Reinforcement Learning Programming Assignment 1

Sooraj Srinivasan CS20B075
Vijaya Sasibind Badde CS20B087

February 2024

1 Introduction

In this report, we aim to implement and compare two Temporal Difference Learning algorithms, SARSA and Q-Learning. To this extent, we use a GridWorld environment and a agent learning a policy through SARSA and Q-Learning. We look at 6 different starting states and find the best hyperparameters and policy for each algorithm and look at the path taken by each agent at the end, illustrating this with the relevant graphs.

2 Environment

The GridWorld environment allows an agent four actions: Up, Down, Left, Right. The agent transitions to the next state determined by the action with a probability of $p \in [0, 1]$. We also define a parameter $b \in [0, 1]$ which acts the bias. The environment may also have a "wind" that can push the agent one additional cell to the right after transitioning to the new state with the probability of 0.4. The episode is terminated when the agent either reaching the goal state or when the timesteps reaches 100.

The grid dimensions is 10×10. It has the following types of states:

- *Start states*: The starting state of the agent
- *Goal states*: The end state for the agent. There are 3 such states.
- *Obstructed state*: These are walls that prevent entry to the respective cells. Transition to these states will not result in any change.
- *Bad state*: Entry into these states will incur a higher penalty than a normal state.
- *Restart state*: Entry into these states will incur a very high penalty and will cause agent to teleport to the start state without the episode ending. Once the restart state is reached, no matter what action is chosen, it goes to the start state at the next step.
- *Normal state*: None of the above. Entry into these states will incur a small penalty.

The rewards for each of the states are as follows:

- *Normal states*: -1
- *Restart states*: -100
- *Bad states*: -6
- *Goal states*: $+10$

3 Algorithms

We are looking at two temporal difference algorithms: SARSA and Q-Learning. SARSA is an on-policy algorithms where the agent learns the value function according to the action chosen by the behaviour policy. Q-Learning is an off-policy algorithm where the agent learns the value functions independent of the behaviour policy. The update equation for both of these algorithms are given below:

For SARSA,

$$Q(s_t, a_t) := Q(s_t, a_t) + \alpha(r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$$

For Q-Learning:

$$Q(s_t, a_t) := Q(s_t, a_t) + \alpha(r_t + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t))$$

4 Experiments

4.1 Initial conditions

For each experiment we have a different initial condition. We test on two different start states: $(0, 4)$ and $(3, 6)$. For each start state, we have different levels of stochasticity:

- $p = 1.0$, no wind
- $p = 0.7$, no wind
- $p = 1.0$, wind present

4.2 Hyper-parameters

We run the algorithms over different hyper-parameters such as the choice of the behaviour policy, the parameter for the behaviour policy, the value of α and γ. For the behaviour policy ϵ-greedy, we vary the value of ϵ over $[0.1, 0.2, 0.3]$. For the behaviour policy softmax, we vary the value of τ over $[0.1, 0.25, 0.75]$. The value of α is varied over $[0.2, 0.4, 0.8]$ and the value of γ is varied over $[0.7, 0.8, 0.9]$.

We choose the best value for the hyper-parameters by looking at the regret of the algorithm over the 5000 episodes it is run on.

4.3 Results

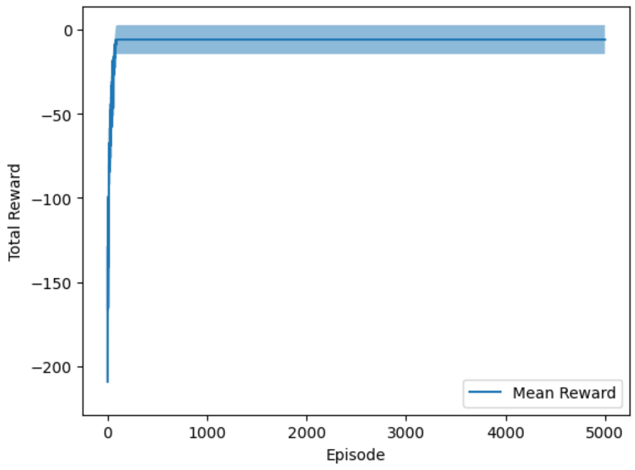
We exhibit the following graphs:

- Reward curves and the number of steps to reach the goal in each episode.
- Heatmap of the grid with state visit counts, i.e., the number of times each state was visited throughout the training phase.
- Heatmap of the grid with Q values after training is complete, and optimal actions for the best policy.

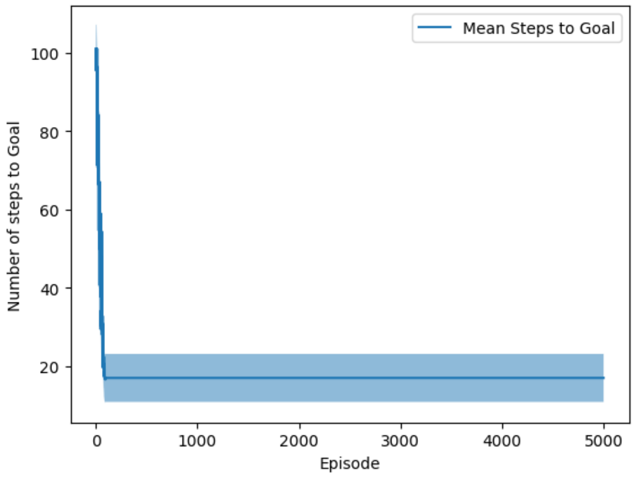
4.3.1 Initial conditions: $(0, 4)$, $p = 1.0$ and no wind

For SARSA, the graphs 1a, 1b, 2a and 2b show the reward vs timesteps graph, the steps vs timesteps graph, the heatmap of the Q values after training is complete and optimal actions for the best policy, and the heatmap of the grid with state visit counts

The best hyper parameters for SARSA were found to be: $\alpha = 0.4, \gamma = 0.9$ and the best policy was softmax with $\tau = 0.1$.

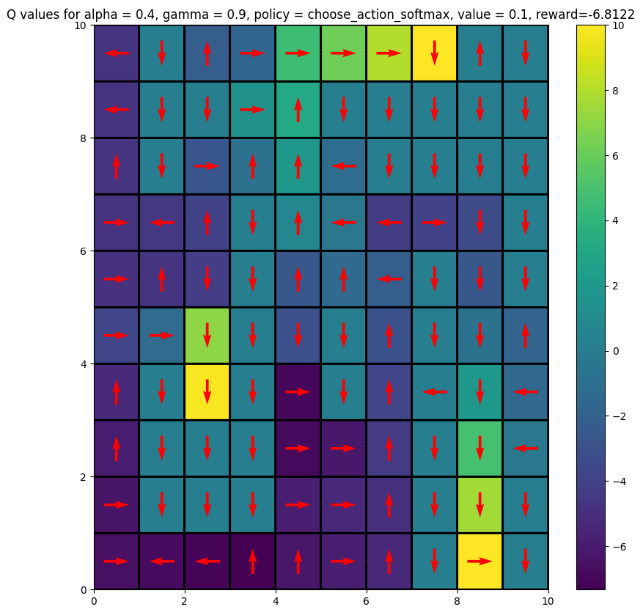


(a) Reward vs Timesteps

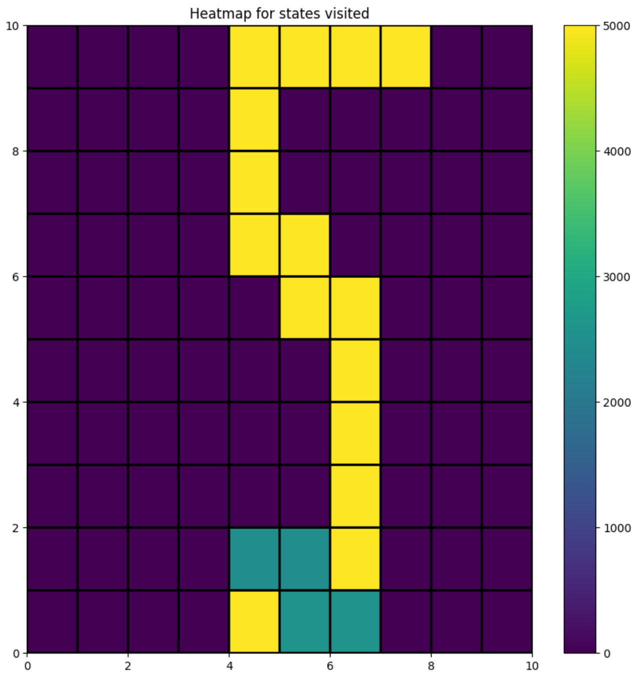


(b) Steps vs timesteps

Figure 1: Reward and steps vs timesteps for SARSA



(a) Heatmap of Q values and optimal actions

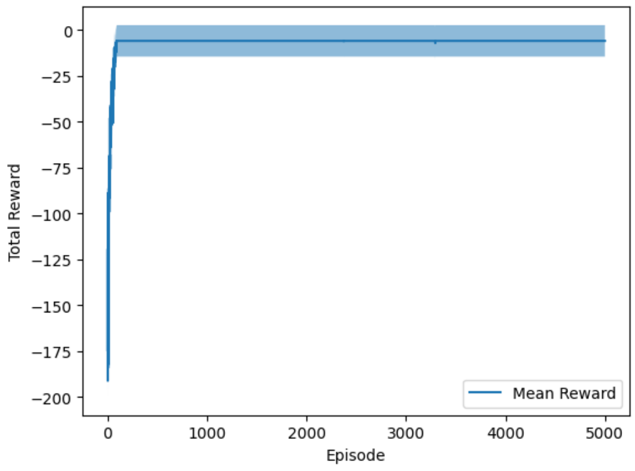


(b) Heatmap of state visit counts

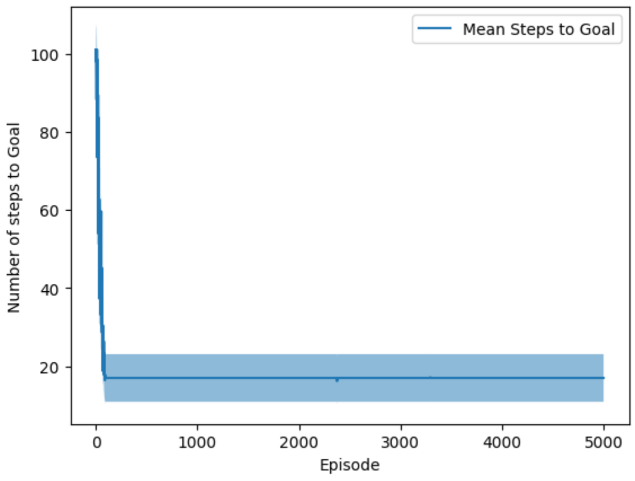
Figure 2: Heatmaps of the optimal policy for SARSA

For Q-Learning, the graphs 3a, 3b, 4a and 4b show the reward vs timesteps graph, the steps vs timesteps graph, the heatmap of the Q values after training is complete and optimal actions for the best policy, and the heatmap of the grid with state visit counts.

The best hyperparameters for Q-Learning was found to be: $\alpha = 0.4, \gamma = 0.9$ and the best policy being softmax with $\tau = 0.1$

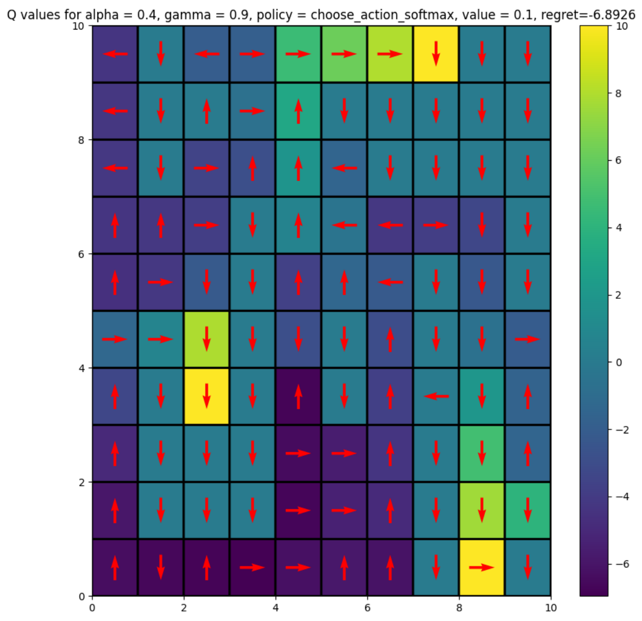


(a) Reward vs Timesteps

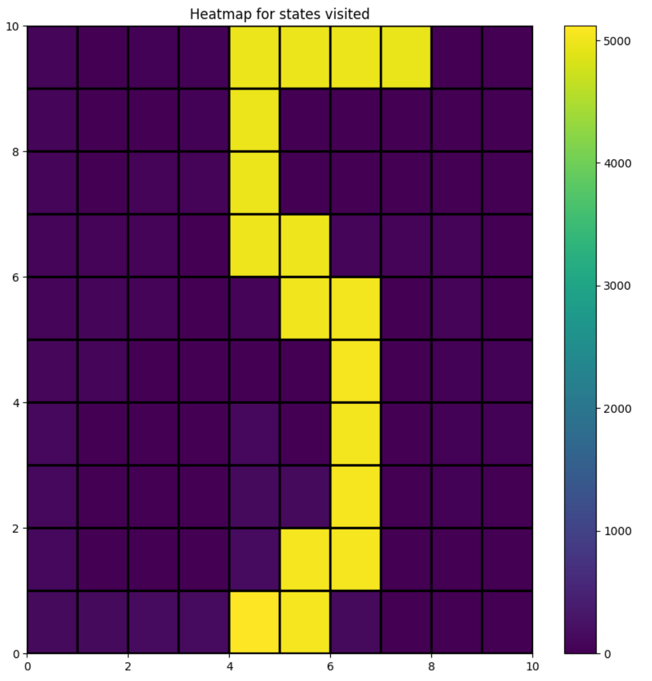


(b) Steps vs timesteps

Figure 3: Reward and steps vs timesteps for Q-Learning



(a) Heatmap of Q values and optimal actions



(b) Heatmap of state visit counts

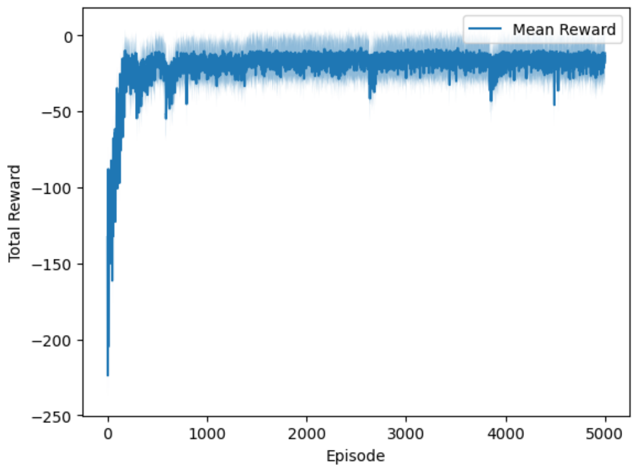
Figure 4: Heatmaps of the optimal policy for Q-Learning

Observations: We see that both SARSA and Q-Learning both learn one of the optimal paths. This shows that once the optimal path has been determined, Q-Learning tends to be more greedy with its explorations as opposed to SARSA. For both the algorithms, a high α and a high γ do well in reducing the regret. Softmax tends to perform better than ϵ-greedy and a lower value of τ allows it to better exploit the optimal path found.

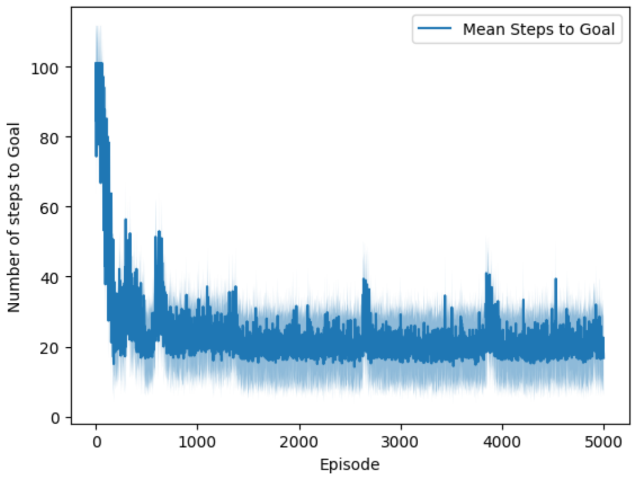
4.3.2 Initial conditions: $(0, 4)$, $p = 0.7$ and no wind

For SARSA, the graphs 5a, 5b, 6a and 6b show the reward vs timesteps graph, the steps vs timesteps graph, the heatmap of the Q values after training is complete and optimal actions for the best policy, and the heatmap of the grid with state visit counts.

The best hyper parameters for SARSA were found to be: $\alpha = 0.2, \gamma = 0.9$ and the best policy was softmax with $\tau = 0.1$.

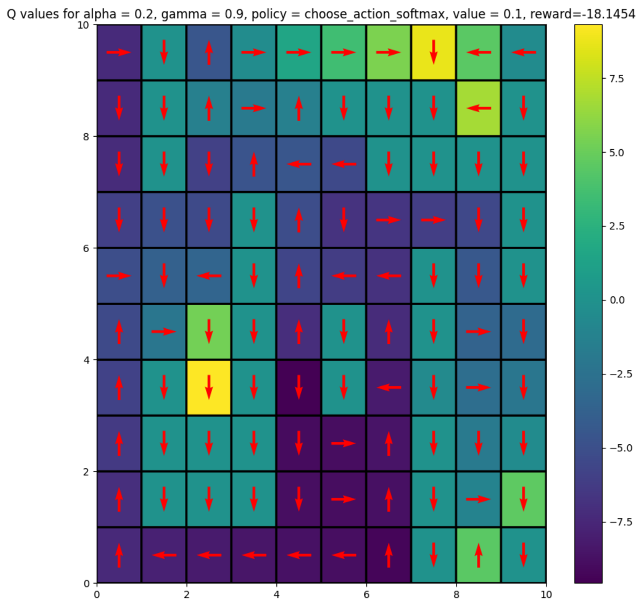


(a) Reward vs Timesteps

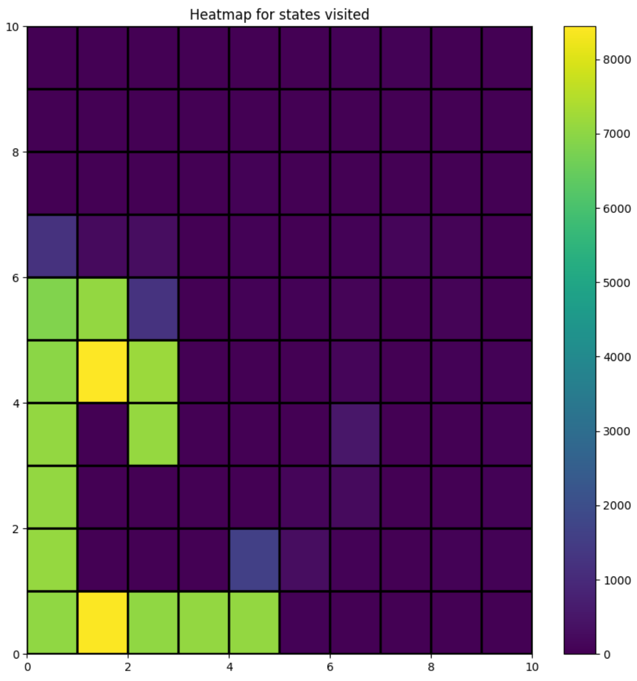


(b) Steps vs timesteps

Figure 5: Reward and steps vs timesteps for SARSA



(a) Heatmap of Q values and optimal actions

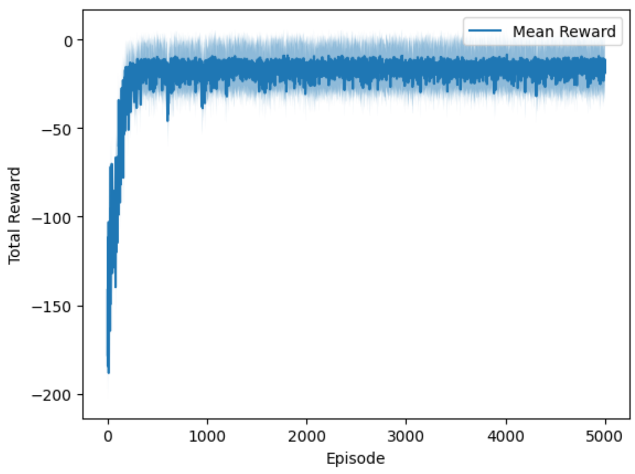


(b) Heatmap of state visit counts

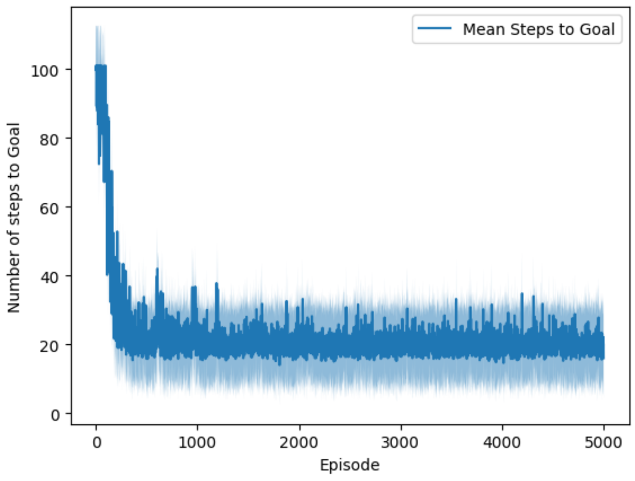
Figure 6: Heatmaps of the optimal policy for SARSA

For Q-Learning, the graphs 7a, 7b, 8a and 8b show the reward vs timesteps graph, the steps vs timesteps graph, the heatmap of the Q values after training is complete and optimal actions for the best policy, and the heatmap of the grid with state visit counts.

The best hyperparameters for Q-Learning was found to be: $\alpha = 0.2, \gamma = 0.9$ and the best policy being softmax with $\tau = 0.1$

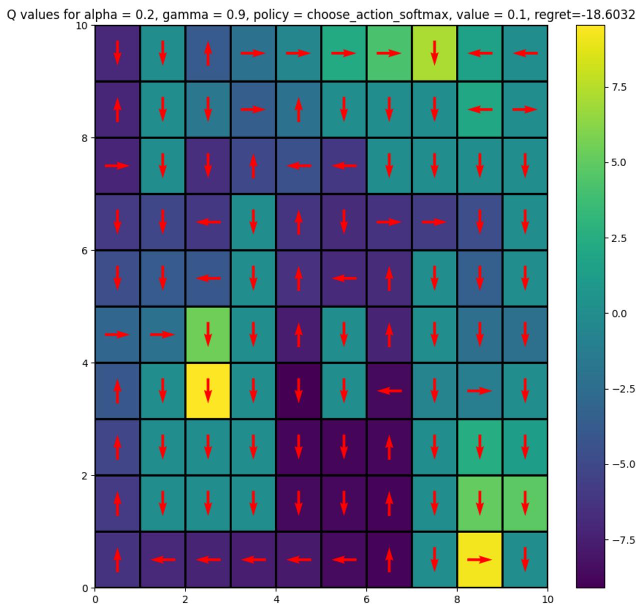


(a) Reward vs Timesteps

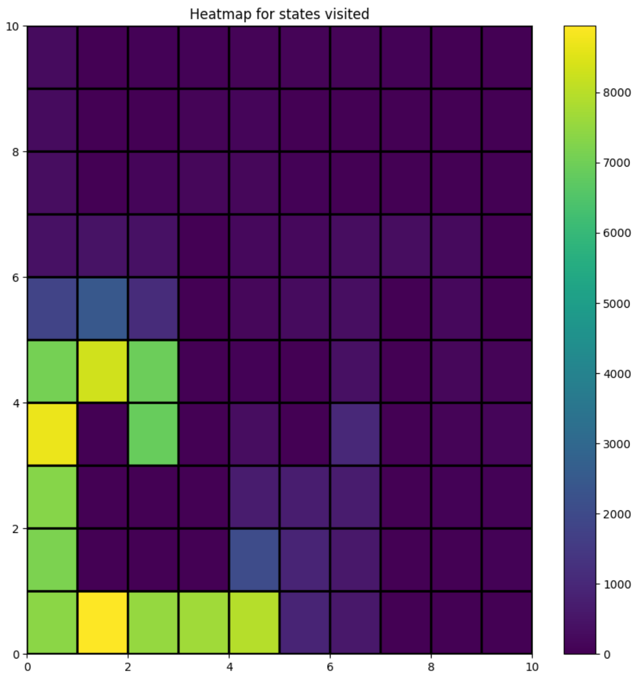


(b) Steps vs timesteps

Figure 7: Reward and steps vs timesteps for Q-Learning



(a) Heatmap of Q values and optimal actions



(b) Heatmap of state visit counts

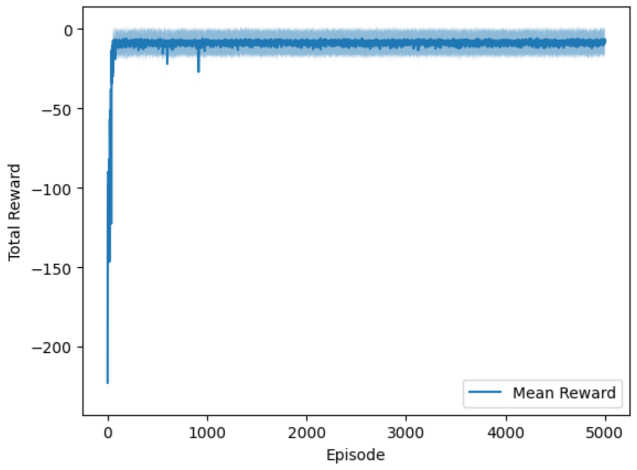
Figure 8: Heatmaps of the optimal policy for Q-Learning

Observations: Both SARSA and Q-Learning learn the optimal paths. Due to the stochasticity, the reward and step graphs are far less continuous than before. From the heatmaps, we can see that SARSA in the process of exploration does not move as much as Q-Learning, which explores the other goals states as well. For this case as well, a high γ does well with softmax being the best policy when $\tau = 0.1$. A low α seems to be better in this case, possibly because a high α might lead to drastic changes in the reward function due to a stochastic move in the wrong direction, leading to more regret.

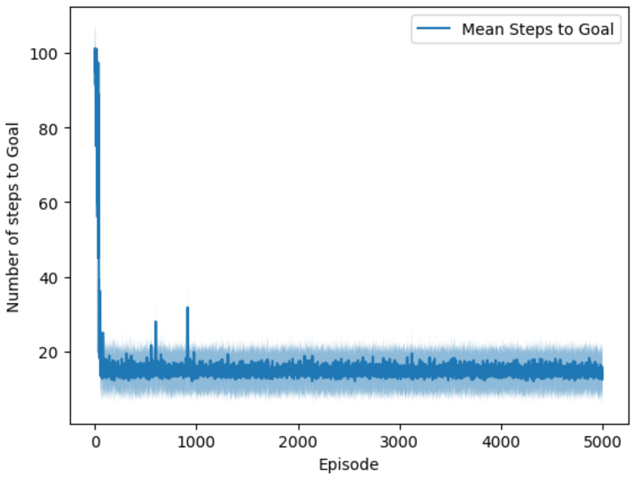
4.3.3 Initial conditions: $(0, 4)$, $p = 1.0$ and wind is present

For SARSA, the graphs 9a, 9b, 10a and 10b show the reward vs timesteps graph, the steps vs timesteps graph, the heatmap of the Q values after training is complete and optimal actions for the best policy, and the heatmap of the grid with state visit counts.

The best hyper parameters for SARSA were found to be: $\alpha = 0.4, \gamma = 0.9$ and the best policy was softmax with $\tau = 0.1$.

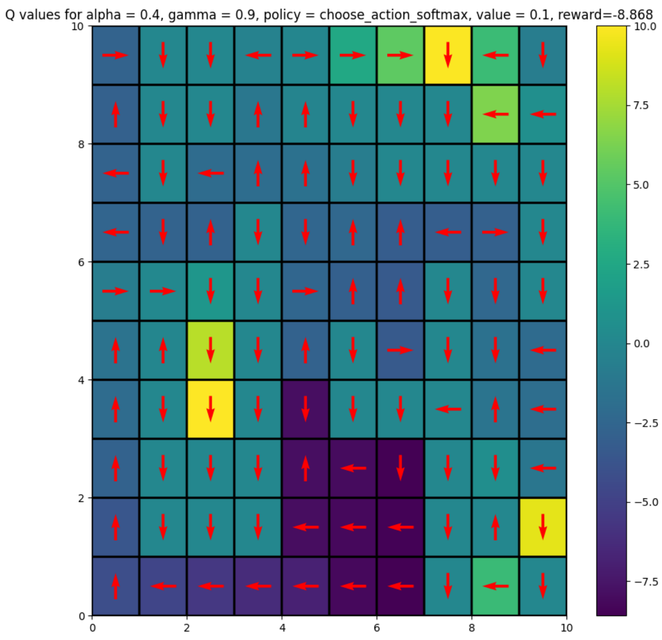


(a) Reward vs Timesteps

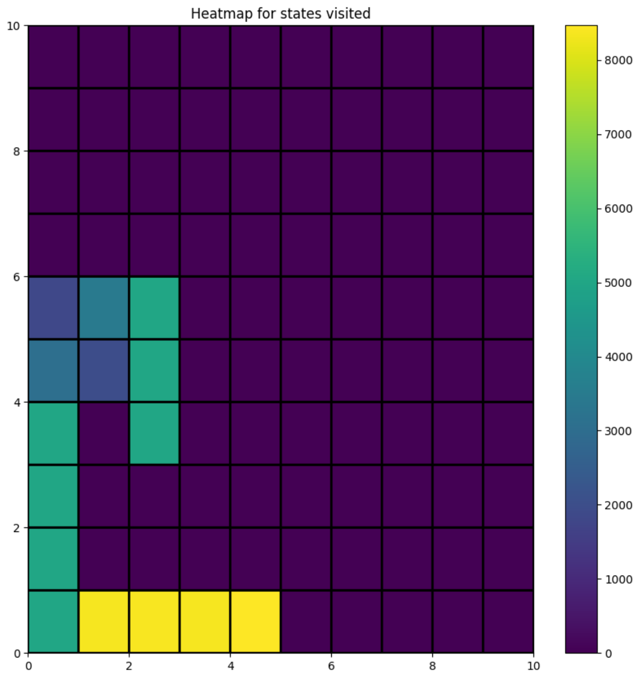


(b) Steps vs timesteps

Figure 9: Reward and steps vs timesteps for SARSA



(a) Heatmap of Q values and optimal actions

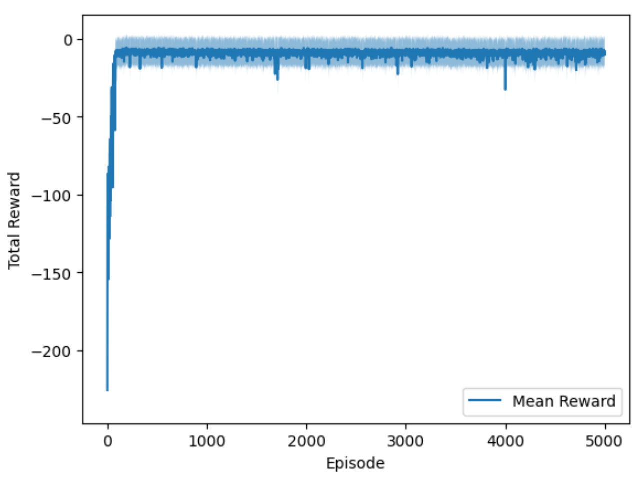


(b) Heatmap of state visit counts

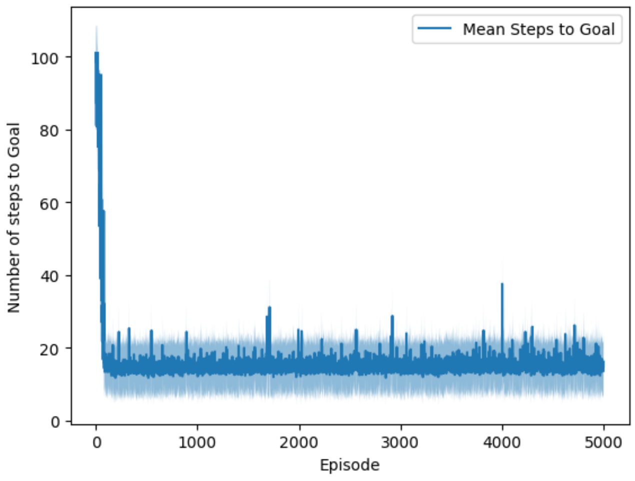
Figure 10: Heatmaps of the optimal policy for SARSA

For Q-Learning, the graphs 11a, 11b, 12a and 12b show the reward vs timesteps graph, the steps vs timesteps graph, the heatmap of the Q values after training is complete and optimal actions for the best policy, and the heatmap of the grid with state visit counts.

The best hyperparameters for Q-Learning was found to be: $\alpha = 0.4, \gamma = 0.9$ and the best policy being softmax with $\tau = 0.1$

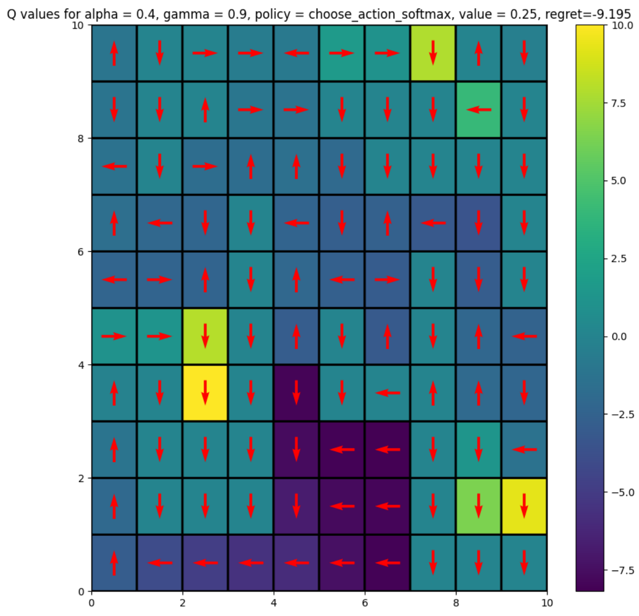


(a) Reward vs Timesteps

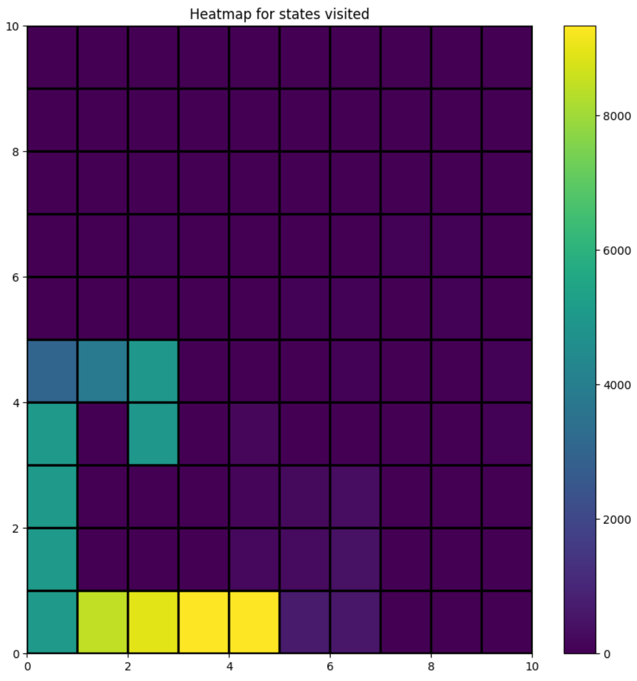


(b) Steps vs timesteps

Figure 11: Reward and steps vs timesteps for Q-Learning



(a) Heatmap of Q values and optimal actions



(b) Heatmap of state visit counts

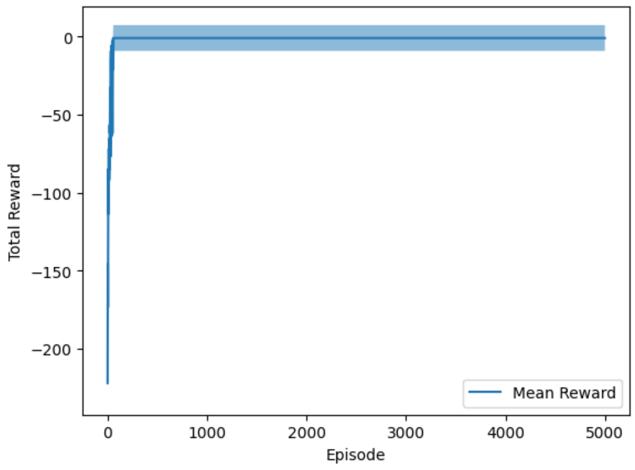
Figure 12: Heatmaps of the optimal policy for Q-Learning

Observations: Similar to the previous experiments, SARSA and Q-Learning both learn the optimal path. Q-Learning does seem to have a higher regret than SARSA which might have been due to the exploratory phase. A high α and a high γ work well in this experiment as well, with softmax having $\tau = 0.1$ be the best policy.

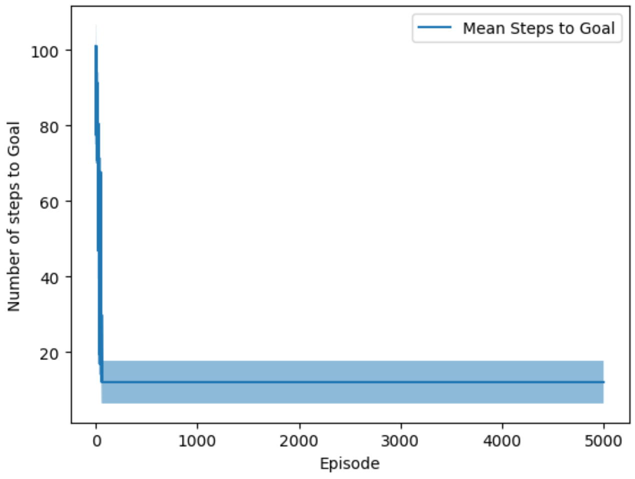
4.3.4 Initial conditions: $(3, 6)$, $p = 1.0$ and no wind

For SARSA, the graphs 13a, 13b, 14a and 14b show the reward vs timesteps graph, the steps vs timesteps graph, the heatmap of the Q values after training is complete and optimal actions for the best policy, and the heatmap of the grid with state visit counts.

The best hyper parameters for SARSA were found to be: $\alpha = 0.4, \gamma = 0.9$ and the best policy was softmax with $\tau = 0.1$.

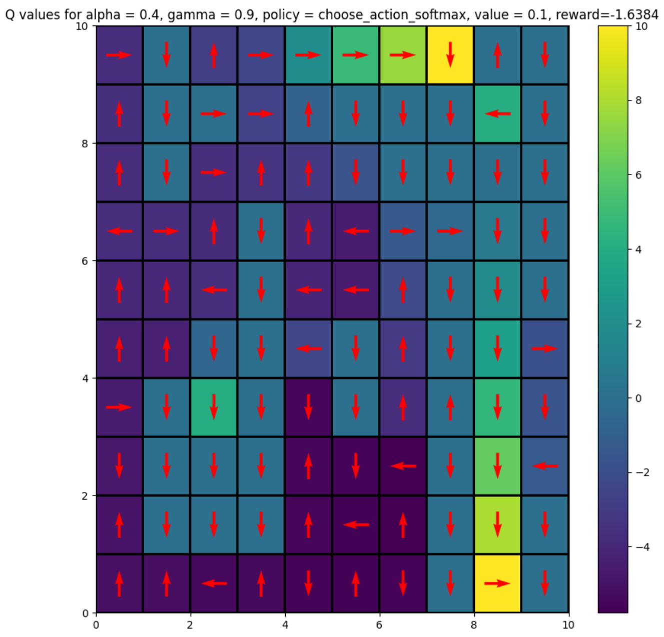


(a) Reward vs Timesteps

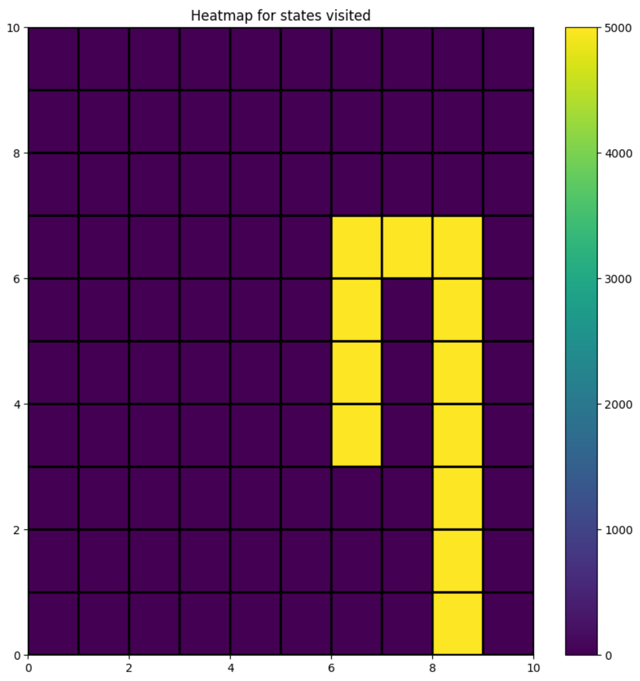


(b) Steps vs timesteps

Figure 13: Reward and steps vs timesteps for SARSA



(a) Heatmap of Q values and optimal actions

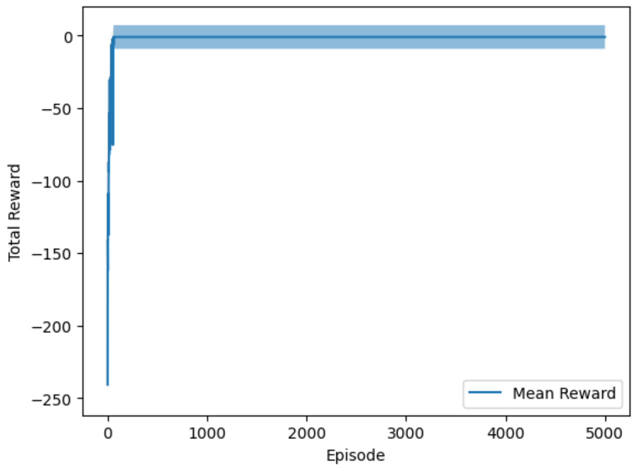


(b) Heatmap of state visit counts

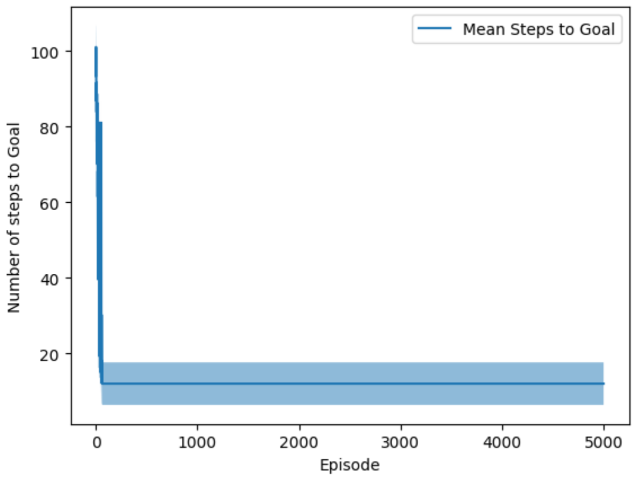
Figure 14: Heatmaps of the optimal policy for SARSA

For Q-Learning, the graphs 15a, 15b, 16a and 16b show the reward vs timesteps graph, the steps vs timesteps graph, the heatmap of the Q values after training is complete and optimal actions for the best policy, and the heatmap of the grid with state visit counts.

The best hyperparameters for Q-Learning was found to be: $\alpha = 0.4, \gamma = 0.9$ and the best policy being softmax with $\tau = 0.1$

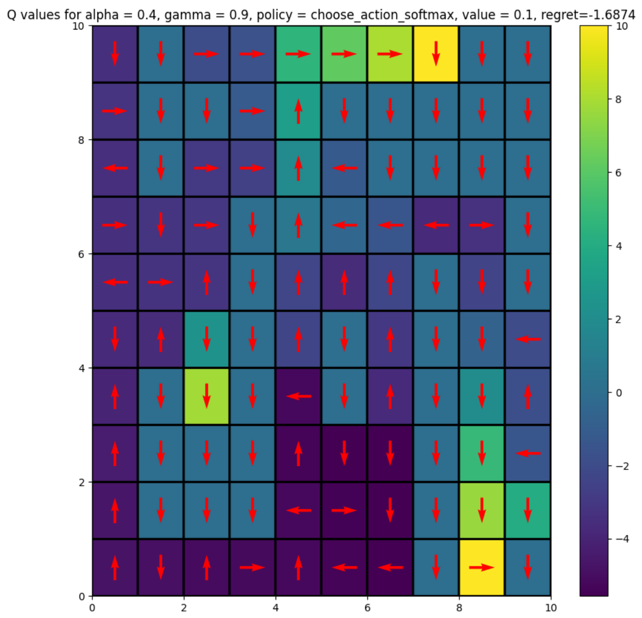


(a) Reward vs Timesteps

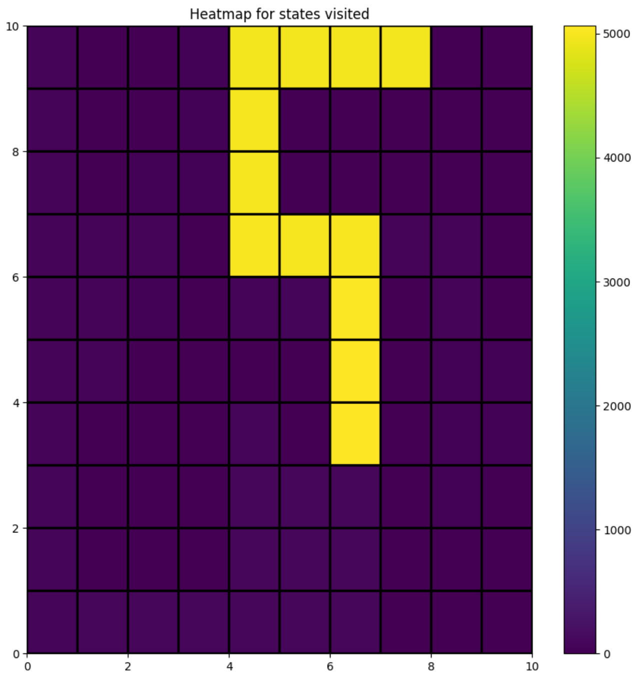


(b) Steps vs timesteps

Figure 15: Reward and steps vs timesteps for Q-Learning



(a) Heatmap of Q values and optimal actions



(b) Heatmap of state visit counts

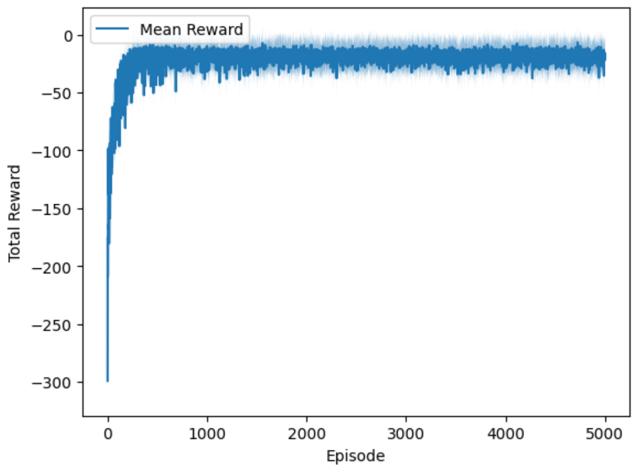
Figure 16: Heatmaps of the optimal policy for Q-Learning

Observations: SARSA and Q-Learning learn different paths in this experiment, although they both have the same reward. This could be due to SARSA preferring a safer path as opposed to Q-Learning. A high α and γ work well here, with softmax having $\tau = 0.1$ be the best policy.

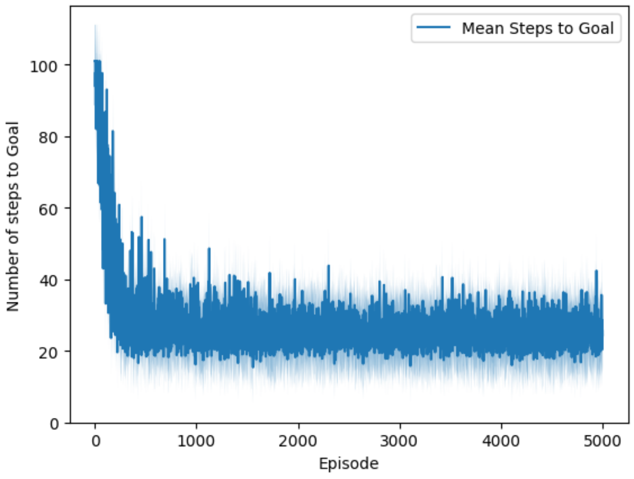
4.3.5 Initial conditions: $(3, 6)$, $p = 0.7$ and no wind

For SARSA, the graphs 17a, 17b, 18a and 18b show the reward vs timesteps graph, the steps vs timesteps graph, the heatmap of the Q values after training is complete and optimal actions for the best policy, and the heatmap of the grid with state visit counts.

The best hyper parameters for SARSA were found to be: $\alpha = 0.2, \gamma = 0.9$ and the best policy was softmax with $\tau = 0.1$.

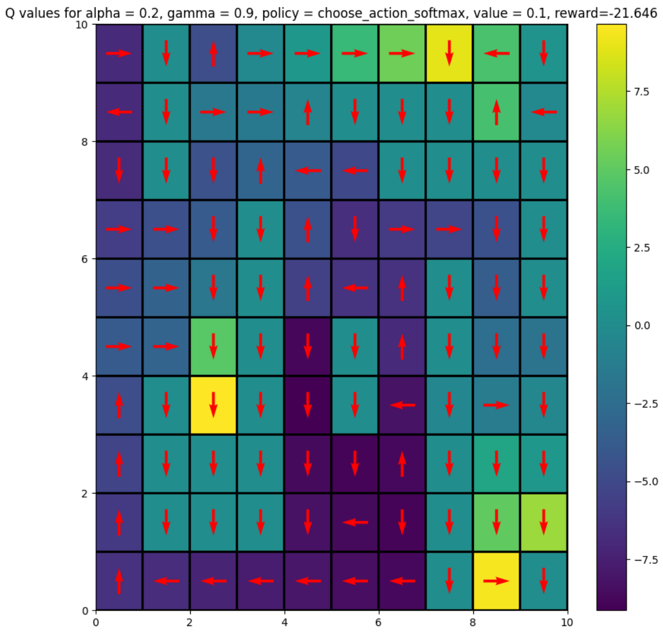


(a) Reward vs Timesteps

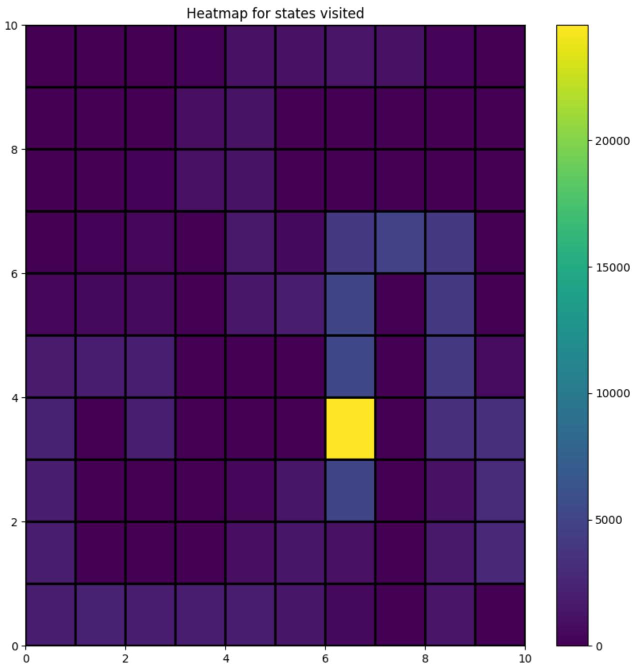


(b) Steps vs timesteps

Figure 17: Reward and steps vs timesteps for SARSA



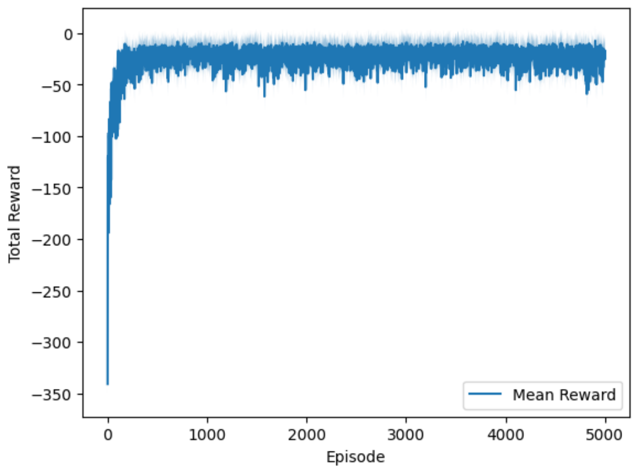
(a) Heatmap of Q values and optimal actions



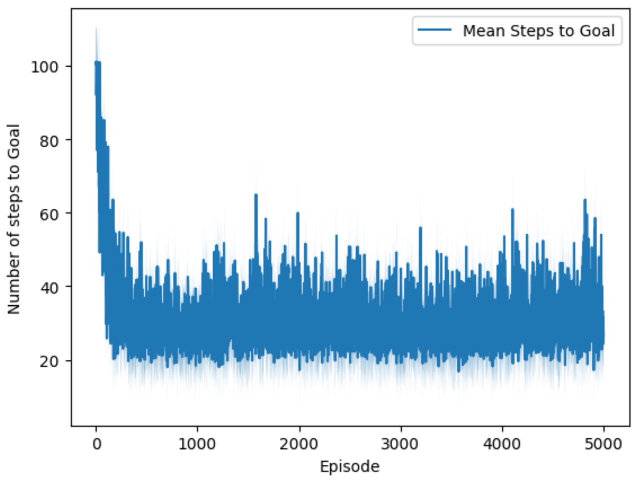
(b) Heatmap of state visit counts

Figure 18: Heatmaps of the optimal policy for SARSA

For Q-Learning, the graphs 19a, 19b, 20a and 20b show the reward vs timesteps graph, the steps vs timesteps graph, the heatmap of the Q values after training is complete and optimal actions for the best policy, and the heatmap of the grid with state visit counts.

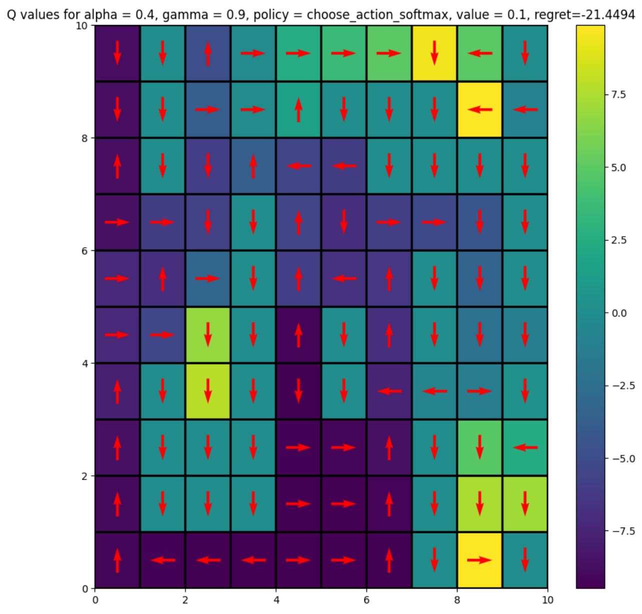


(a) Reward vs Timesteps

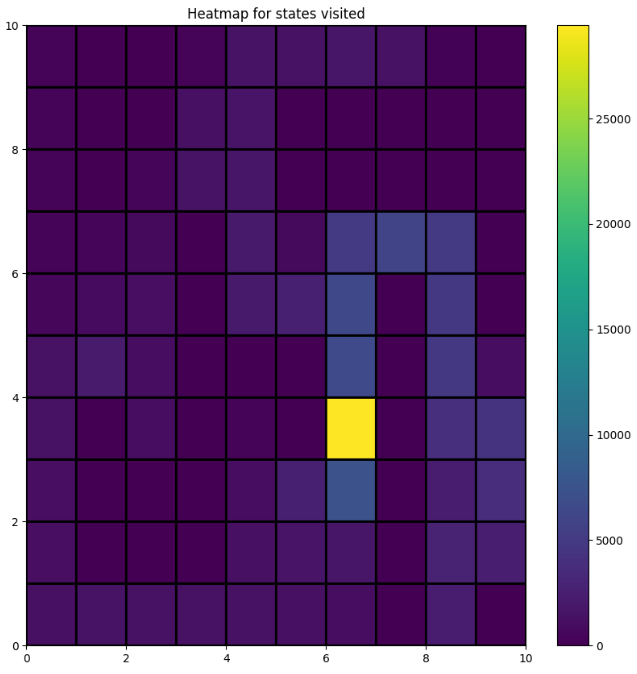


(b) Steps vs timesteps

Figure 19: Reward and steps vs timesteps for Q-Learning



(a) Heatmap of Q values and optimal actions



(b) Heatmap of state visit counts

Figure 20: Heatmaps of the optimal policy for Q-Learning

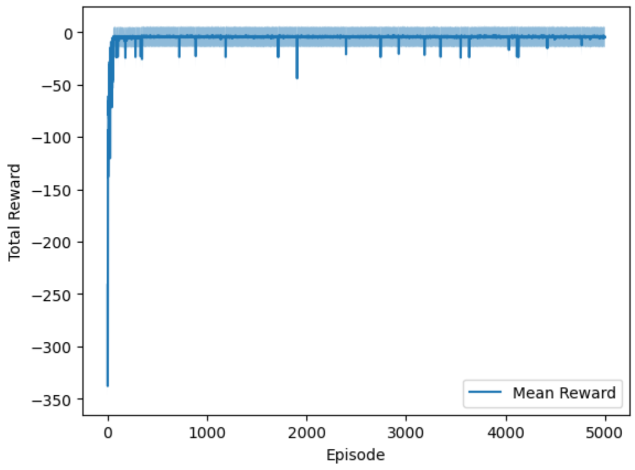
The best hyperparameters for Q-Learning was found to be: $\alpha = 0.4, \gamma = 0.9$ and the best policy being softmax with $\tau = 0.1$

Observations: Both SARSA and Q-Learning learn the same optimal path, although due to the stochastic nature, the algorithms have explored a lot more of the grid. We can see that the graphs for rewards and steps versus timesteps are far less continuous due to the stochasticity. An interesting comparision can be made with 4.3.2 where a lower α for the very case seemed to work out better. Indeed, for Q-Learning, all other parameters being the same values, $\alpha = 0.2$ has regret -21.825 whereas $\alpha = 0.4$ has regret -21.4494. For SARSA, all other parameters having the same values, $\alpha = 0.2$ has regret -21.646 whereas $\alpha = 0.4$ has regret -21.8582. This shows that although a higher α can help you get to the optimal path faster, a lower α helps in smoothing out the curve, allowing you to make less mistakes. For the other parameters, a high γ with softmax having $\tau = 0.1$ work well here.

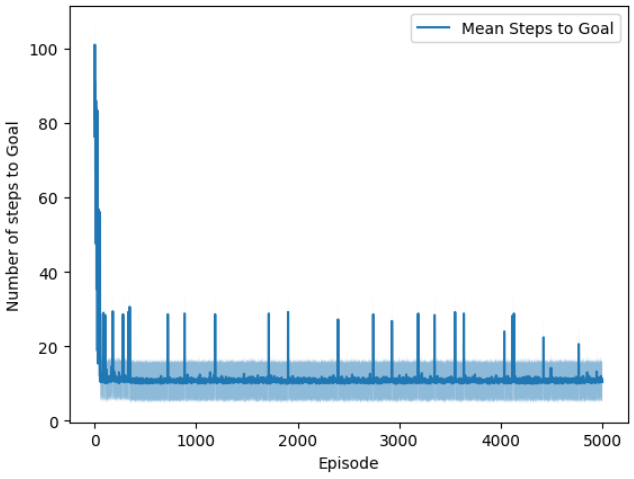
4.3.6 Initial conditions: $(3, 6)$, $p = 1.0$ and wind is present

For SARSA, the graphs 21a, 21b, 22a and 22b show the reward vs timesteps graph, the steps vs timesteps graph, the heatmap of the Q values after training is complete and optimal actions for the best policy, and the heatmap of the grid with state visit counts.

The best hyper parameters for SARSA were found to be: $\alpha = 0.4, \gamma = 0.9$ and the best policy was softmax with $\tau = 0.1$.

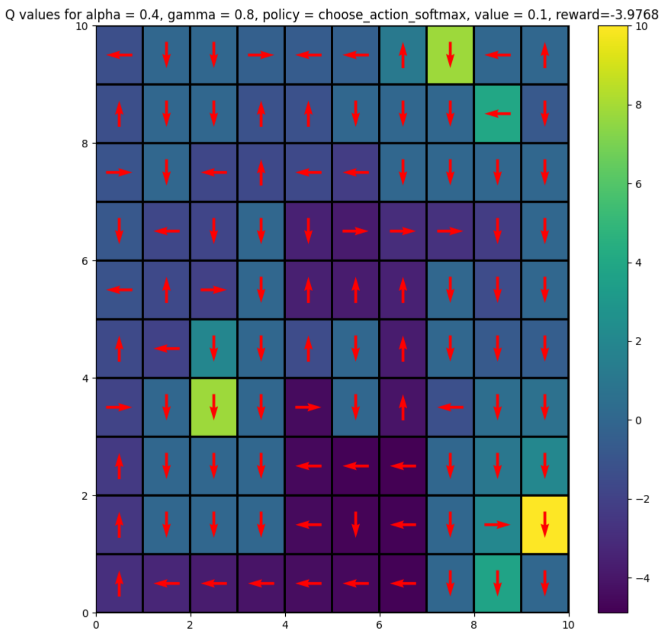


(a) Reward vs Timesteps

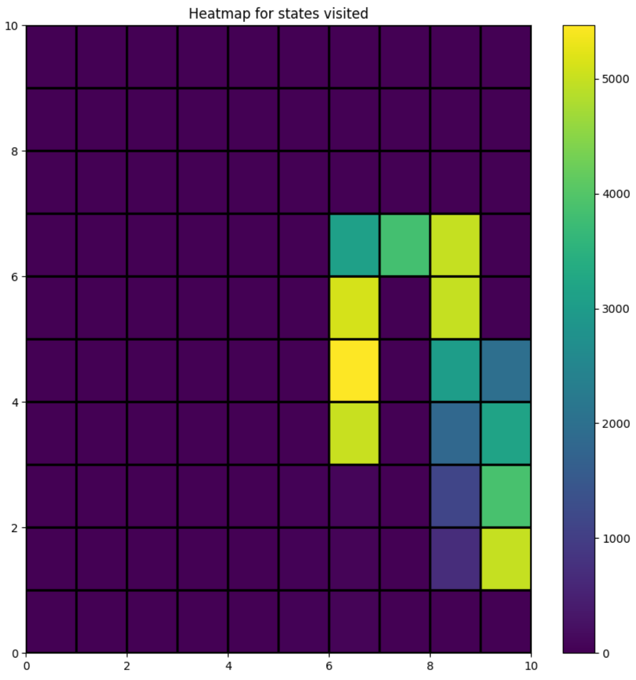


(b) Steps vs timesteps

Figure 21: Reward and steps vs timesteps for SARSA



(a) Heatmap of Q values and optimal actions

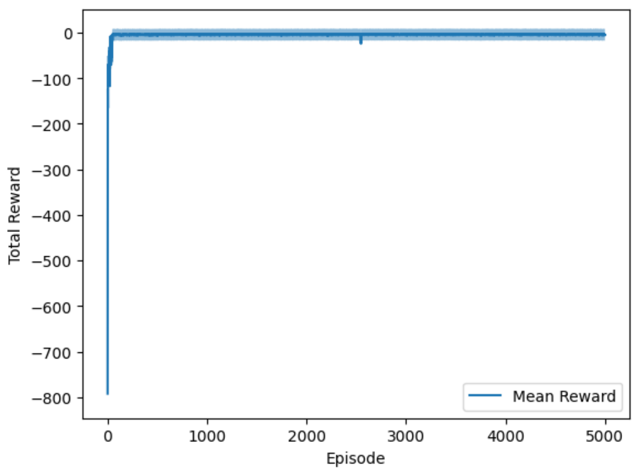


(b) Heatmap of state visit counts

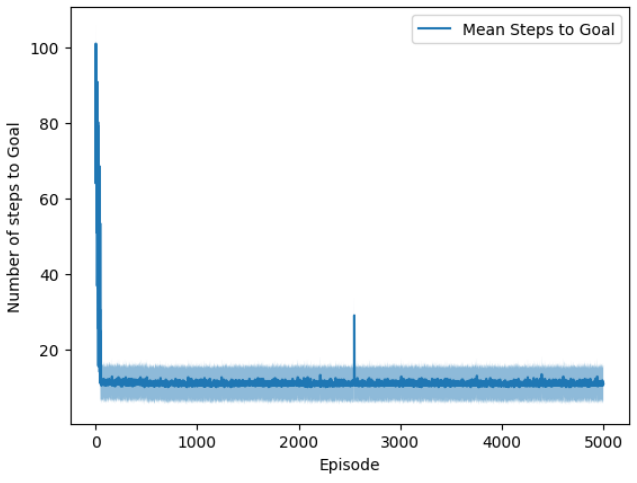
Figure 22: Heatmaps of the optimal policy for SARSA

For Q-Learning, the graphs 23a, 23b, 24a and 24b show the reward vs timesteps graph, the steps vs timesteps graph, the heatmap of the Q values after training is complete and optimal actions for the best policy, and the heatmap of the grid with state visit counts.

The best hyperparameters for Q-Learning was found to be: $\alpha = 0.4, \gamma = 0.9$ and the best policy being softmax with $\tau = 0.1$

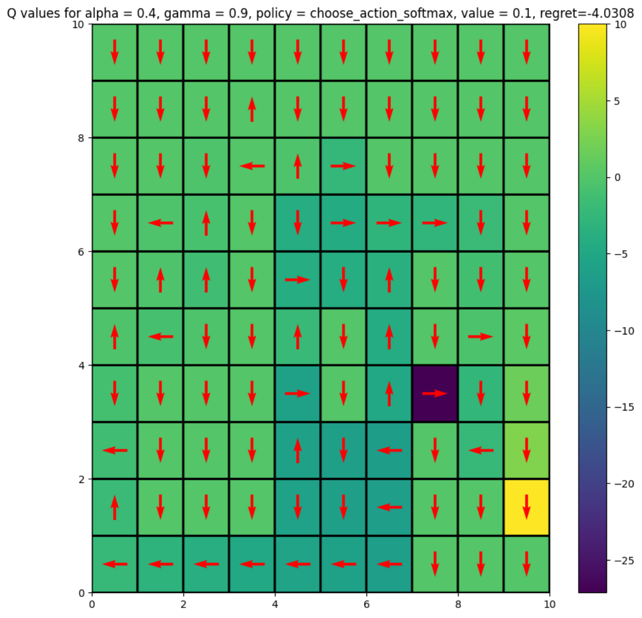


(a) Reward vs Timesteps

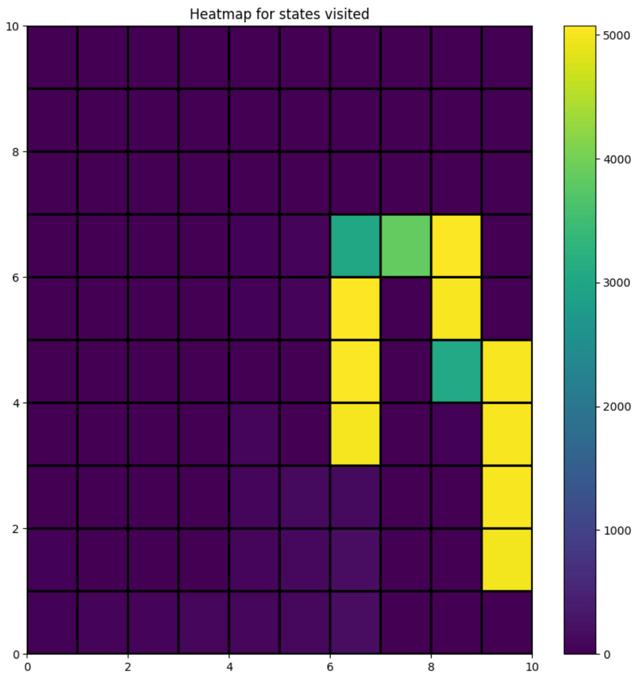


(b) Steps vs timesteps

Figure 23: Reward and steps vs timesteps for Q-Learning



(a) Heatmap of Q values and optimal actions



(b) Heatmap of state visit counts

Figure 24: Heatmaps of the optimal policy for Q-Learning

Observations: Both SARSA and Q-Learning learn the same optimal path. Similar to 4.3.1, we see that SARSA tends to explore a bit more once it finds the optimal path, whereas Q-Learning sticks to the optimal path once it has been found. A high α and γ with softmax having $\tau = 0.1$ work well here.

5 Overall Observations

After running the 12 experiments for the 6 initial conditions for SARSA and Q-Learning, we observe:

- A high γ value works well with both SARSA and Q-Learning. This is because a high γ allows you to better communicate the position of the goal state to the start state for longer paths leading to lesser regret overall.
- A high α will generally work well for both SARSA and Q-Learning. This is because a high α if found both lead to faster convergence to the optimal path. However, as we see in 4.3.2 and 4.3.5, a lower α can lead to lesser regret in stochastic conditions. This could be interpreted as a smoothening of the reward curve, which allows the algorithm to focus less on a fortuitous correct move and more on the general pattern observed.
- Softmax performs better than ϵ-greedy in every experiment. This is because ϵ-greedy has a fixed value of ϵ for the explore action whereas softmax reduces the probability of exploring as the timesteps increase, which allows it to better reduce regret.
- For softmax, a lower τ works well with both SARSA and Q-Learning. A lower τ means it explores less and exploits more, which works well for the current experiment setup. A similar setup where the actual optimal goal state is much further away from the suboptimal goal state might not have a lower τ work as well.
- SARSA and Q-Learning always learn the optimal paths for all the algorithms. The only difference that can be observed is that is when the optimal path has been found by both algorithms, SARSA tends to explore a bit more whereas Q-Learning is a bit more greedy and sticks to the optimal path found.

6 Conclusion

We have analyzed the performance of both SARSA and Q-Learning over 12 different experiments. On the whole, SARSA and Q-Learning both work well in finding the optimal path. A higher value of α and γ tends to work well with the behaviour policy of softmax having $\tau = 0.1$.