Management scenarios involve decision-making under uncertainty, necessitating the development of policies that optimize expected rewards, with convergence achieved when these policies stabilize, marked by minimal changes in value functions or policies between iterations. Frozen Lake is characterized by its spatial and grid-based environment, focusing on physical navigation through a probabilistic landscape, where complexity arises from grid layout and movement uncertainty. In contrast, Forest Management operates in a non-spatial context, concentrating on temporal resource management, with complexity stemming from managing forest growth stages and the unpredictability of forest fires. Decision-making in Frozen Lake entails immediate spatial consequences, whereas Forest Management involves strategic, temporal decisions that weigh short-term gains against potential future rewards. The reward system in Frozen Lake is binary, prioritizing safe navigation, while Forest Management employs a more intricate reward structure, requiring a balance between immediate and future rewards under the influence of forest growth and fire probabilities.

2 FRONZEN LAKE

In the analysis of the Frozen Lake Markov Decision Problem, two distinct grid sizes are considered: a 4x4 and a 16x16 grid, with a primary focus on the 4x4 grid. The 4x4 and 16x16 grid map was generated to have 4 and 42 traps, respectively, which can end the game. The criterion for convergence in these models is established as the stabilization of rewards, indicating that a consistent policy has been achieved. A critical parameter in this analysis is 'Epsilon' (ϵ). In the context of the Frozen Lake problem, ϵ plays a pivotal role in determining when the state can be considered stabilized. It essentially measures the threshold for changes in the policy or value functions, with smaller values of ϵ indicating a more stringent requirement for convergence. Another primary parameter under consideration is the discount rate (γ), which balances immediate versus future rewards.

2.1 Value and Policy Iteration on Frozen Lake

In both Value Iteration and Policy Iteration, ϵ is used to define convergence. The impact of ϵ is primarily on the number of iterations and the time taken to converge, rather than on the final reward of the model. As depicted in Figure 1, while the reward is insensitive to changes in ϵ, the number of iterations required for convergence varies with its value.



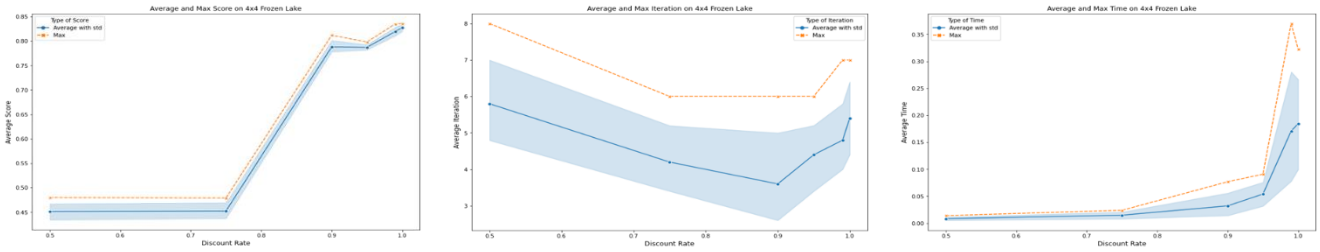
(a) V.I.: Epsilon vs. Reward

(b) V.I.: Epsilon vs. Converge Iteration

(c) P.I.: Epsilon vs. Converge Iteration

Figure 1. Frozen Lake 4x4 – Epsilon vs. Metrics

Figure 2 illustrates the relationship between the discount rate and key metrics like reward, convergence iterations, and time. Both Value and Policy Iteration exhibit a similar trend where the rewards are approximately equal. However, Value Iteration tends to require significantly more iterations to converge compared to Policy Iteration, which converges in relatively fewer steps. Interestingly, despite requiring more iterations, Value Iteration generally converges in less time than Policy Iteration. This indicates a trade-off between the number of iterations and the total time to convergence. The difference in convergence times can be attributed to the inherent computational complexity of each algorithm. Value Iteration, while iterative in nature, often requires simpler computations per iteration than Policy Iteration, which involves a more complex policy evaluation step.



(a) V.I.: Discount Rate vs. Reward

(b) P.I.: Discount Rate vs. Converge Iter.

(c) P.I.: Discount Rate vs. Time

Figure 2. Frozen Lake 4x4 – Discount Rate vs. Metrics

As shown in Figure 3, both Value and Policy Iteration generate the same policy for the Frozen Lake problem. This consistency arises from the deterministic nature of the optimal policy in such environments, where both algorithms effectively identify the best actions to maximize rewards. Figure 4 summarizes the performance metrics for the two iteration strategies. In summary, a higher discount rate generally leads to more iterations and longer convergence times but results in higher rewards. This is because a larger γ values future rewards more, prompting the algorithm to explore longer-term strategies. On the other hand, a smaller ϵ leads to more iterations to ensure finer convergence but does not influence the reward, as it is a measure of the stability of the policy rather than its effectiveness.

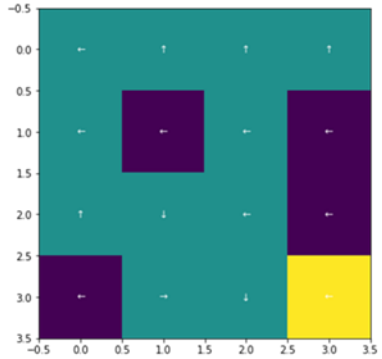
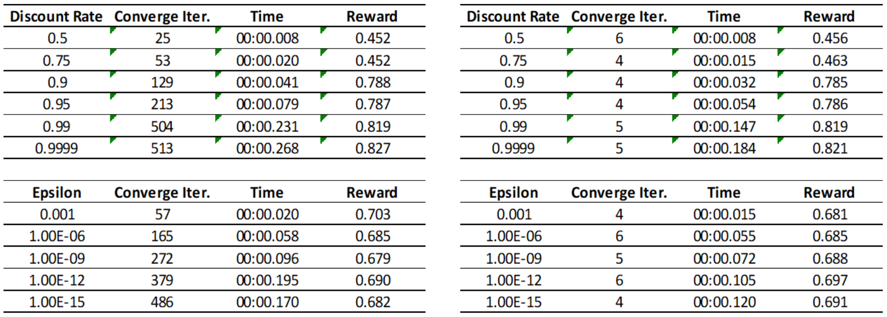


Figure 3. Value and Policy Iteration Policy



Discount Rate	Converge Iter.	Time	Reward
0.5	25	00:00.008	0.452
0.75	53	00:00.020	0.452
0.9	129	00:00.041	0.788
0.95	213	00:00.079	0.787
0.99	504	00:00.231	0.819
0.9999	513	00:00.268	0.827

Epsilon	Converge Iter.	Time	Reward
0.001	57	00:00.020	0.703
1.00E-06	165	00:00.058	0.685
1.00E-09	272	00:00.096	0.679
1.00E-12	379	00:00.195	0.690
1.00E-15	486	00:00.170	0.682

(a) V.I.: Epsilon vs. Converge Iteration

Discount Rate	Converge Iter.	Time	Reward
0.5	6	00:00.008	0.456
0.75	4	00:00.015	0.463
0.9	4	00:00.032	0.785
0.95	4	00:00.054	0.786
0.99	5	00:00.147	0.819
0.9999	5	00:00.184	0.821

Epsilon	Converge Iter.	Time	Reward
0.001	4	00:00.015	0.681
1.00E-06	6	00:00.055	0.685
1.00E-09	5	00:00.072	0.688
1.00E-12	6	00:00.105	0.697
1.00E-15	4	00:00.120	0.691

(b) P.I.: Epsilon vs. Converge Iteration

Figure 4. Performance Metrics for various Discount Rate and Epsilon

2.2 Q-Learning on Frozen Lake

Q-Learning, a prominent model-free reinforcement learning algorithm, operates by learning the value of actions in each state without a model of the environment, making it well-suited for problems like Frozen Lake where the environment's stochastic nature poses significant challenges. In this analysis, a constant epsilon (ϵ) value is employed, a strategy that balances the exploration-exploitation trade-off efficiently. This constant ϵ helps the algorithm maintain a consistent level of exploration throughout the learning process.

The analysis delves into the impact of key parameters such as the discount rate (γ), iterations, learning rate (alpha), and decay rates. The discount rate in Q-Learning, like in other algorithms, influences how future rewards are valued against immediate rewards. In the context of Frozen Lake, a higher discount rate encourages strategies that prioritize reaching the goal over immediate, but possibly risky, moves. The learning rate (alpha) determines the rate at which new information overrides old information, a critical factor in an environment where adapting to new paths is essential. Decay rates are employed to reduce exploration over time as the algorithm gains more knowledge about the environment, a strategy that becomes increasingly important as learning progresses.

The alignment of the policy generated by Q-Learning with those from Value and Policy Iteration, as shown in Figure 5, is a key finding. The rewards from Q-Learning are also similar to the Value and Policy Iteration. It demonstrates that despite their different operational mechanisms, all three algorithms are capable of effectively identifying the optimal strategy in the Frozen Lake environment. This consistency underscores the deterministic nature of the optimal solution within this stochastic setting. Figure 6, illustrating the reward-iteration relationship in Q-Learning, emphasizes the importance of iterations. The model's convergence after about 10^5 iterations indicates that ample iterations are vital for the algorithm to thoroughly explore and understand the Frozen Lake environment.

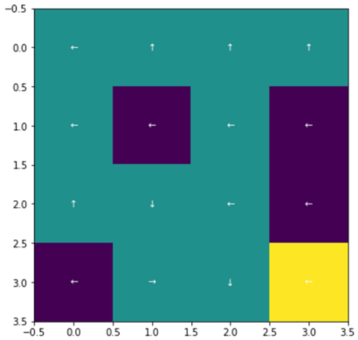
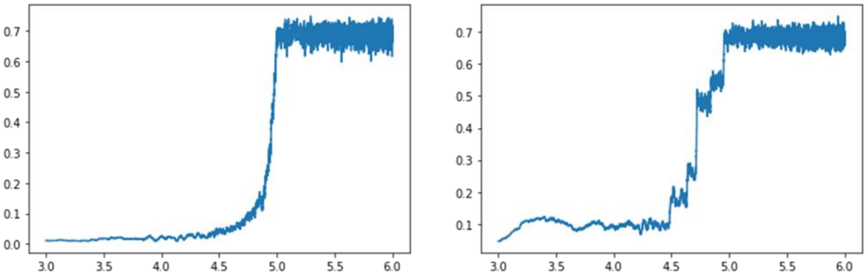


Figure 5. Q-Learner policy



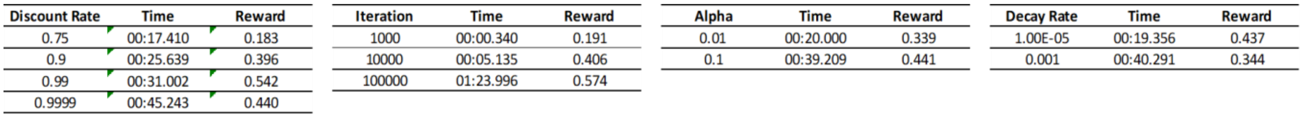
(a) Q-Learner.: Decay 1e-03

(b) Q-Learner.: Decay 1e-05

Figure 6. Reward vs. Iteration (log-scale) with Different Decay Values

Table 1 summarizes the parametric analysis results. The trend of lower iterations leading to quicker but less rewarding convergence is attributed to insufficient exploration, preventing the algorithm from discovering more rewarding paths. Similarly, a lower discount rate speeds up convergence by focusing on immediate rewards but at the cost of potentially higher future gains. A higher rate, conversely, places greater emphasis on future rewards, leading to more extensive exploration and higher eventual rewards, albeit with increased convergence time. The role of a higher alpha value in achieving greater rewards, despite requiring more time, lies in its capacity to rapidly assimilate new information, crucial in the dynamic Frozen Lake environment. Lastly, an increased decay rate, by prolonging the exploration phase, eventually leads to higher rewards, emphasizing the importance of balancing exploration and exploitation. In contrast, a lower decay rate can result in faster convergence but with the potential drawback of trapping the algorithm in local optima as shown in Figure 6 b, limiting its ability to discover the most effective overall strategies in the complex and dynamic environment of Frozen Lake.

Table 1. Frozen Lake 4 x 4 Q-Learning Parametric Analysis



Discount Rate	Time	Reward
0.75	00:17.410	0.183
0.9	00:25.639	0.396
0.99	00:31.002	0.542
0.9999	00:45.243	0.440

Iteration	Time	Reward
1000	00:00.340	0.191
10000	00:05.135	0.406
100000	01:23.996	0.574

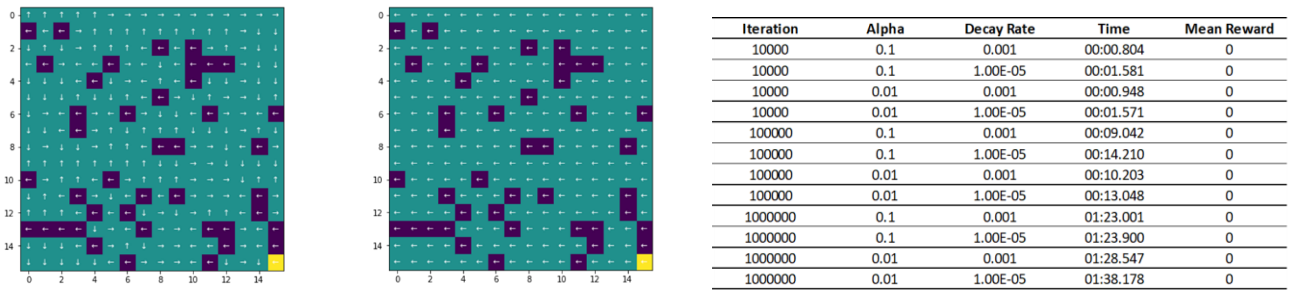
Alpha	Time	Reward
0.01	00:20.000	0.339
0.1	00:39.209	0.441

Decay Rate	Time	Reward
1.00E-05	00:19.356	0.437
0.001	00:40.291	0.344

2.3 Higher State (16 x 16)

The analysis of the Frozen Lake problem was extended to a more complex 16x16 state environment to understand the impact of increased state space. While the parametric results are broadly similar to those observed in the 4x4 state analysis, key differences emerged, particularly in the performance of Q-Learning. The 16x16 state environment, due to its increased complexity, resulted in generally lower rewards and longer times for convergence. This is primarily because the larger state space increases the number of potential paths, making it more challenging to identify the optimal strategy. Unlike in the 4x4 state environment, Q-Learning failed to converge in the 16x16 version, as evidenced in Figure 7. While Value Iteration and Policy Iteration successfully devised strategies, Q-Learning couldn't find a viable policy. This divergence in performance can be attributed to the increased complexity and the reward structure. The reward setup, identical to the 4x4 environment where only reaching the exit yields a reward of 1, proves inadequate for the 16x16 version. In such a large state space, this binary reward system fails to provide enough feedback for Q-Learning to effectively navigate the environment. For the 16x16 Frozen Lake problem, a more nuanced reward strategy is needed. Incremental rewards for reaching intermediate milestones or penalties for specific actions can provide more consistent feedback, aiding the learning process in complex environments. To improve Q-Learning's performance in the 16x16 environment, adjustments could include increasing the exploration phase, fine-tuning the learning rate and decay rate, and possibly integrating a more complex reward structure. These changes could help the algorithm better navigate the increased complexity.

The transition from a 4x4 to a 16x16 state environment in the Frozen Lake problem highlights that the larger state space in the 16x16 version significantly raises the problem's complexity, impacting both the performance metrics and the efficacy of the algorithms, particularly Q-Learning. This comparison underscores the importance of tailoring the learning approach and reward structure to the specific challenges posed by the environment's size and complexity.



Iteration	Alpha	Decay Rate	Time	Mean Reward
10000	0.1	0.001	00:00.804	0
10000	0.1	1.00E-05	00:01.581	0
10000	0.01	0.001	00:00.948	0
10000	0.01	1.00E-05	00:01.571	0
100000	0.1	0.001	00:09.042	0
100000	0.1	1.00E-05	00:14.210	0
100000	0.01	0.001	00:10.203	0
100000	0.01	1.00E-05	00:13.048	0
1000000	0.1	0.001	01:23.001	0
1000000	0.1	1.00E-05	01:23.900	0
1000000	0.01	0.001	01:28.547	0
1000000	0.01	1.00E-05	01:38.178	0

(a) V.I. and P.I. Policy

(b) Q-Learner Policy

(c) Q-Learner Performance

Figure 7. Frozen Lake 16 x 16 Analysis

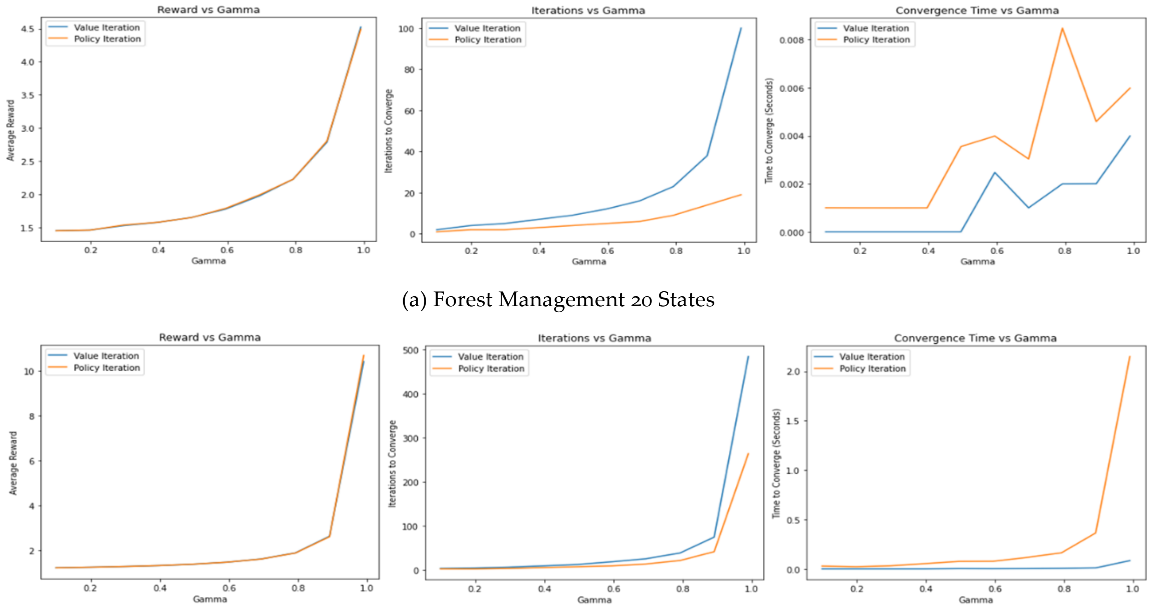
3 FOREST MANAGEMENT

The analysis of the Forest Management problem focuses on two distinct scenarios: one with 20 states and another with 500 states. For the 20 states scenario, the problem parameters are set as follows: R1 (immediate reward for cutting the forest) = 10, R2 (reward for waiting) = 6, and P (probability of forest growth) = 0.1. In the 500 states scenario, the parameters are adjusted to R1 = 100, R2 = 15, and P = 0.01. The necessity for different reward setups in the 20 and 500 states scenarios stems from the need to reflect the increased complexity and decision-making scope in larger state spaces. In larger environments, such as the 500 states scenario, the potential for future rewards is greater, warranting a higher reward structure (R1 and R2). Additionally, the lower probability of forest growth (P) in the 500 states scenario models the increased uncertainty and risk associated with managing a larger forest, making the decision-making process more intricate. The Forest Management problem in both scenarios was analyzed using Value Iteration, Policy Iteration, and Q-Learning. A consistent convergence strategy as Frozen Lake problem was employed across all algorithms, utilizing a constant epsilon (ϵ) value.

3.1 Value and Policy Iteration on Forest Management

In the analysis of the Forest Management problem, the discount rate, represented by gamma (γ), emerges as a critical parameter. γ dictates how future rewards are valued against immediate rewards. In the context of forest management, where decisions involve a trade-off between immediate logging profits and potential future gains from forest growth, γ plays a crucial role. A higher γ value suggests a greater emphasis on long-term rewards, aligning with sustainable forest management practices that prioritize future growth over immediate harvesting.

As illustrated in Figure 8, the analysis for both Value Iteration and Policy Iteration reveals a trend similar to the Frozen Lake problem. Both algorithms show the same rewards over varying γ values. However, Value Iteration requires more iterations to converge but less time overall, whereas Policy Iteration converges with fewer iterations but takes more time. This similarity in trends between the Frozen Lake and Forest Management problems can be attributed to the fundamental mechanics of the algorithms. Value Iteration's iterative approach, though requiring more iterations, involves simpler computations per iteration, making it quicker overall. Policy Iteration, with its complex policy evaluation steps, converges in fewer iterations but takes longer due to the increased computational complexity in each iteration.



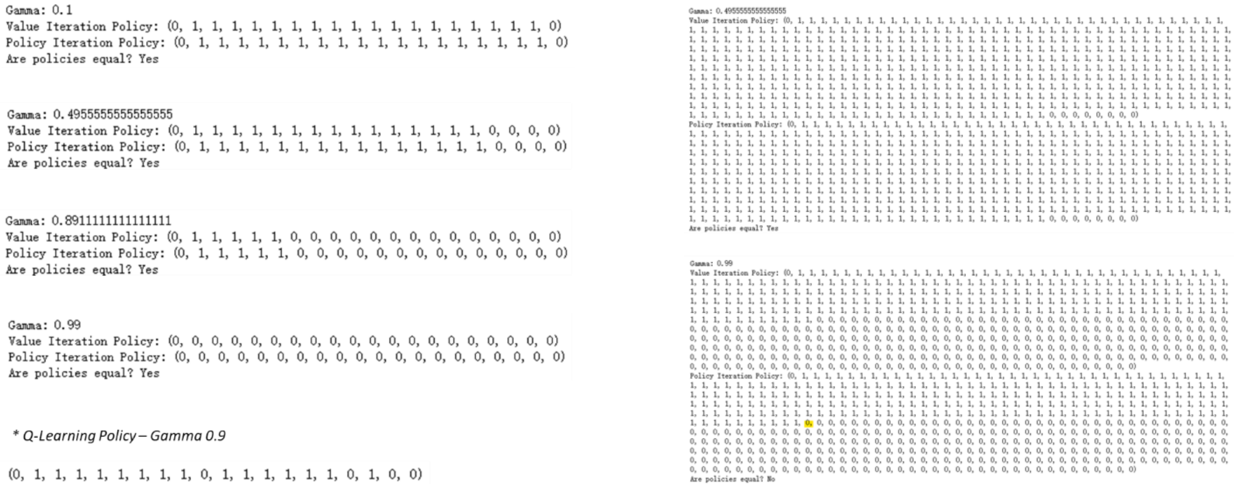
(a) Forest Management 20 States

(b) Forest Management 500 States

Figure 8. Forest Management Value and Policy Iteration Analysis

The identical policy outcomes over different γ values for both analyses were observed, which can be explained by the deterministic nature of the optimal policy in such environments. Both algorithms are effective in identifying the best actions (cut or wait) at each state to maximize rewards, considering the given γ value. Figure 9 also demonstrates how policies shift from more cutting (1) to more waiting (0) as γ increases. For lower γ values, immediate rewards from cutting (1) are prioritized, while higher γ values lead to strategies that favor waiting (0), accounting for the potential of future rewards. This shift reflects the balance between short-term gains and long-term sustainability in forest management. The analysis reveals that managing the 500 states problem requires significantly more time compared to the 20 states scenario. This increase in time is due to the larger state space in the 500 states problem, which adds complexity and requires more extensive exploration to identify the optimal policy.

In summary, the analysis of Value Iteration and Policy Iteration in the Forest Management problem highlights the influence of γ in shaping optimal policies. It underscores the algorithms' ability to adapt to the scale of the problem and the importance of considering future rewards in decision-making processes, particularly in scenarios involving resource management and sustainability.



```
Gamma: 0.1
Value Iteration Policy: (0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0)
Policy Iteration Policy: (0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0)
Are policies equal? Yes


Gamma: 0.4955555555555555
Value Iteration Policy: (0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0)
Policy Iteration Policy: (0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0)
Are policies equal? Yes


Gamma: 0.8911111111111111
Value Iteration Policy: (0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
Policy Iteration Policy: (0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
Are policies equal? Yes


Gamma: 0.99
Value Iteration Policy: (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
Policy Iteration Policy: (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
Are policies equal? Yes


 * Q-Learning Policy – Gamma 0.9

(0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0)
```

(a) 20 States Policy

```
Gamma: 0.4955555555555555
Value Iteration Policy: (0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0)
Policy Iteration Policy: (0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0)
Are policies equal? Yes


Gamma: 0.99
Value Iteration Policy: (0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
Policy Iteration Policy: (0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
Are policies equal? No
```

(b) 500 States Policy

Figure 9. Optimum Policies from Different Algorithms

3.2 Q-Learning on Forest Management

The Q-Learning analysis for Forest Management involved several key parameters, each playing a distinct role in the learning process:

- Iterations: Determines how many times the algorithm updates its knowledge base. Higher iterations allow for more comprehensive exploration and learning from the environment.

- Epsilon (ϵ): Balances exploration (trying new actions) with exploitation (relying on known information). Crucial for avoiding local optima and ensuring diverse strategy testing.

- Epsilon Decay: Controls the rate at which ϵ decreases, shifting the strategy from exploration towards exploitation as learning progresses.

- Alpha Decay: Influences how quickly the learning rate (alpha) decreases, impacting the incorporation of new information over time.

- Alpha Min: Sets the minimum value for alpha, ensuring the algorithm continues to learn, albeit at a reduced rate.

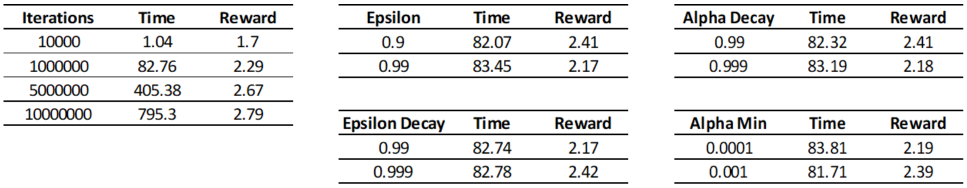
Table 2 summarizes the Q-Learning results with the various parameters. Similar to the Frozen Lake problem, iterations are crucial for achieving higher rewards in Forest Management. A higher number of iterations allows for better exploration of the decision space, but also significantly increases the time required. Notably, even at 10^7 iterations, the rewards reached a maximum of only 2.8, significantly lower than the over 10 rewards observed in Value and Policy Iteration. This discrepancy could be due to Q-Learning's model-free nature, which might struggle in complex environments like the 500-state Forest Management scenario, where understanding long-term consequences of actions is vital. Enhancing Q-Learning's performance in this context could involve fine-tuning the balance between exploration and exploitation, adjusting learning rates, and possibly integrating more nuanced reward structures to better guide the learning process.

The observation that higher epsilon decay leads to more rewards without increasing time significantly can be attributed to the algorithm quickly moving from a broad exploration phase to efficiently exploiting the gathered knowledge, thus arriving at effective strategies in a stable manner. Lower alpha decay leading to more rewards suggests that a slower reduction in the learning rate allows the algorithm to continually adapt its strategy based on new information, which is beneficial in the dynamic environment of Forest Management.

One of the key findings to note is that Q-Learning gives different policy when compared to the Value and Policy Iterations (Figure 9 a). The difference in the policy generated by Q-Learning compared to Value and Policy Iteration could stem from its different approach to learning. Q-Learning, being model-free, derives its policy solely based on rewards received and actions taken, without an underlying model of the environment. This can lead to different strategic preferences, especially in a complex setting like the 500-state Forest Management, where the long-term implications of actions are not immediately apparent.

In summary, the Q-Learning analysis in the context of Forest Management highlights the importance of parameter tuning and the challenges inherent in applying model-free approaches to complex, state-rich environments. The divergence in policies underscores the need for a nuanced understanding of the environment and the potential benefits of combining different learning approaches for optimal strategy development.

Table 2. Forest Management 500 States Q-Learning Parametric Analysis



Iterations	Time	Reward
10000	1.04	1.7
1000000	82.76	2.29
5000000	405.38	2.67
10000000	795.3	2.79

Epsilon	Time	Reward
0.9	82.07	2.41
0.99	83.45	2.17

Epsilon Decay	Time	Reward
0.99	82.74	2.17
0.999	82.78	2.42

Alpha Decay	Time	Reward
0.99	82.32	2.41
0.999	83.19	2.18

Alpha Min	Time	Reward
0.0001	83.81	2.19
0.001	81.71	2.39

4 SUMMARY AND CONCLUSION

This analysis compared Value Iteration, Policy Iteration, and Q-Learning across two distinct environments: the Frozen Lake and Forest Management problems.

For **Frozen Lake**, Value and Policy Iteration converged effectively to the same policy, with Value Iteration being faster but requiring more iterations, and Policy Iteration taking longer but with fewer iterations. Q-Learning aligned well in simpler scenarios like the 4x4 grid but struggled in the more complex 16x16 grid. In **Forest Management**, Value and Policy Iteration again showed consistency in policy outcomes, adeptly balancing immediate and future rewards. However, Q-Learning encountered difficulties in the 500-state scenario, failing to converge effectively, likely due to its model-free approach and the limitations of the reward structure in complex environments.

The analysis suggests that **Value Iteration** is better suited for scenarios requiring quick convergence in simpler or smaller state spaces, whereas **Policy Iteration** is preferable in contexts where fewer iterations are needed, despite potentially higher computational complexity per iteration. **Q-Learning** shows promise in less complex environments but struggles with scalability in larger state spaces or in scenarios where long-term strategic planning is crucial. For Q-Learning, integrating more nuanced reward structures or combining model-based and model-free elements could improve outcomes. Fine-tuning key parameters such as discount rates, learning rates, and decay rates is essential for all algorithms, especially in complex environments. Addressing scalability issues in Q-Learning might involve incorporating advanced techniques like function approximation or deep learning. This comparative analysis underscores the importance of selecting the right algorithm for a given problem and the potential benefits of customizing and combining approaches for complex, large-scale decision-making scenarios.