




# Markov Decision Processes

Abdullatif Jarkas

Georgia Institute of Technology

Atlanta, Georgia, USA

@gatech.edu

## I. INTRODUCTION

Markov Decision Process (MDP) is an imperative tool in reinforcement learning and decision making where an agent explores and exploits an environment in order to maximize it's potential reward. This is different from supervised and unsupervised learning as we've previously experimented with as reinforcement learning is a more trial and error technique which takes into account states, utility, and reward and learns through previously acquired interactions and data.

In this experiment, we will be exploring two separate MDP problems of contrasting state sizes, BlackJack and CartPole. We will also be explicitly experimenting and assessing small and large CartPole problems, as suggested by TA Jake in the FAQs, to hopefully uncover unique results and characteristics for an identical problem of differing state space sizes. And then finally we will be running our reinforcement learning algorithm, Q-Learning, on both problems and comparing results. Overall, we will be assessing Value Iteration (VI), Policy Iteration (PI), and Q-Learning for each MDP problem comparatively.

## II. PROBLEM SELECTION

As mentioned, we selected two distinct MDP problems with contrasting state space size, black jack and cart pole. We will use both the default state size of Cart Pole and a reduced state size version for comparative analysis isolating state sizes.

### A. BlackJack (Small MDP)

- **Description:** A card game where the ideal state is to reach a sum of 21 of cards in your hand without going over. However, a high near 21 will yield a high likelihood of winning against the competing hands.
- **Reward:** +1 if a player wins, 0 if draw, -1 if loss.
- **State Space:** 290 states = 29 Player Hands \* 10 Dealer Hands.
- **Why is it Interesting:** Straight forward game with discrete sparse outcomes and stochastic random variables. Interesting to compare against a contrasting MDP problem that is continuous, dense, and deterministic in nature.
- **Summary:** Small, Discrete, Sparse, Stochastic

### B. CartPole (Large MDP)

- **Description:** The objective here to maintain a pole balanced on top of a cart while maintaining continuous variables such as position and velocity of the cart, and the angle and angular velocity of the pole

- **Reward:** +1 for every timestep (second) the pole is balanced upright on top of the cart.
- **State Space:** Continuous, however when discretized: 23,000 states = 10 position bins \* 10 velocity bins \* 23 angle bins \* 10 angular velocity bins.
- **Why is it Interesting:** Much more complex problem with dense rewards. Is inherently a continuous problem with deterministic dynamics. Interesting to compare against a sparse, discrete, and stochastic problem like BlackJack. Bins can be adjusted here to reduce or increase discretized state size, which is imperative for our small vs. large state space comparative analysis.
- **Summary:** Large, Continuous (can be Discretized), Dense, Deterministic

### C. CartPole Small (Reduced State Size)

- **Description:** Same as CartPole original.
- **Reward:** Same as CartPole original.
- **State Space:** Continuous, however when discretized with 50% smaller number of bins than our original CartPole, we now get 2875 states = 5 position bins \* 5 velocity bins \* 23 angle bins \* 5 angular velocity bins.
- **Why is it Interesting:** Intended for State Size analysis. Comparing against the original larger CartPole, we can investigate and uncover any profound findings purely based on state size as our reduced CartPole remains generally identical in reward and objective structure.
- **Summary:** Smaller CartPole Problem, Continuous (can be Discretized), Dense, Deterministic

## III. METHODOLOGY

### A. Experiment 1

Here we compare BlackJack and CartPole through both value iteration (VI) and policy iteration (PI) and interpret our findings. We will also do the same experiments with CartPole (Original) and CartPole (Reduced) for state space analysis.

1) *Value Iteration:* Value iteration updates the value of each state using the Bellman equation which takes into account the maximum expected utility over all plausible actions. It computes value each state at each iteration.

*Bellman equation in value iteration:*

$$V(s) = \max_{a \in A} \left[ R(s, a) + \gamma \sum_{s' \in S} P(s' | s, a) V(s') \right] \quad (1)$$

2) *Policy Iteration*: Unlike *Value iteration*, at each step of *Policy iteration* we evaluate the current policy across all states. After that, we update the current policy with the action that results in the highest expected value.

*Bellman equation in policy iteration*:

$$V^\pi(s) = R(s, \pi(s)) + \gamma \sum_{s' \in S} P(s' | s, \pi(s)) V^\pi(s') \quad (2)$$

*Policy update rule*:

$$\pi_{\text{new}}(s) = \arg \max_{a \in A} \left[ R(s, a) + \gamma \sum_{s' \in S} P(s' | s, a) V^\pi(s') \right] \quad (3)$$

## B. Experiment 2

Finally, we apply Q-learning (our reinforcement learning algorithm) on both MDP problems, BlackJack and CartPole, as well as CartPole Reduced for further analysis.

## IV. HYPOTHESES

### A. Experiment 1

1) *BlackJack vs. CartPole*: I hypothesize that BlackJack, given its sparse, small, and discrete state space will outperform CartPole in convergence iteration steps for both value and policy iteration. However, because BlackJack is inherently a stochastic problem with plausible negative reward and CartPole only having dense positive rewards, I hypothesize that our mean value over iterations using yielded from both value and policy iteration for BlackJack will be significantly lower than that of CartPole.

2) *Value Iteration vs. Policy Iteration*: Given that *Policy Iteration*'s inherent structure is to iteratively compute the value for all states for a given policy, and given that *Value Iteration* iteratively computes the value at each state relatively trivially through the Bellman equation without policy evaluation, I hypothesize that Value Iteration will overall significantly be more computationally efficient than Policy Iteration, however convergence will require much more iterations. Hence, Policy iteration will converge at a shorter iteration step.

3) *CartPole vs. CartPole Reduced (State Space Analysis)*: I hypothesize that CartPole reduced will converge much faster than CartPole original given its much smaller discretized state space. However, given that they are essentially the same problem, I hypothesize that they will share identical characteristics in convergence and mean value over iterations, however at a much faster and lower scale.

### B. Experiment 2

I hypothesize that Q-Learning will overall perform best on CartPole over BlackJack policy wise, because BlackJack's inherent rewards structure are stochastically attributed given that it is a randomly sorted card game. CartPole's rewards are not randomized, yet they are deterministic and can be learned hence my hypothesis favoring CartPole. However, given BlackJack's trivial state size, I hypothesize convergence will be much faster on the BlackJack problem over the CartPole problem.

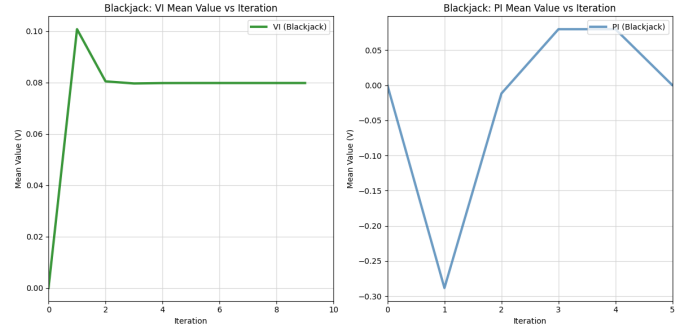


Fig. 1. BlackJack: Mean Value over Iteration (Value Iteration **green** vs. Policy Iteration **blue**)

Comparing CartPole to CartPole Small, I hypothesize that CartPole Small will be much more computationally efficient, however suffer in performance given it's smaller state space to learn from.

## V. EXPERIMENTATION

For all experiments using both value and policy iteration on all MDP problems, we will set our discount factor  $\gamma$  to 0.99. The reason for this is to remain relative across all problems for comparative analysis while also allowing convergence to occur in a presentable manner.

### Experiment 1

#### A. BlackJack vs. CartPole

1) *Convergence*: Here we will assess at how many iterations do we find convergence for both value iteration and policy iteration across both MDP problems. We will be using mean value over iterations, and delta convergence. Mean value is the the average value of all states at a given iteration. Here we can find convergence and also the mean value. However, to better visualize and detect convergence, we will also use a delta convergence plot which takes the difference in mean value over each iteration, we will define convergence at any iteration where the values made  $\geq 0.001$  improvement compared to the previous iteration.

**BlackJack (Small MDP)** As shown in *Figure 1*, we see Mean value over iterations for value iteration (green), and policy iteration (blue) for the BlackJack problem. Quickly we see value iteration reach a global high of 0.10 mean value at just 1 iteration, then soon after converges to 0.08 mean value at 2 iterations. The spike at iteration 1 reflects the transition from the initial values (all zeros or random) to the first meaningful estimate  $V$ . The convergence soon after at iteration 2 indicates that our problem is quite trivial, which makes sense as this is our small MDP with a discrete number states.

In comparison to BlackJack policy iteration, we see that Policy iteration actually went negative at 1 iteration, this is due to our policy iteration initially selecting a random policy to begin with, which is likely to be suboptimal. However, over iterations we eventually see our policy iteration reach the same convergence of 0.08 mean value at iteration 3. Indicating

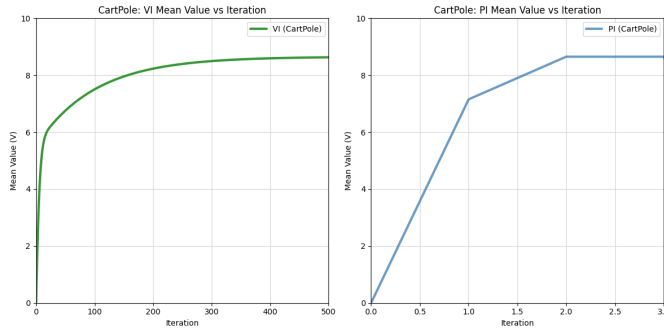


Fig. 2. CartPole: Mean Value over Iteration (Value Iteration **green** vs. Policy Iteration **blue**)

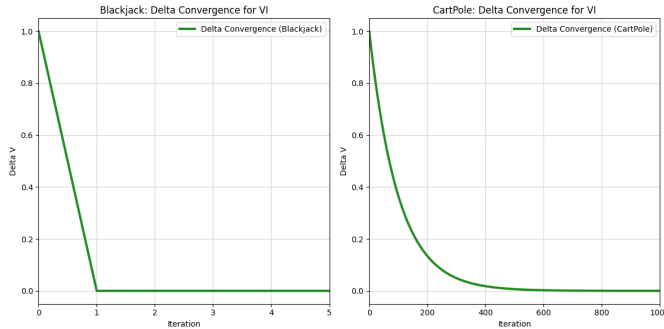


Fig. 3. Delta Convergence for Value Iteration (BlackJack vs. CartPole)

similar results, however achieving this at a much later iteration. But, keep in mind, one value iteration is not equivalent to one policy iteration as we will explore in our experimentation later in this analysis.

**CartPole (Large MDP)** Now looking at *Figure 2*, we see contrasting results to our previous experimentation with BlackJack. Remember that our discretized state size of CartPole is nearly 80 times larger than that of BlackJack. Look at the charts starting with value iteration **green** we see completely different shape, this one appearing similar to a natural log graph will converging at the maximum mean value of 8.65 at iteration 688, which is much larger than that of BlackJack. However, we should also look at the scale of this chart. We reached convergence nearly 700 iterations into our experiment. This indicates that as our value iteration is progressing, it is continuously updating our state values for the better through the Bellman equation previously stated. These contrasting results to our BlackJack value iteration are likely due to the fact that one, our problem here in nature is continuous and not discrete, and two, our discretized state size is much larger than BlackJack (23,000 vs. 290).

Now looking at the same *Figure 2*, specifically the policy iteration **blue**, we see nearly identical results in terms of convergence at 2 iterations as we saw in BlackJack policy iteration, however unlike the BlackJack policy iteration, we did not begin with any negative mean value. A naive's perspective is that our policy iteration chose a relatively optimal policy at random, however this is likely not the case. Looking at the

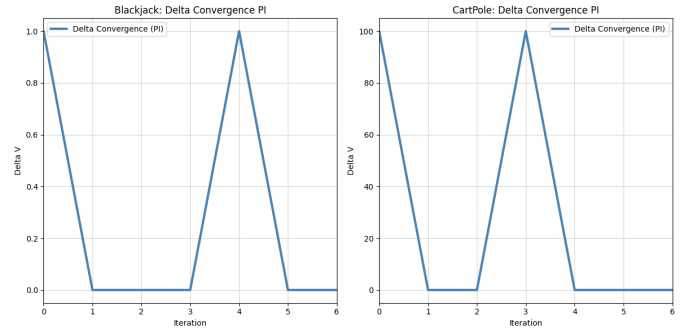


Fig. 4. Delta Convergence for Policy Iteration (BlackJack vs. CartPole)

nature of cart pole, we see that our reward structure is always positive, as it is the number of timesteps a pole is balanced on top of a cart, as opposed to BlackJack which can result in negative rewards when at a state of a loss. Going back to convergence, our policy did reach an identical convergence (8.49 mean value) to our value iteration, however at a much lower iteration count. However, we must take into account the computation expense of each iteration per value and policy iteration to make conclusive meaning of this.

**Mean Value Analysis** As revealed in our convergence of mean values per problem, we see CartPole converge at a much higher mean value of 8.49 in comparison to BlackJack at 0.08. This is partially due to the fact of how we calculate our mean value and the inherent structure of our MDP problems. Given that BlackJack rewards are -1, 0, and 1, the game is stochastic and based on a randomly sorted deck of cards and that rewards are not cumulative, we should expect a mean value in between -1 and 1 per iteration. However, given that our CartPole problem is not random based and only yields positive rewards for the number of timesteps the pole is balanced on top of the cart, we can expect cumulative positive rewards, hence our convergence at 8.49 mean value.

**Value and Policy Iteration Analysis** As shown in the Delta convergence plots *Figure 3* and *Figure 4*, we find repeating results to our mean value analysis prior. Within value iteration over the two MDP problems, we see a direct and short convergence with Black Jack and a more gradual elongated convergence with CartPole. As previously analyzed, this is likely because BlackJack is has a small state space with discrete values, while CartPole has a state space much larger with continuous values.

When looking at Policy iteration's delta convergence across both MDP problems we find similar results, however with CartPole converging at a lower iteration count (4 iterations) than BlackJack (5 iterations). Again, this is likely due to BlackJacks inherent problem dealing with negative rewards and randomized features.

Now, comparing value iteration to policy iteration for both MDPs, we find identical findings to our previous analysis due to the same reasons, so we'll briefly brush on that. Value iteration overall showed convergence at much later iterations when compared to policy iteration for the larger MDP CartPole,

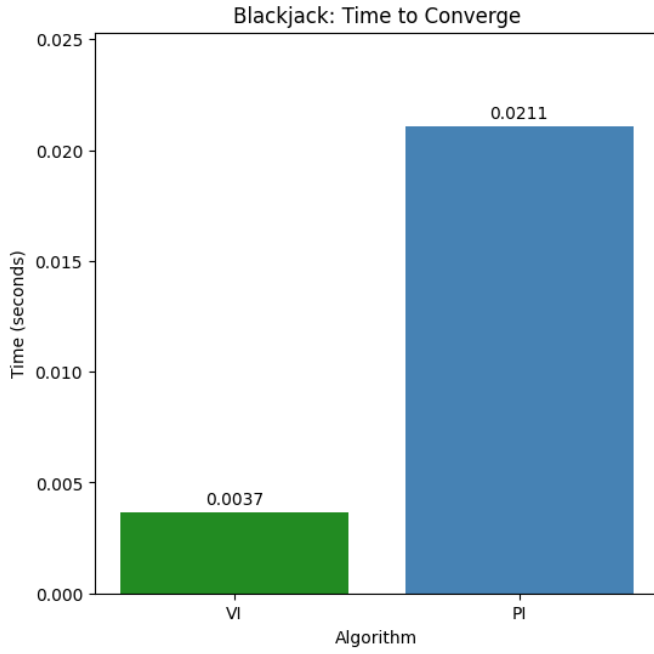


Fig. 5. BlackJack: Time to converge (in seconds)

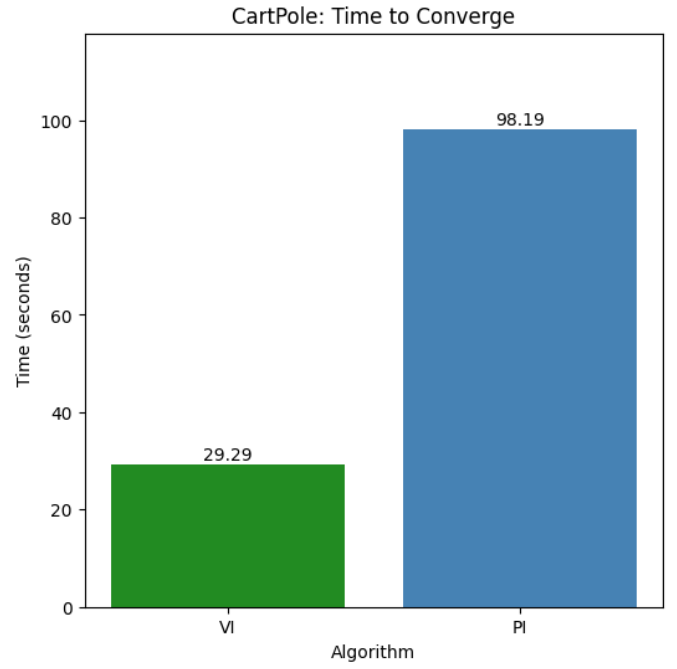


Fig. 6. CartPole: Time to converge (in seconds)

while converging at earlier iterations for our smaller MDP BlackJack. Again for the reasons previously stated.

#### Convergence: Conclusion and Hypothesis Evaluation

Overall, we saw convergence at a much faster iteration step in the BlackJack problem given it's sparse and trivial structure, validating our hypothesis that BlackJack will converge in shorter iterations. When comparing mean value convergence for both problems, we find that value iteration and policy iteration achieved identical results, however BlackJack achieved a far lower mean value than CartPole given it's inherent problem structure, also validating our hypothesis. Also, when comparing Value Iteration and Policy Iteration, we see Policy Iteration convergence occur at much lower iterations for CartPole, however, contrastingly, slightly more iterations than value iteration for BlackJack, revealing contrasting results hence not validating our hypothesis that Policy Iteration will always converge at a smaller iteration count. But again, we must understand that each iteration for each method is not necessarily comparable as shown in our next experiment.

2) *Computation Expense:* We will now assess the computation expense for value and policy iteration across both MDP problems. To do this, we will compare computation time to converge for each in seconds.

**BlackJack** Now looking at BlackJack's time to converge in seconds (Figure 5), we find that Value Iteration converged in just 0.0037 seconds, while Policy Iteration converged in 0.0211 seconds. Revealing that Value Iteration here is nearly 5.7 times faster than Policy Iteration. Although significant in difference, there is no practical difference between the two as they are both unnoticeably different in real world scenarios. So far we find that for small discrete MDPs such

as BlackJack, either Value Iteration or Policy Iteration are not practically significantly different in the real world, however from a technical perspective, Value Iteration clearly shows to be significantly more cost efficient.

**CartPole** Figure 6 shows the time to converge for CartPole. Immediately we see that CartPole significantly shows to be much more computationally expensive with Value Iteration converging at 29.29 seconds, and Policy Iteration converging at 98.19 seconds. This finding is understandable given that CartPole innately is a continuous problems that when discretized using bins, it is still nearly 80 times larger than BlackJack in state space. Like in BlackJack's time to converge, we see a similar relationship between VI and PI, with VI being nearly 3.4 times as efficient as Policy Iteration. Unlike BlackJack, these findings are both statistically significant and practically significant, meaning that even in real world practices there is a substantial advantage or disadvantage selecting between the two. This finding in both BlackJack and CartPole is due to Policy Iterations intensive calculations in Policy Evaluation for all states per iteration in to Value Iteration's more trivial calculations based iteratively adjusting state values as we go. Meaning that although Value Iteration took hundreds more iterations to converge over Policy Iteration in CartPole, it was substantially more cost efficient.

**Computation Expense: Conclusion and Hypothesis Evaluation** Overall, for both MDP problems we found that Value Iteration is significantly more cost efficient than Policy Iteration despite taking much greater number of iterations for CartPole to converge. Validating our hypothesis that Policy Iteration may converge at fewer iterations, however at a much larger overall computation cost compared to convergence

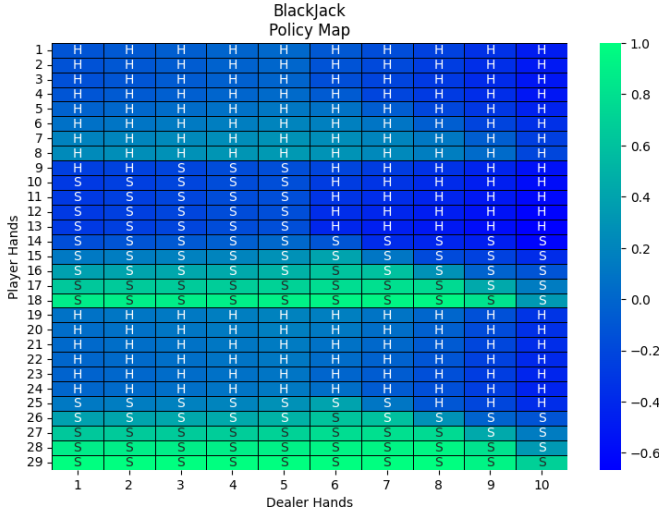


Fig. 7. BlackJack Policy Heat Map ( $H$  = Hit,  $S$  = Stay)

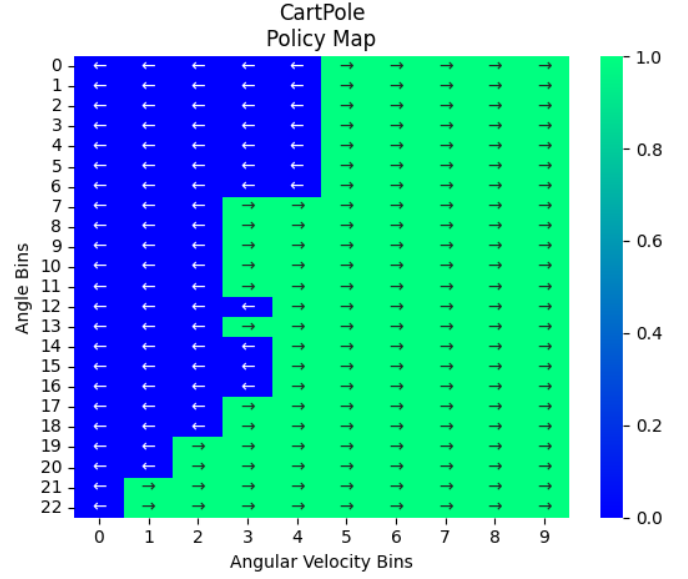


Fig. 8. CartPole Policy Map

under Value Iteration.

3) **Policy Mapping: BlackJack** The policy map, Figure 7, was generated by both value and policy iteration as they both yielded identical policies. They likely converged on identical policies due to the fact that BlackJack is trivial, small state space, and sparse reward problem. Visualized in the plot is the action on whether to hit ( $H$ ) or stay ( $S$ ) at each state (player's hand vs. dealer's hand). As shown in the heatmap, the policy reveals that the best action is to "stay" at higher player hand states (around 16 and above) and "hit" at lower hand states. Which aligns well with BlackJack's conventional practice. The strong stand at 29 likely results from the model counting an Ace as 11 rather than 1 ( $18 + 11 = 29$ ). While anything over 21 are busts in the game, this policy map reveals the strength of having an Ace card in your hand.

**CartPole** Now moving on top CartPole's converged policy map in Figure 8. Unlike BlackJack a  $29 \times 10$  innate state size which is easily plottable in 2D, CartPole's innate state size is  $10 \times 10 \times 23 \times 10$  corresponding to *position bins  $\times$  velocity bins  $\times$  angle bins  $\times$  angular velocity bins* which is not plottable as it is 4D. To reduce it to 2D, we kept the most impactful variables in the CartPole game (angle bins and angular velocity bins) and fixed the remaining variables at a constant set of 5 arbitrarily. By doing this we are now able to visually plot our policy map.

Now analyzing the policy map, it suggests moving left (**blue**) for majority states where the angle is significantly tilted or angular velocity is quite low. This is intuitive as it's purpose is to counteract the pole's falling direction. In contrast, moving right (**green**) is preferred when the angle is near vertical or when angular velocity gets larger. The decision boundary where the policy shifts actions from left to right indicates the learned trade-off between stabilizing the pole's angle and counteracting velocity changes, which is inherently the problem of the game.

**Policy Mapping: Conclusion** Overall when comparing the two policy maps, we find contrasting characteristics which

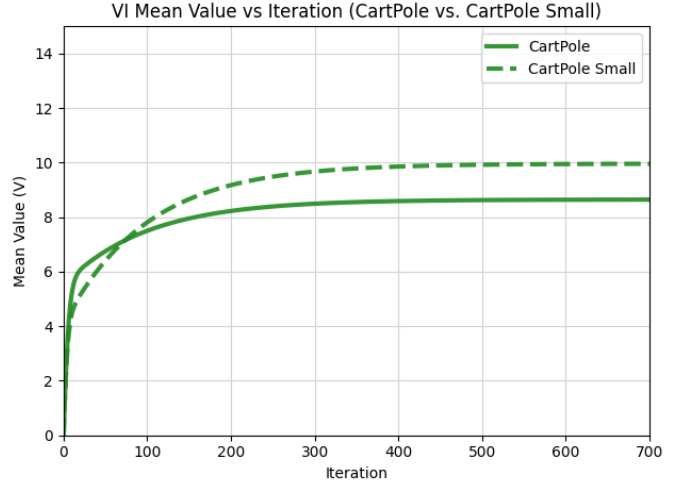


Fig. 9. VI Mean Value over Iterations (CartPole vs. CartPole Small)

are due to the inherent MDP problems and their unique game structure. BlackJack's policy reveals preferable state levels of player vs. dealer hands dependent on stochastic and probabilistic reasons, which is not a guarantee, While CartPole deterministically reveals a decision boundary on whether to move left or right depending on angle and angle velocity. Overall, different policy maps due to different MDP problem structures and objectives.

#### B. State Space Analysis (CartPole vs. CartPole Small)

Given that our original CartPole MDP problem has 23,000 total discretized states, our intention here is to compare the identical problem to itself, however at a lower state space to uncover any findings purely on state space differences. Our reduced CartPole Small's state space is 2,875.

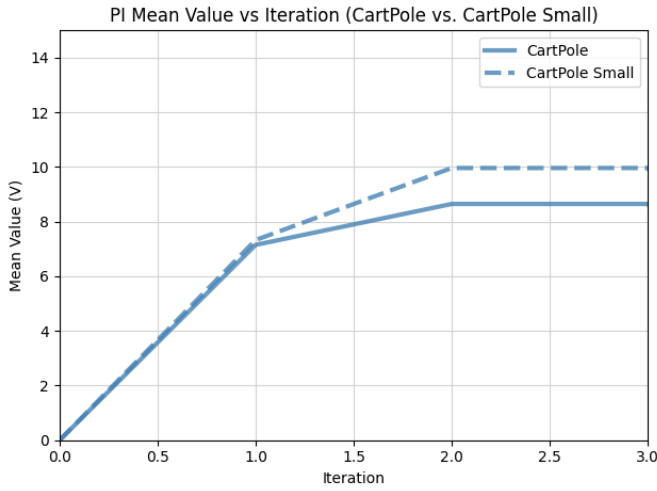


Fig. 10. PI Mean Value over Iterations (CartPole vs. CartPole Small)

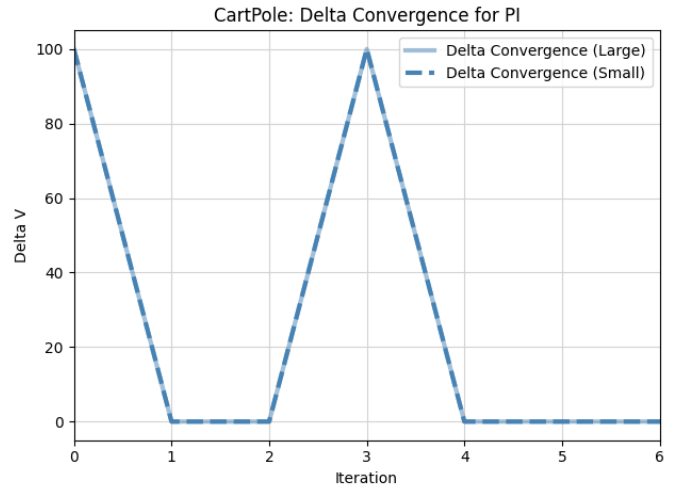


Fig. 12. PI Delta Convergence (CartPole vs. CartPole Small)

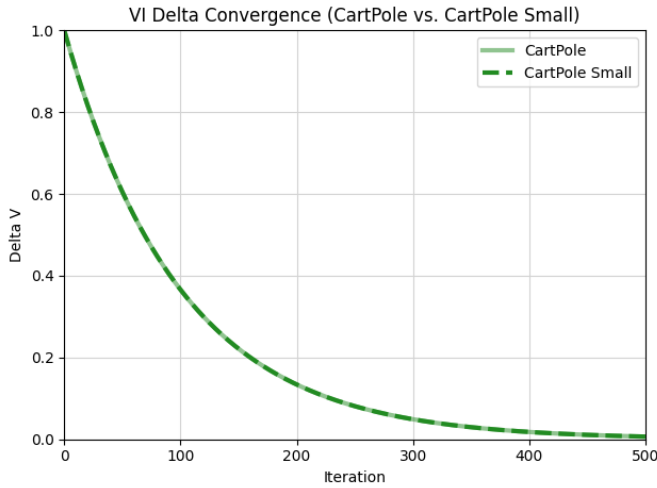


Fig. 11. VI Delta Convergence (CartPole vs. CartPole Small)

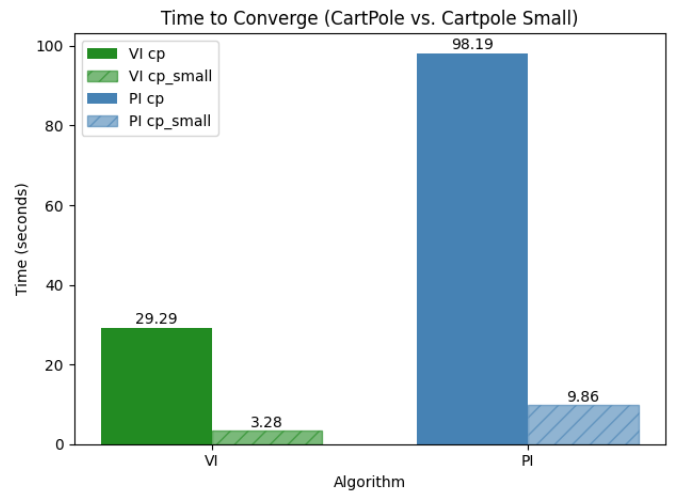


Fig. 13. Time to Converge (seconds) for VI and PI (CartPole vs. CartPole Small)

Now looking at the Mean Value over Iterations for both Value Iteration and Policy iteration (*Figures 9 and 10*), we find convergence of both CartPole and CartPole Small converging at identical iteration steps (688 iterations for VI, and 2 iterations for PI). However, surprisingly CartPole Small converged at a much higher mean value of 9.964 compared to 8.65 mean value for the original CartPole. This indicates that MDP problems with smaller state sizes, both value and policy iteration are able to converge at a much more effective value. Indicating CartPole Small's superiority in terms of policy formation.

Now looking at our Delta convergence comparing both CartPole and CartPole Small for both VI and PI (*Figures 11 and 12*), we find that both problems converge to their policies identically over their respective iterations. This indicates, that iteration over iteration, the same progress in policy is achieved across both MDP problems. However, this does not indicate they are equally computationally efficient.

Finally, assessing computation efficiency. In *Figure 13*, we find that our small CartPole problem took nearly 10% of the time to converge compared to the original CartPole problem for both VI and PI. This makes sense as we did decrease our bins by 50% which then decreased our total state space to nearly 10% of our original problem. Revealing that time to converge is directly related to state size and both the time to converge and state space size difference were 10%. Across both problems, PI was significantly (nearly 3 times) more computationally expensive when compared to VI, reinforcing our previous conclusion that this is due to PI's inherent algorithm which is based on policy evaluation.

**State Space Analysis: Conclusion and Hypothesis Evaluation** Overall CartPole Small did converge on a higher mean value when compared to its original, disproving our hypothesis that stated otherwise. CartPole Small did converge at faster time step, however equally from an iterations perspective



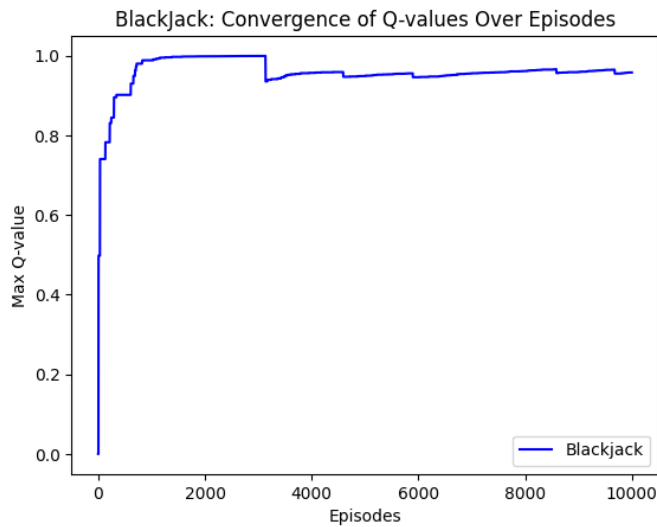


Fig. 14. Max Q-Value over Episode (BlackJack)

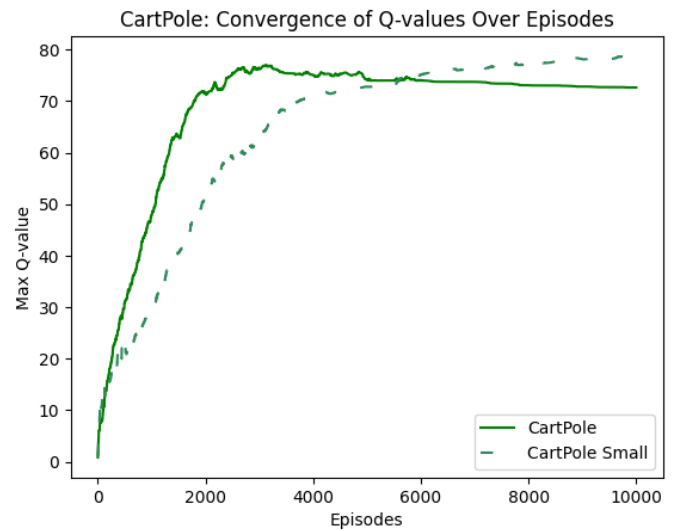


Fig. 15. Max Q-Value over Episode (CartPole, CartPole Small)

validating half of our hypothesis relating to that and disproving the other half. CartPole Small, being 10% the state space size of CartPole original, also took 10% the amount of time to converge in comparison to CartPole original, indicating that time to converge is directly related to state space sizing.

## VI. REINFORCEMENT LEARNING: Q-LEARNING

Here we will be applying Q-Learning and further analyzing, convergence, rewards, and computation expense. Here we defined convergence as 0.5% below it's maximum Q-value as we want to capture relative convergence and not exact maximums.

Our exploration and exploitation strategy uses an epsilon-greedy approach which starts with a high exploration rate,  $\epsilon = 1.0$ , and an epsilon decay rate of 0.9, indicating strong early exploration and gradually shifting the agent toward exploiting learned actions. However, we set a minimum  $\epsilon = 0.1$ , meaning in all cases we will still have some level of exploration. Finally, we have a decaying learning rate starting with 0.5 and decaying till 0.01, which slows down updates as episodes go on.

Looking at the Max Q-value over episode for the BlackJack MDP problem (Figure 14), we find that our Q-Learner quickly converged to 0.995 Q-value at episode 1260. This indicates that our problem here is relatively trivial, which makes sense as our BlackJack problem is small, sparse, and discrete problem with only 290 states. Indicating that our Q-Learner is well adept to problems like BlackJack. Now, looking at the CartPole plot (Figure 15), we find some counterintuitive and surprising results. We find that our original CartPole MDP converged quickly at episode 2702 with a Q-value of 76.59, which was much earlier than our CartPole small converging at episode 8659 with a substantially larger Q-Value of 81.04. This is counterintuitive because you'd expect the smaller problem to converge faster, with a lower local Q-value, however the opposite was the case here. This is likely due to our larger

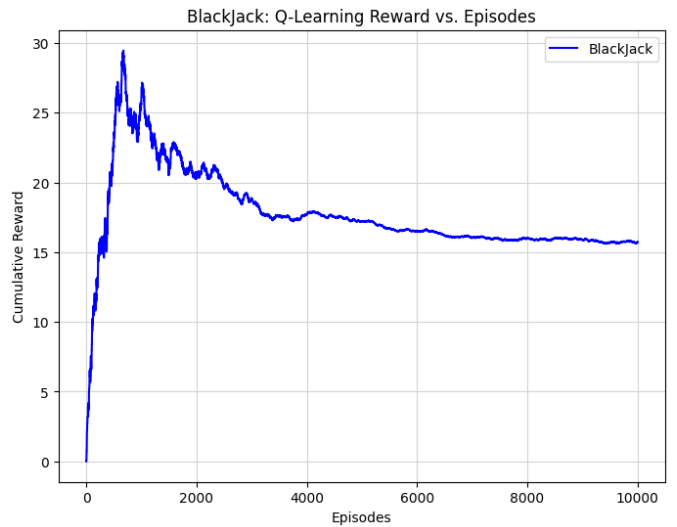


Fig. 16. Q-Learning Cumulative Reward over Episode (BlackJack)

state space offering more exploration opportunities, while our smaller CartPole problem may have trouble overfitting initially, hence the later convergence. In comparison to BlackJack, both problems converged much later and gradually, indicating our Q-learner and exploration method may favor smaller discrete and sparse MDP problems like Black Jack, despite BlackJack being stochastic. As shown in our Cumulative Rewards over Episodes plot for BlackJack (Figure 16) we see our Q-learner rapidly learn during it's exponential phase, peaking at 674 episodes at a cumulative reward of 29.47. However, soon after it is evident that the Q-learner agent is converging and shifting away from exploration to exploitation as it coverages to around 15 cumulative rewards.

Now looking at the Cumulative Rewards plot for CartPole (Figure 17), we find that our larger CartPole problem saw much more rapid growth during it's exploration phase, which

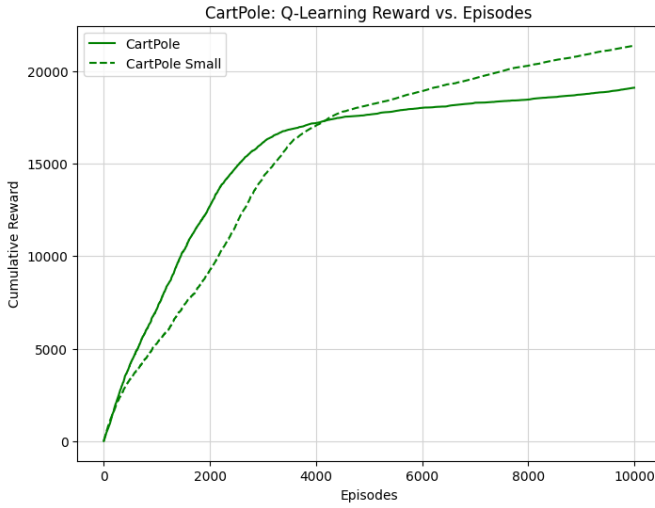


Fig. 17. Q-Learning Cumulative Reward over Episode (CartPole vs. CartPole Small)

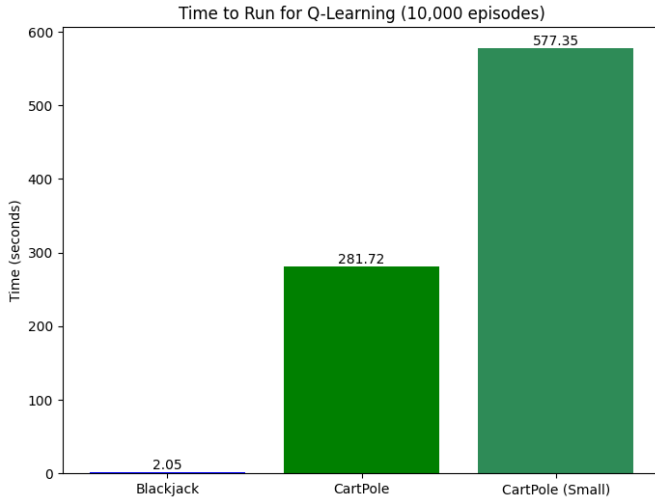


Fig. 18. Q-Learning Runtime (in seconds) over 10,000 episodes

is intuitive as we have a much larger state space. However, as we shifted away from exploration into exploitation, we saw the rapid growth of cumulative rewards from CartPole Small which remained during the remaining episodes while our original CartPole saw gradual gains. This indicates that our Q learners exploration phase prefers larger state spaces over smaller ones, while our exploitation would prefer smaller state spaces to take advantage of. In comparison to BlackJack, we see much more stability in the CartPole problems, likely due to the fact that CartPole offers deterministic dense rewards while BlackJack offers stochastic sparse rewards. Stochastic problems should increase variability inherently, which is evident in comparing BlackJack to CartPole.

Now looking at our computational expense of each MDP problem, we find BlackJack being the most efficient at 2.05 seconds, CartPole at 281.72 seconds, and CartPole Small

at 577.35 seconds. These findings are surprising and also not. BlackJack intuitively should have a low runtime given its sparse structure and small discrete state space. However, following the same intuition, you would expect CartPole Small would be much faster than CartPole. This is not the case, as for larger, dense MDP problems may not follow the same pattern as small discrete problems. This finding is likely because our CartPole Small state space has finer granularity in the angular velocity, which may increase the complexity of learning and overall slow down convergence. Another factor may be CartPole Small's state transitioning may cause the increased computation cost.

**Q-learning: Conclusion and Hypothesis Evaluation** In conclusion, our findings both validate and disprove initial hypotheses. As hypothesized, BlackJack converged much faster to its counterparts due to its trivial state size which validates our hypothesis. However, contrary to our hypothesis, CartPole Small did not show greater computational efficiency but instead took longer to converge than the original CartPole, disproving that part of our hypothesis. CartPole Small also achieved an overall higher Max Q-value than CartPole, which disproves our hypothesis that it would suffer in performance due to its small state space. These results suggest that smaller state spaces do not necessarily guarantee faster convergence or lower performance in Q-learning, indicating that factors like state granularity and the complexity of state transitions are potentially directly related to the efficiency and effectiveness of Q-Learning.

## VII. CONCLUSION

In conclusion, our experiments find that reinforcement learning through value iteration, policy iteration, and Q-Learning is directly related to the specific MDP problems we select to assess. MDP problem features, such as sparsity, stochasticity, and state size are some of the main components that may affect performance, efficiency, and interpretability especially when comparing against counterpart MDP problems. As hypothesized, small, sparse, and discrete MDP problems (i.e BlackJack) will almost always out-perform larger dense MDP problems (CartPole) in terms of efficiency and iterations to converge to our optimal policy. When comparing differing state sizes of larger dense MDP problems (CartPole vs. CartPole Small), we can expect similar results in regards to value iteration and policy iteration where state size has a notable impact in efficiency and effectivity, however this finding is not necessarily conclusive nor applies to our Q-learning finding. Here other factors such as granularity and state transition complexity do account for impact.