CS6700: Reinforcement Learning

Programming Assignment # 1

Shreya .S. Ramanujam, EE21B126
Srikar Babu Gadipudi, EE21B138

February 28, 2024

Contents

1 Introduction

The setting for this assignment is the Grid World shown in Figure 1. We have shown the grid for a start state of (0,4).

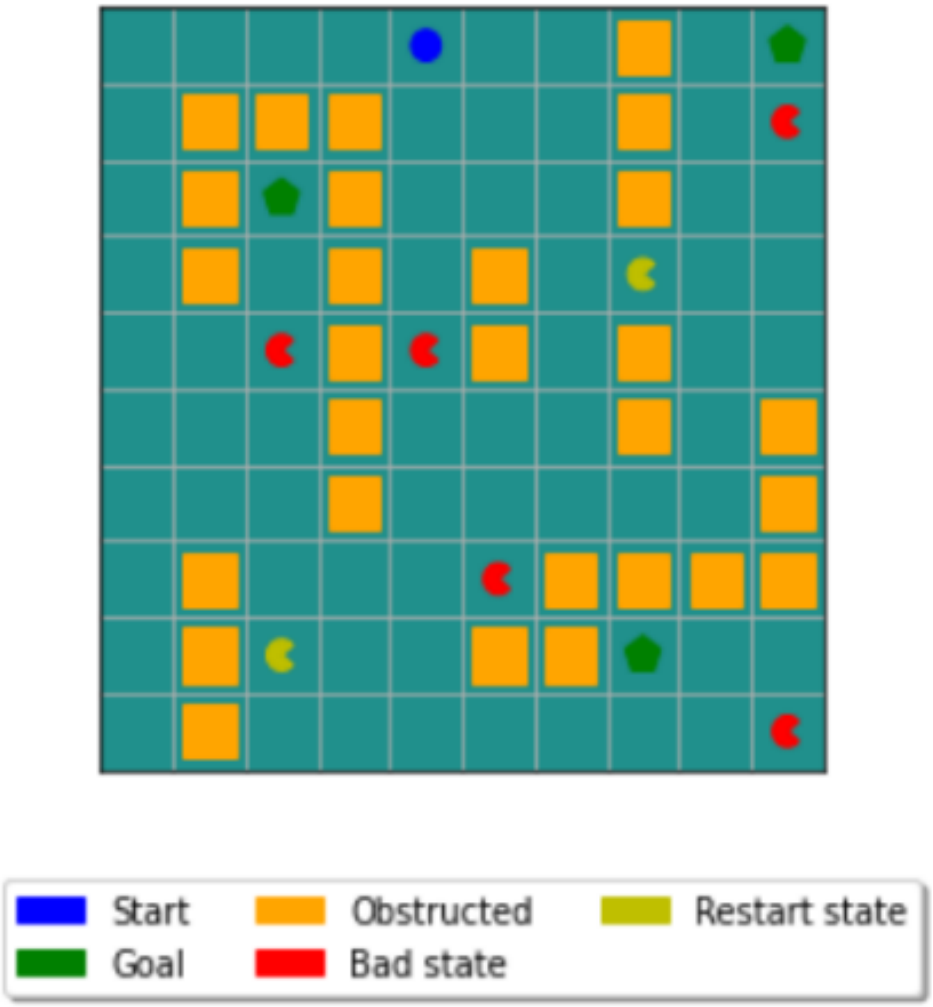


Figure 1: Grid World used in our experiments

The descriptions for the various states are as follows:

- **Start State:** The state from which the agent starts at the beginning of each episode. In our experiments, there are two different start states: (3,6) and (0,4).
- **Goal State:** The agent's goal is to reach one of the 3 goal states in each episode. The goal states are (0,9), (2,2) and (8,7). Agent gets a +10 reward on reaching a goal state.
- **Obstructions:** These states are walls which the agent cannot enter into. Transitions to these states will not result in any change.
- **Bad State:** The agent gets a higher penalty (-6) if it visits these states. In our grid, there are 5 Bad states, shown in red.

- **Restart State:** Entering these states will incur a very high penalty (-100) and teleport the agent back to the start state in the same ongoing episode. Once the restart state is reached, no matter what action is chosen, the agent goes to the start state at the next step. In our grid, there are 2 restart states: (3,7) and (8,2).
- **Normal State:** The light blue coloured states that fall into none of the other categories are the Normal states. Entering these states will give the agent a small penalty of -1.

The actions and environment settings are as described below:

- The agent has 4 available actions in any state: UP, DOWN, LEFT and RIGHT. Transitions that take you off the grid will result in no change of state.
- The agent transitions to the next state determined by the direction of the action chosen with a probability of $p \, \epsilon \, [0, 1]$.
- We also define a parameter $p \, \epsilon \, [0, 1]$, which determines the agent's transition probabilities into states perpendicular to the direction of the chosen action. For example, if the direction of the agent's chosen action is considered 'North', the agent transitions to the state 'West' of the chosen action with probability $(1 - p) \times b$, and to the 'East' of the chosen action with probability $(1 - p) \times (1 - b)$.
- The environment may also have a wind blowing that can push the agent one additional cell to the right after transitioning to its new state with a probability of 0.4. The wind in the environment can be activated by setting the variable $wind = True$.

Our goal in this assignment is to train an RL agent to navigate through this Grid World setting (as close to the optimal path as possible) while maximizing reward as much as possible. We run experiments by using different settings of the environment, tweaking other hyperparameters and training using different algorithms to identify the best performing variants in each case.

2 Experiments

We run experiments by changing the following parameters:

- Environment Setup:
  - Start state: (3,6) or (0,4)
  - Grid World parameters: ($wind$ = False, $p = 1.0$), ($wind$ = False, $p = 0.7$) or ($wind$ = True, $p = 1.0$)
- Algorithm: SARSA or Q Learning

where $wind$ is a boolean variable used to control wind in the environment and p is the probability of the agent transitioning to the next state determined by the action we take.

For each environment setup, we try two different algorithms: SARSA and Q Learning. The pseudocodes for these algorithms are given in Algorithms 1 and 2 respectively.

Algorithm 1: SARSA

Initialize $Q(s, a)$ arbitrarily
episodes $\leftarrow$ number of episodes
for $i = 1$ *to episodes* **do**
  Initialize state s
  Choose action a from state s using chosen policy (Softmax/Epsilon-greedy)
  for *each step of episode until state s is terminal* **do**
    Take action a, observe reward r and next state s'
    Choose action a' from state s' using chosen policy (Softmax/Epsilon-greedy)
    $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma Q(s', a') - Q(s, a)]$
    $s \leftarrow s'$
    $a \leftarrow a'$
  end
end

Algorithm 2: Q Learning

Initialize $Q(s, a)$ arbitrarily
episodes $\leftarrow$ number of episodes
for $i = 1$ *to episodes* **do**
  Initialize state s
  for *each step of episode until state s is terminal* **do**
    Choose action a from state s using chosen policy (Softmax/Epsilon-greedy)
    Take action a, observe reward r and next state s'
    $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma max_{a'} Q(s', a') - Q(s, a)]$
    $s \leftarrow s'$
  end
end

We know, from theory, that the SARSA algorithm converges to the safer/low-risk path, whereas the Q Learning algorithm favours the shorter path, even if it has great risk of penalties. We make further case-by-case observations and comparisons for each setting in the sections below.

For each algorithm, we also experiment with different action selection policies, namely softmax and epsilon greedy policies. These policies have been explained in the sections below.

2.0.1 Epsilon-Greedy Policy

The epsilon-greedy policy is a simple strategy used in reinforcement learning. It balances exploration and exploitation by selecting the optimal action with probability $1 - \epsilon$ and exploring a random action with probability ϵ.

The action selection probability function is given by:

$$\pi(a|s) = \begin{cases} 1 - \epsilon + \epsilon/\text{num\_actions} & \text{if } a = \text{argmax}_{a'} Q(s, a') \\ \epsilon/\text{num\_actions} & \text{otherwise} \end{cases}$$

where:

$\pi(a|s)$ is the probability of selecting action $a$from state s,
$Q(s, a')$ is the estimated action value for state s and action a',
$\text{argmax}_{a'} Q(s, a')$ is the action that maximizes $Q(s, a')$,
ϵ is the exploration-exploitation trade-off parameter,
num_actions is the total number of possible actions.

This policy ensures that with high probability, the agent exploits the action with the highest estimated value, while occasionally exploring new actions.

2.0.2 Softmax Policy

The softmax policy is a probabilistic strategy used in reinforcement learning. It converts Q values into a probability distribution over actions. The action selection probability function is given by the softmax function:

$$\pi(a|s) = \frac{e^{Q(s,a)/\tau}}{\sum_{a'} e^{Q(s,a')/\tau}}$$

where:

$\pi(a|s)$ is the probability of selecting action $a$from state s,
$Q(s, a)$ is the estimated action value for state s and action a,
τ is the temperature parameter controlling the level of exploration.

The softmax function ensures that actions with higher estimated values receive higher probabilities, and the temperature parameter controls the stochasticity of the policy. A higher temperature leads to a more uniform distribution (more exploration), while a lower temperature emphasizes the action with the maximum value (more exploitation).

For each experiment, we compare the rewards from these two policies and present the results of the best policy.

We experiment with different values of hyperparameters $\alpha, \gamma, \epsilon, \tau$ for each environment setting to find the best performing hyperparameters.

- We vary τ in [0.01, 0.1, 1, 10], α in [0.4, 0.5, 0.6], γ in [0.7, 0.8, 0.9, 1] for softmax policy.
- We vary ϵ in [0.001, 0.01, 0.1, 0.5, 0.75], α in [0.4, 0.5, 0.6], γ in [0.7, 0.8, 0.9, 1] for epsilon greedy policy.

We train every algorithm with the chosen policy for 5000 episodes and present the results averaged over 5 runs, with different seeds for each run.

3 Starting State: (3,6)

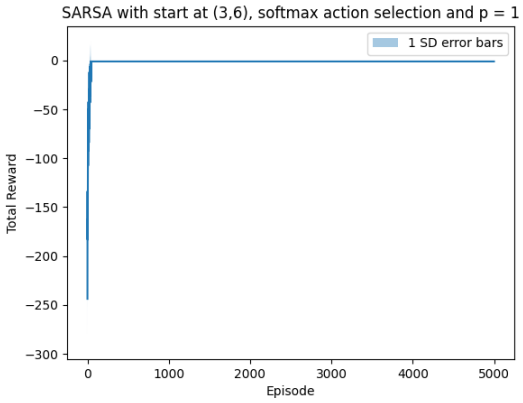
3.1 Grid World Variant: wind = False and p = 1.0

3.1.1 SARSA

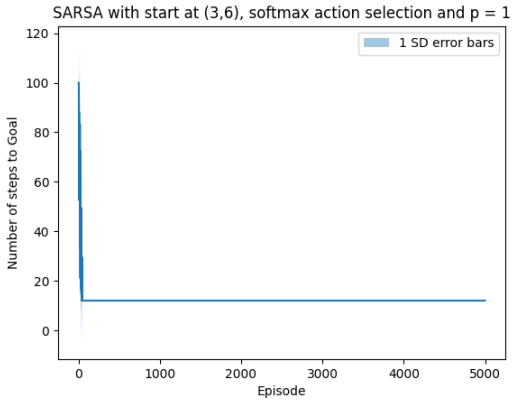
We observe that the best average reward (averaged over 5 runs and over all episodes) is **-1.52944** for the softmax action selection policy and **-1.60132** for the epsilon-greedy action selection policy. Since the best reward for softmax is higher, we conclude that it is the better policy to use for this experiment.

Thus, the best configuration (from our experiments) is the Softmax action selection policy with $\tau = 0.01$, $\alpha = 0.6$ and $\gamma = 0.9$.

We give the plots of the total reward for an episode (averaged over 5 runs) v/s episode number and number of steps in an episode (averaged over 5 runs) v/s episode number for the best configuration below. We also depict the 1 standard deviation error bars in the figure.

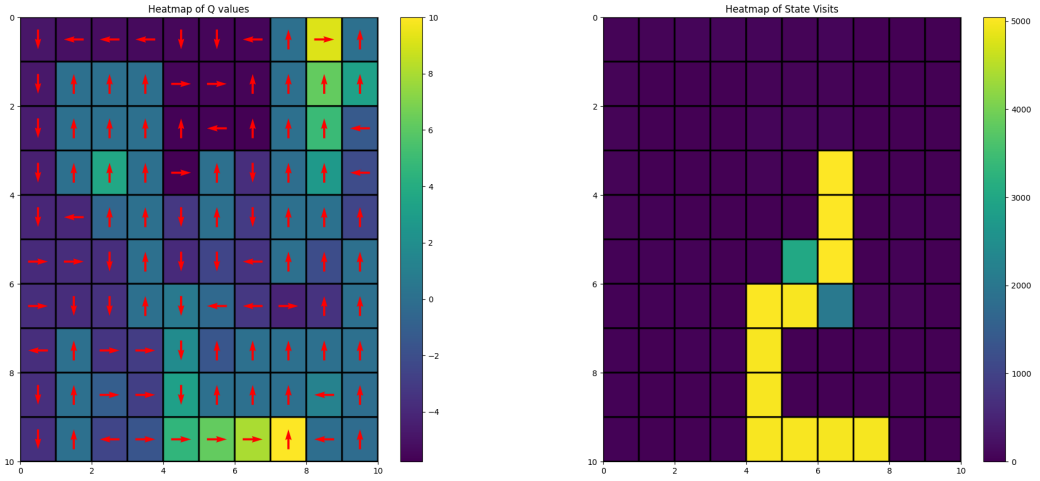


(a) Plot of total reward v/s episode number



(b) Plot of no. of steps v/s episode number

Figure 2: Metric plots for SARSA algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.



(a) Heatmap of Q values

(b) Heatmap of number of visits per state

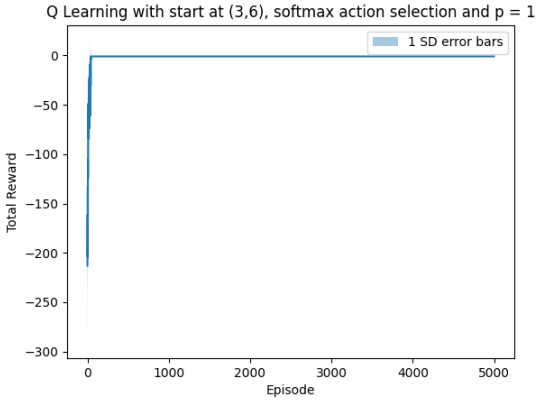
Figure 3: Q values and no. of visits per state for SARSA algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.

3.1.2 Q Learning

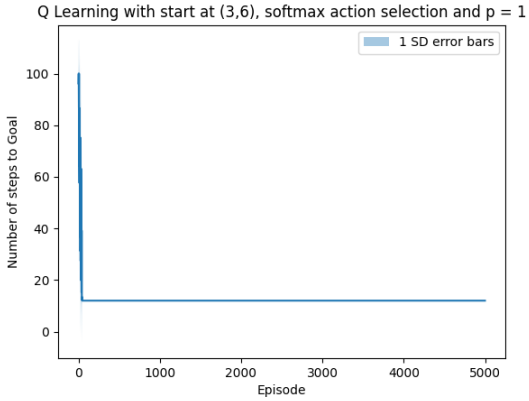
We observe that the best configuration average reward (averaged over 5 runs and over all episodes) is **-1.52236** for the Softmax action selection policy and **-1.59488** for the Epsilon-greedy action selection policy. Since the best reward for Softmax is higher, we conclude that it is the better policy to use for this experiment.

Thus, the best configuration (from our experiments) is the Softmax action selection policy with $\tau = 0.1, \alpha = 0.6$ and $\gamma = 1$.

We give the plots of the total reward for an episode (averaged over 5 runs) v/s episode number and number of steps in an episode (averaged over 5 runs) v/s episode number for the best configuration below. We also depict the 1 standard deviation error bars in the figure.

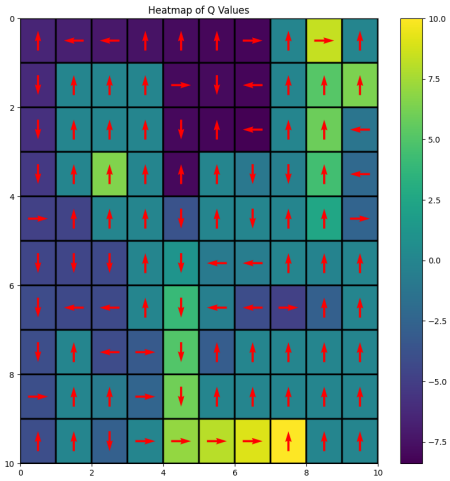


(a) Plot of total reward v/s episode number

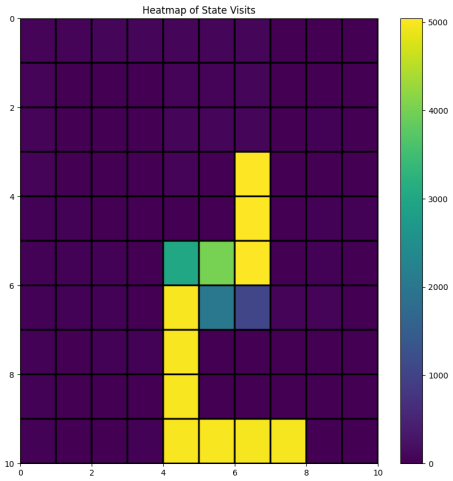


(b) Plot of no. of steps v/s episode number

Figure 4: Metric plots for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.



(a) Heatmap of Q values



(b) Heatmap of number of visits per state

Figure 5: Q values and no. of visits per state for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.

3.1.3 Observations

- From the best average rewards for both algorithms, we conclude that SARSA and Q Learning perform very similarly for this environment setting, with Q Learning having a marginally better average reward.
- From the state visitation heatmaps in 3b and 5b, we observe that the paths explored by SARSA and Q Learning are very similar. However, the (5,4) state is visited much more often in Q Learning than in SARSA. We hypothesize that this is because this state lies adjacent to the Bad state at (4,4). As mentioned earlier, SARSA tries to

avoid dangerous paths, while Q Learning looks for the shortest path. Hence, SARSA avoids this state, due to the danger of transitioning into the Bad state.

- We also observe that the two algorithms converge to the same optimal path, which is coherent with our previous observation of similar average rewards and visitation heatmaps.
- From 2a and 2b we observe that there are no visible errorbars because the transition probability is deterministic ($p = 1$).

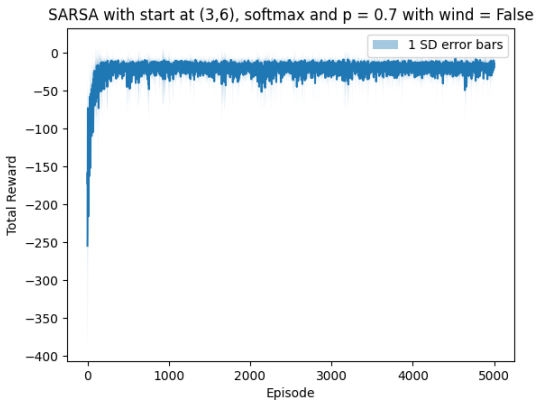
3.2 Grid World Variant: wind = False and p = 0.7

3.2.1 SARSA

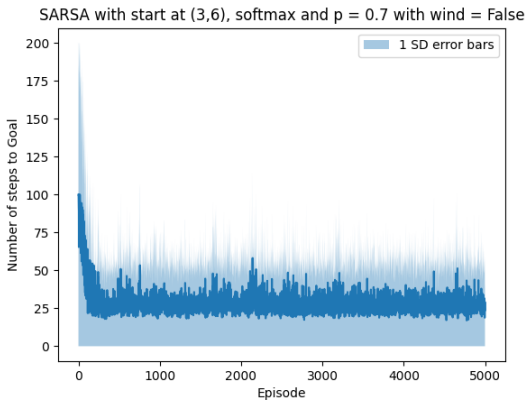
We observe that the best average reward (averaged over 5 runs and over all episodes) is **-20.6416** for the Softmax action selection policy and **-21.49352** for the Epsilon-greedy action selection policy. Since the best reward for Softmax is higher, we conclude that it is the better policy to use for this experiment.

Thus, the best configuration (from our experiments) is the Softmax action selection policy with $\tau = 0.01, \alpha = 0.4$ and $\gamma = 0.9$.

We give the plots of the total reward for an episode (averaged over 5 runs) v/s episode number and number of steps in an episode (averaged over 5 runs) v/s episode number for the best configuration below. We also depict the 1 standard deviation error bars in the figure.

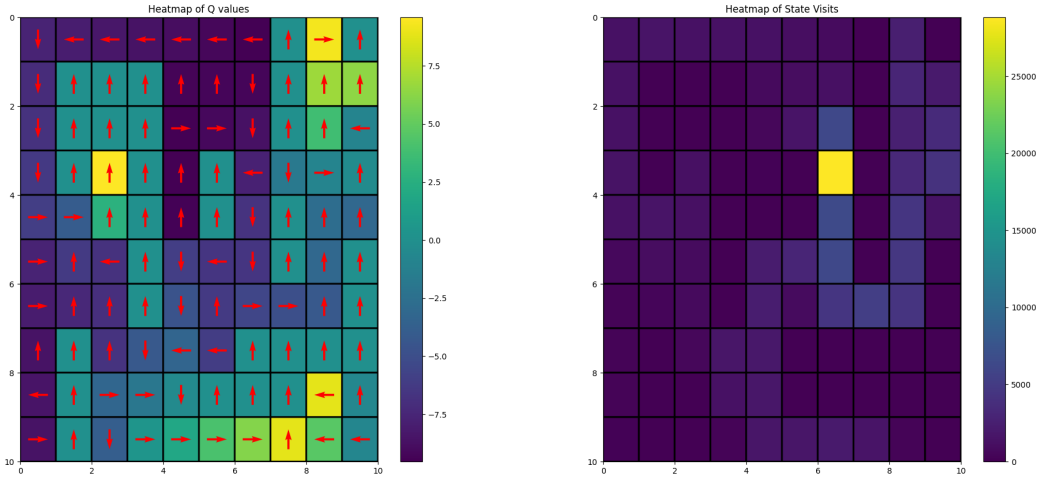


(a) Plot of total reward v/s episode number



(b) Plot of no. of steps v/s episode number

Figure 6: Metric plots for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.



(a) Heatmap of Q values

(b) Heatmap of number of visits per state

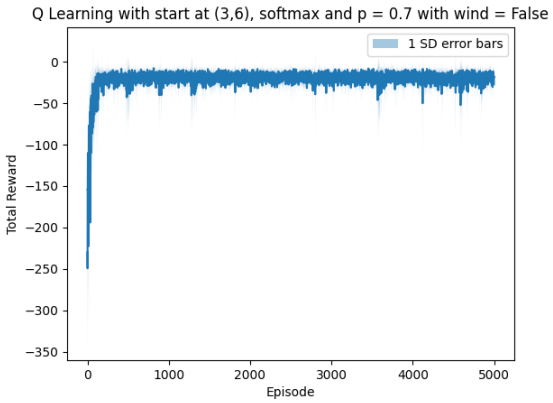
Figure 7: Q values and no. of visits per state for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.

3.2.2 Q Learning

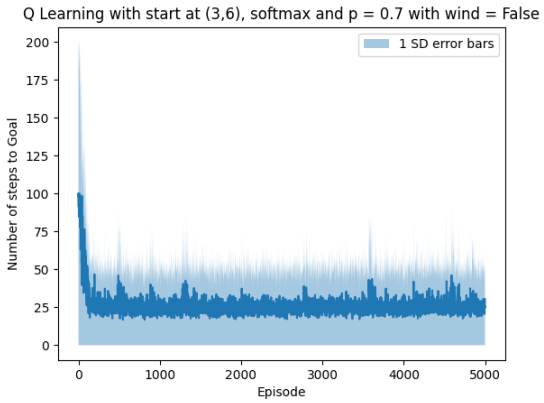
We observe that the best average reward (averaged over 5 runs and over all episodes) is **-20.65416** for the Softmax action selection policy and **-20.90648** for the Epsilon-greedy action selection policy. Since the best reward for Softmax is higher, we conclude that it is the better policy to use for this experiment.

Thus, the best configuration (from our experiments) is the Softmax action selection policy with $\tau = 0.1, \alpha = 0.4$ and $\gamma = 1$.

We give the plots of the total reward for an episode (averaged over 5 runs) v/s episode number and number of steps in an episode (averaged over 5 runs) v/s episode number for the best configuration below. We also depict the 1 standard deviation error bars in the figure.

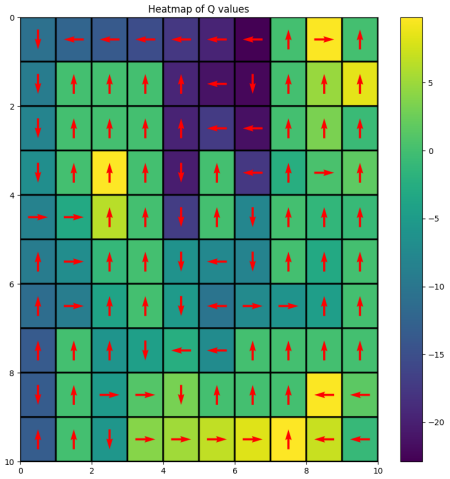


(a) Plot of total reward v/s episode number

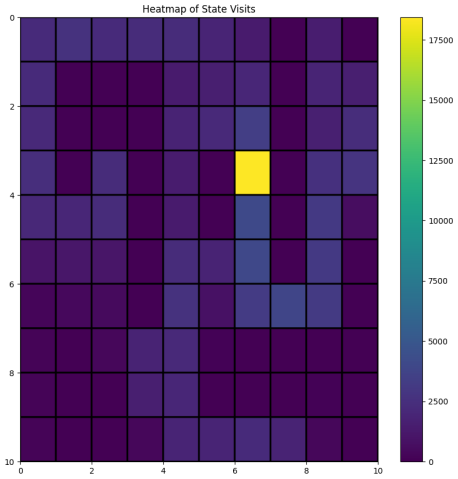


(b) Plot of no. of steps v/s episode number

Figure 8: Metric plots for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.



(a) Heatmap of Q values



(b) Heatmap of number of visits per state

Figure 9: Q values and no. of visits per state for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.

3.2.3 Observations

- From the best average rewards for both algorithms, we conclude that SARSA and Q Learning perform very similarly for this environment setting, with SARSA having a marginally better average reward.
- From the state visitation heatmaps in 7b and 9b, we observe that Q Learning explores more paths and terminal states than the SARSA algorithm.
- Specifically, we observe that Q Learning explores the path to the goal state at (2,2) much more than SARSA. Once again, we hypothesize that this is because the path

to this goal state contains a Bad state. As we know, SARSA is known to avoid risky paths, hence it explores this route much lesser than Q Learning.

- We also observe that the two algorithms ultimately converge to the same optimal path, which is coherent with our previous observation of similar average rewards.

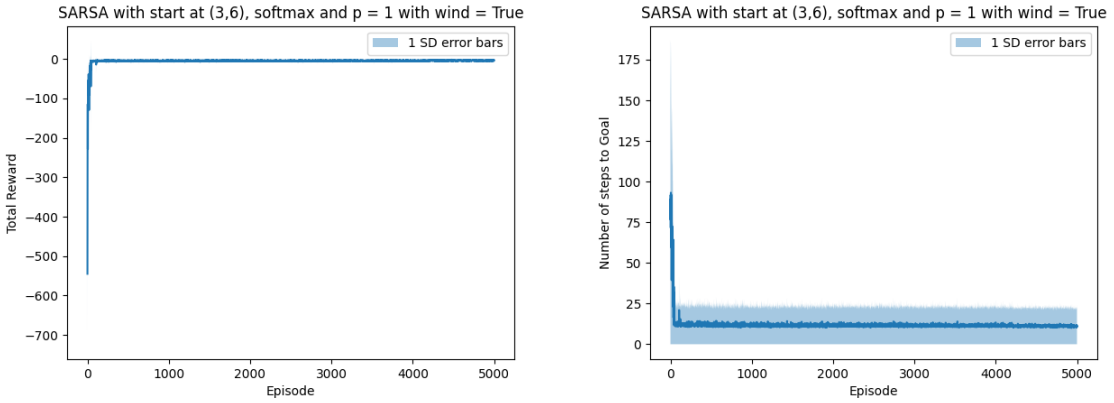
3.3 Grid World Variant: wind = True and p = 1

3.3.1 SARSA

We observe that the best average reward (averaged over 5 runs and over all episodes) is **-4.46236** for the Softmax action selection policy and **-6.4736** for the Epsilon-greedy action selection policy. Since the best reward for Softmax is higher, we conclude that it is the better policy to use for this experiment.

Thus, the best configuration (from our experiments) is the Softmax action selection policy with $\tau = 0.01, \alpha = 0.4$ and $\gamma = 0.8$.

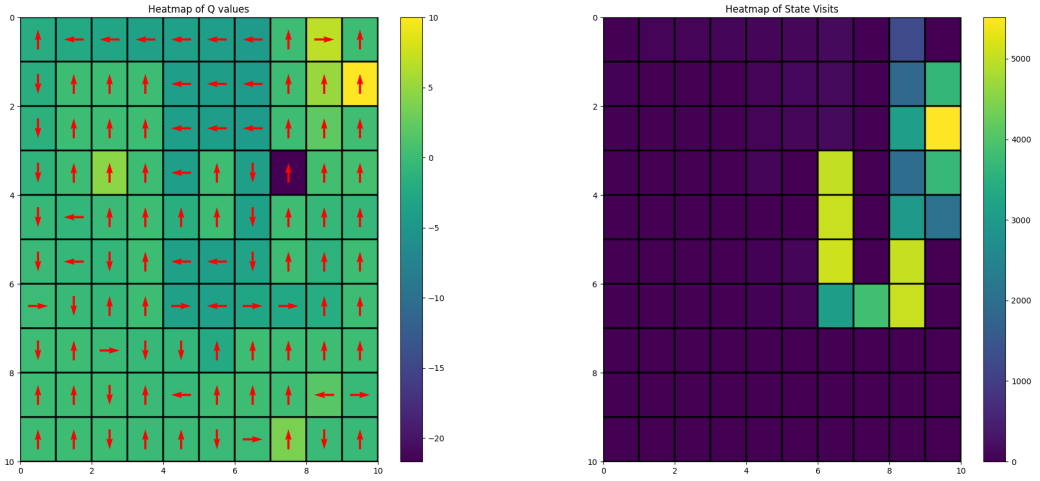
We give the plots of the total reward for an episode (averaged over 5 runs) v/s episode number and number of steps in an episode (averaged over 5 runs) v/s episode number for the best configuration below. We also depict the 1 standard deviation error bars in the figure.



(a) Plot of total reward v/s episode number

(b) Plot of no. of steps v/s episode number

Figure 10: Metric plots for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.



(a) Heatmap of Q values

(b) Heatmap of number of visits per state

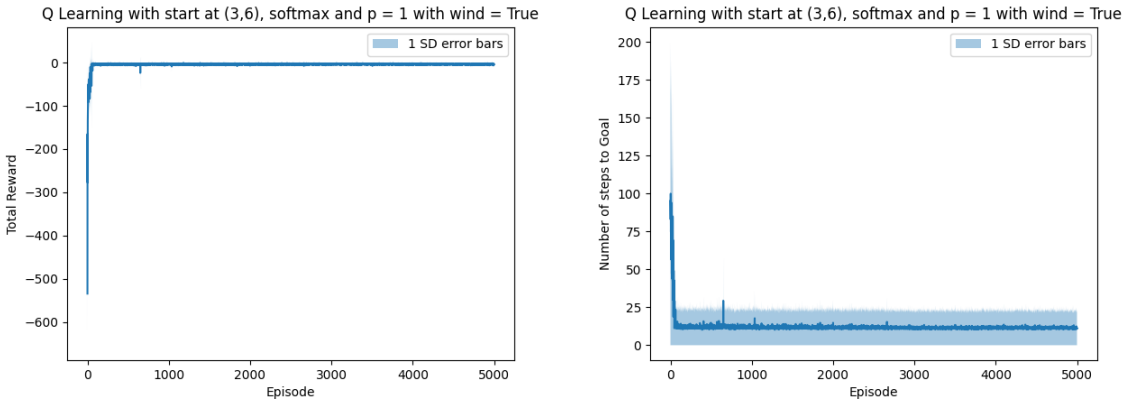
Figure 11: Q values and no. of visits per state for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.

3.3.2 Q Learning

We observe that the best average reward (averaged over 5 runs and over all episodes) is **-4.45596** for the Softmax action selection policy and **-4.6066** for the Epsilon-greedy action selection policy. Since the best reward for Softmax is higher, we conclude that it is the better policy to use for this experiment.

Thus, the best configuration (from our experiments) is the Softmax action selection policy with $\tau = 0.01, \alpha = 0.4$ and $\gamma = 0.7$.

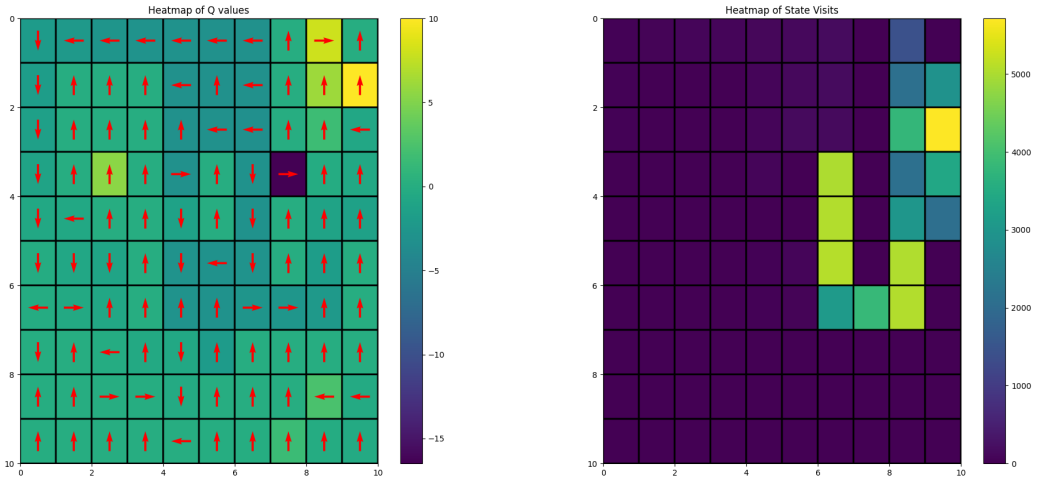
We give the plots of the total reward for an episode (averaged over 5 runs) v/s episode number and number of steps in an episode (averaged over 5 runs) v/s episode number for the best configuration below. We also depict the 1 standard deviation error bars in the figure.



(a) Plot of total reward v/s episode number

(b) Plot of no. of steps v/s episode number

Figure 12: Metric plots for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.



(a) Heatmap of Q values

(b) Heatmap of number of visits per state

Figure 13: Q values and no. of visits per state for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.

3.3.3 Observations

- From the best average rewards for both algorithms, we conclude that SARSA and Q Learning perform very similarly for this environment setting, with Q Learning having a marginally better average reward.
- From the state visitation heatmaps in 11b and 13b, we observe that the paths explored by SARSA and Q Learning are very similar.
- We observe that the two algorithms ultimately converge to the same optimal path,

which is coherent with our previous observation of similar average rewards and state visitation heatmaps.

- However, the goal state that the agents converge to in this case is (0,9), different from the goal state of (8,7) that the agents converged to in the deterministic no wind experiment in Section 3.1.1. We hypothesize that this may be due to the additional rightward wind in this setting, pushing the agent towards the goal state on the right of the board.
- For the same reason (rightward wind), we also observe that the path to the goal through Column 9 is more visited than the path through Column 8, even though the path through Column 9 has a bad state in it.

4 Starting State: (0,4)

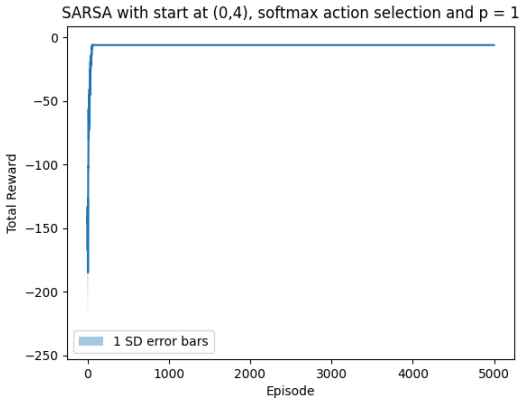
4.1 Grid World Variant: wind = False and p = 1.0

4.1.1 SARSA

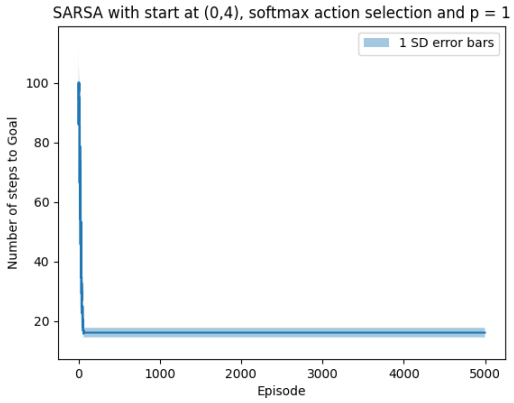
We observe that the best average reward (averaged over 5 runs and over all episodes) is **-6.57792** for the Softmax action selection policy and **-6.96128** for the Epsilon-greedy action selection policy. Since the best reward for Softmax is higher, we conclude that it is the better policy to use for this experiment.

Thus, the best configuration (from our experiments) is the Softmax action selection policy with $\tau = 0.01, \alpha = 0.6$ and $\gamma = 1$.

We give the plots of the total reward for an episode (averaged over 5 runs) v/s episode number and number of steps in an episode (averaged over 5 runs) v/s episode number for the best configuration below. We also depict the 1 standard deviation error bars in the figure.

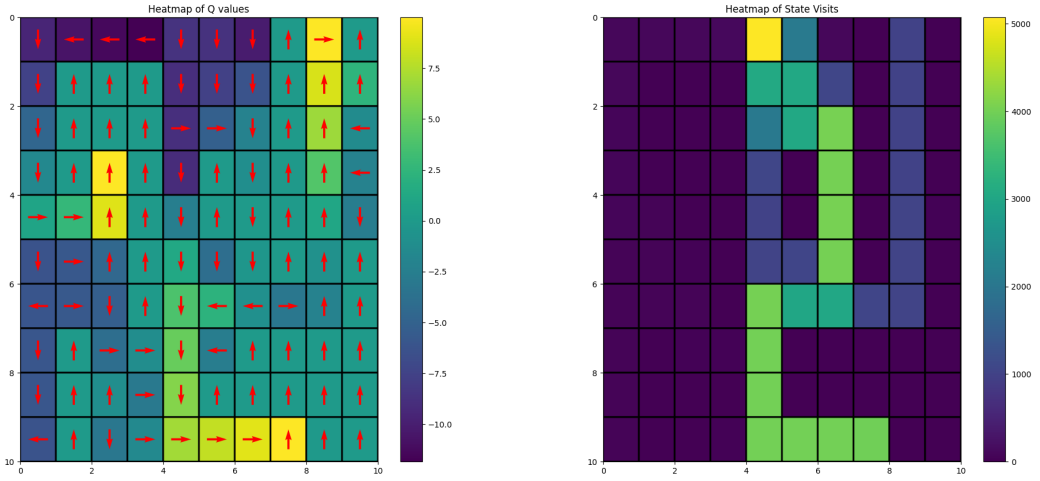


(a) Plot of total reward v/s episode number



(b) Plot of no. of steps v/s episode number

Figure 14: Metric plots for SARSA algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.



(a) Heatmap of Q values

(b) Heatmap of number of visits per state

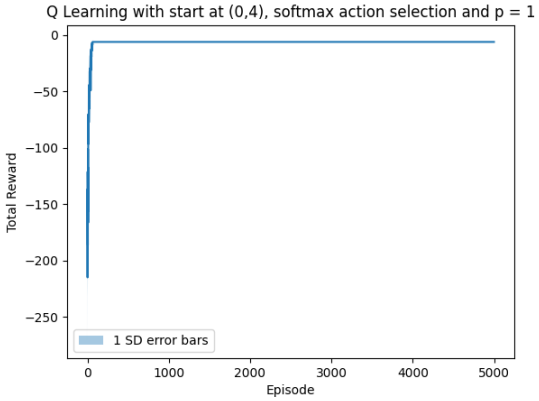
Figure 15: Q values and no. of visits per state for SARSA algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.

4.1.2 Q Learning

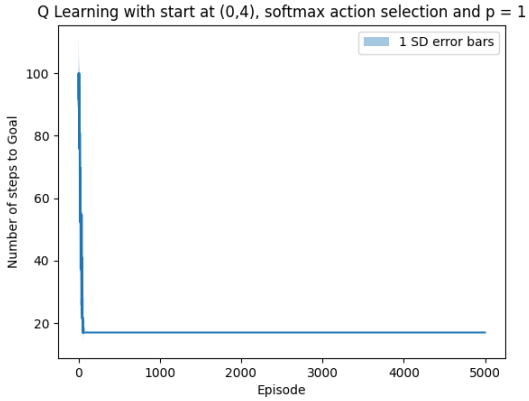
We observe that the best average reward (averaged over 5 runs and over all episodes) is **-6.5719199** for the Softmax action selection policy and **-6.6330399** for the Epsilon-greedy action selection policy. Since the best reward for Softmax is higher, we conclude that it is the better policy to use for this experiment.

Thus, the best configuration (from our experiments) is the Softmax action selection policy with $\tau = 0.1, \alpha = 0.6$ and $\gamma = 1$.

We give the plots of the total reward for an episode (averaged over 5 runs) v/s episode number and number of steps in an episode (averaged over 5 runs) v/s episode number for the best configuration below. We also depict the 1 standard deviation error bars in the figure.

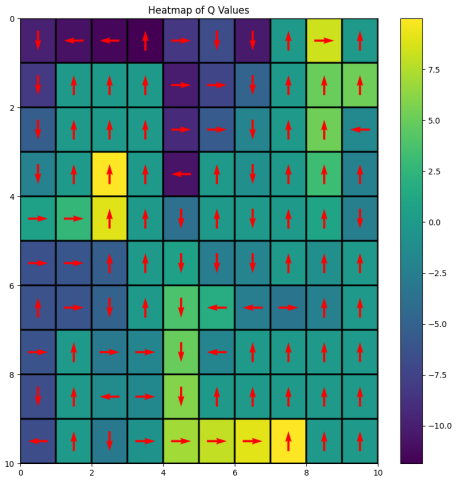


(a) Plot of total reward v/s episode number

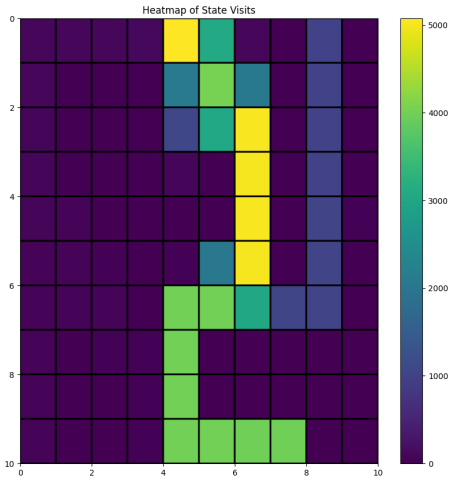


(b) Plot of no. of steps v/s episode number

Figure 16: Metric plots for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.



(a) Heatmap of Q values



(b) Heatmap of number of visits per state

Figure 17: Q values and no. of visits per state for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.

4.1.3 Observations

- From the best average rewards for both algorithms, we conclude that SARSA and Q Learning perform very similarly for this environment setting, with Q Learning having a marginally better average reward.
- From the state visitation heatmaps in 15b and 17b, we observe that SARSA and Q Learning have similar heatmaps, with SARSA exploring one extra path from (3,4) to (5,4).

- We also observe that the two algorithms ultimately converge to the same optimal path, which is coherent with our previous observation of similar average rewards.

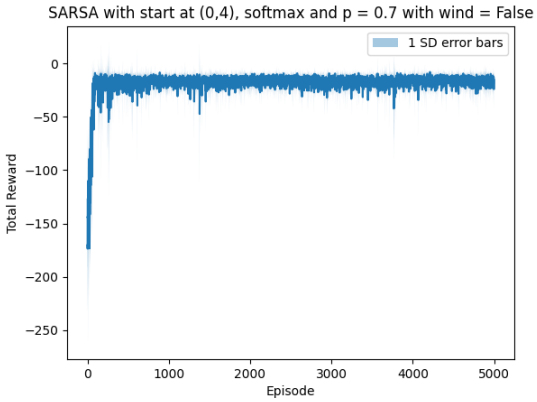
4.2 Grid World Variant: wind = False and p = 0.7

4.2.1 SARSA

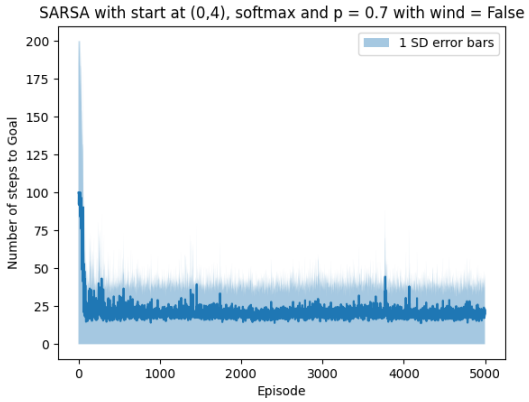
We observe that the best average reward (averaged over 5 runs and over all episodes) is **-16.80056** for the Softmax action selection policy and **-17.43808** for the Epsilon-greedy action selection policy. Since the best reward for Softmax is higher, we conclude that it is the better policy to use for this experiment.

Thus, the best configuration (from our experiments) is the Softmax action selection policy with $\tau = 0.1, \alpha = 0.4$ and $\gamma = 1$.

We give the plots of the total reward for an episode (averaged over 5 runs) v/s episode number and number of steps in an episode (averaged over 5 runs) v/s episode number for the best configuration below. We also depict the 1 standard deviation error bars in the figure.

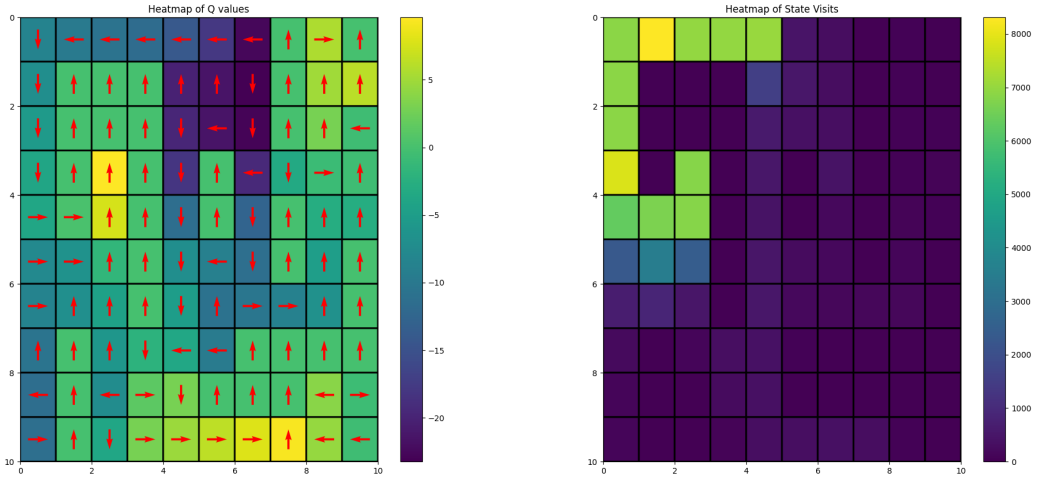


(a) Plot of total reward v/s episode number



(b) Plot of no. of steps v/s episode number

Figure 18: Metric plots for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.



(a) Heatmap of Q values

(b) Heatmap of number of visits per state

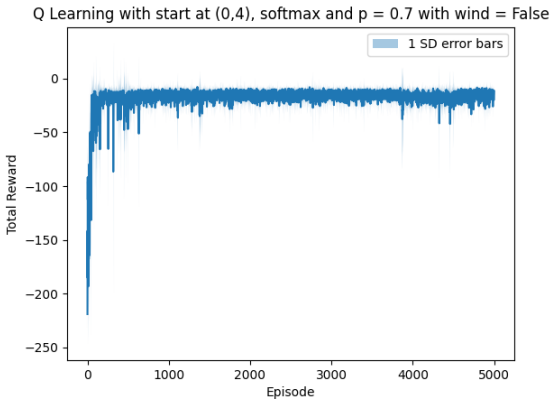
Figure 19: Q values and no. of visits per state for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.

4.2.2 Q Learning

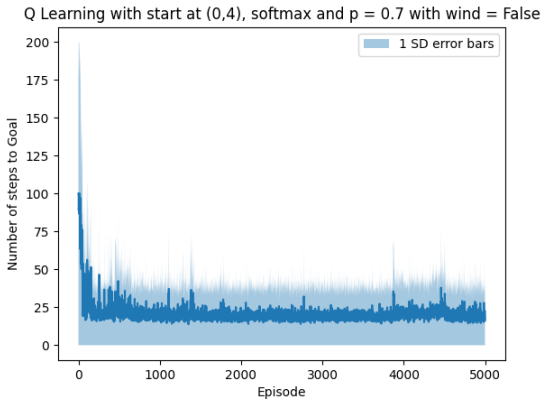
We observe that the best average reward (averaged over 5 runs and over all episodes) is **-17.07112** for the Softmax action selection policy and **-17.79824** for the Epsilon-greedy action selection policy. Since the best reward for Softmax is higher, we conclude that it is the better policy to use for this experiment.

Thus, the best configuration (from our experiments) is the Softmax action selection policy with $\tau = 0.1, \alpha = 0.5$ and $\gamma = 1$.

We give the plots of the total reward for an episode (averaged over 5 runs) v/s episode number and number of steps in an episode (averaged over 5 runs) v/s episode number for the best configuration below. We also depict the 1 standard deviation error bars in the figure.

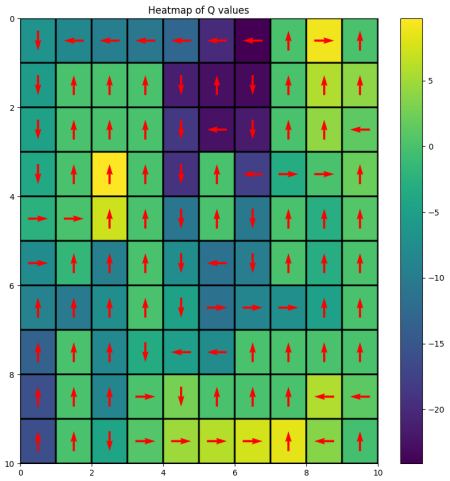


(a) Plot of total reward v/s episode number

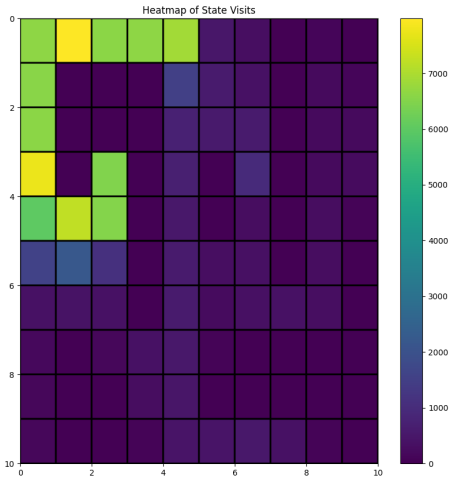


(b) Plot of no. of steps v/s episode number

Figure 20: Metric plots for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.



(a) Heatmap of Q values



(b) Heatmap of number of visits per state

Figure 21: Q values and no. of visits per state for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.

4.2.3 Observations

- From the best average rewards for both algorithms, we observe that SARSA performs better than Q Learning, with SARSA having 0.2 less penalty on an average.
- From the state visitation heatmaps in 19b and 21b, we observe that the paths explored by SARSA and Q Learning are very similar. However, Q Learning has explored far-away states more than the SARSA algorithm.
- We also observe that the two algorithms ultimately converge to the same optimal path,

which is coherent with our previous observation of similar average rewards and state visitation heatmaps.

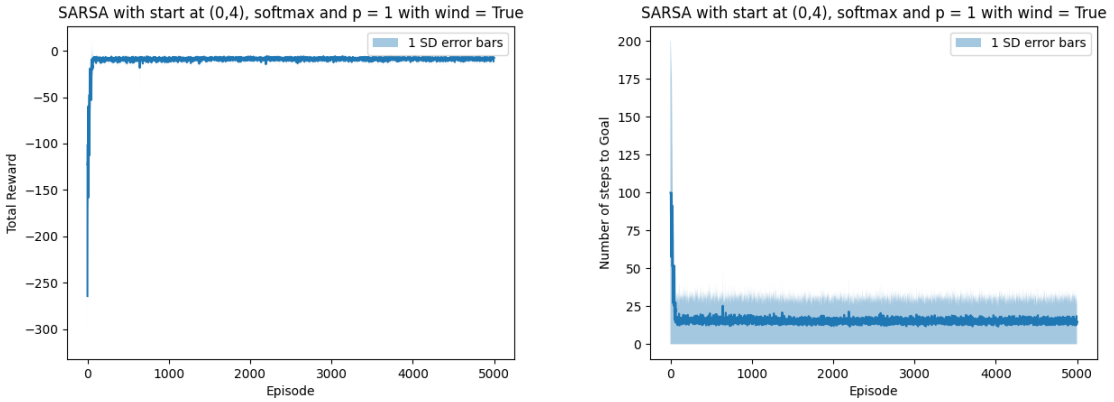
4.3 Grid World Variant: wind = True and p = 1

4.3.1 SARSA

We observe that the best average reward (averaged over 5 runs and over all episodes) is **-8.71636** for the Softmax action selection policy and **-8.97728** for the Epsilon-greedy action selection policy. Since the best reward for Softmax is higher, we conclude that it is the better policy to use for this experiment.

Thus, the best configuration (from our experiments) is the Softmax action selection policy with $\tau = 0.1, \alpha = 0.5$ and $\gamma = 1$.

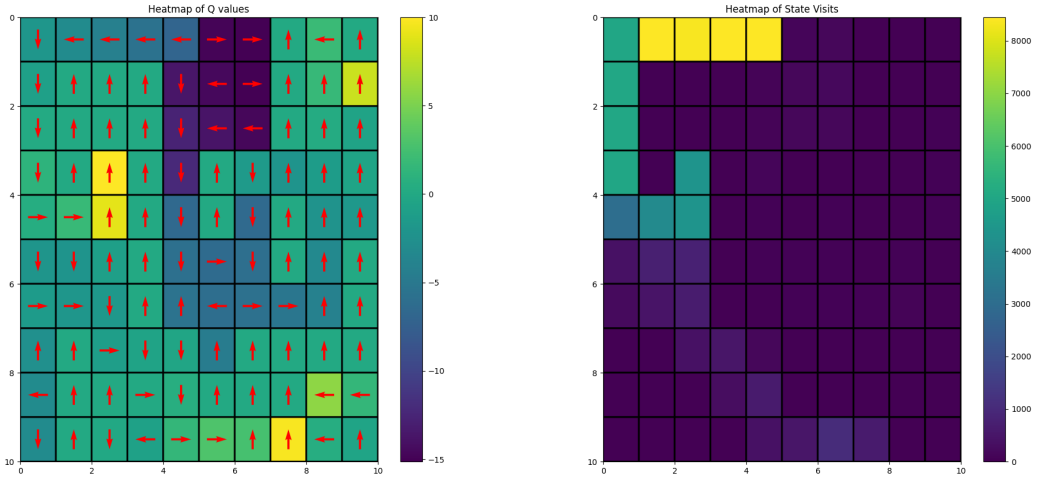
We give the plots of the total reward for an episode (averaged over 5 runs) v/s episode number and number of steps in an episode (averaged over 5 runs) v/s episode number for the best configuration below. We also depict the 1 standard deviation error bars in the figure.



(a) Plot of total reward v/s episode number

(b) Plot of no. of steps v/s episode number

Figure 22: Metric plots for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.



(a) Heatmap of Q values

(b) Heatmap of number of visits per state

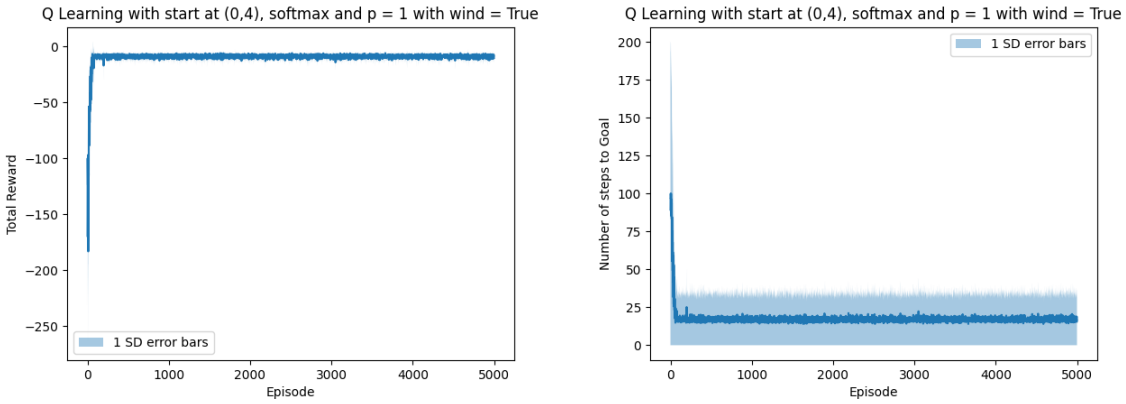
Figure 23: Q values and no. of visits per state for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.

4.3.2 Q Learning

We observe that the best average reward (averaged over 5 runs and over all episodes) is **-8.75332** for the Softmax action selection policy and **-8.79252** for the Epsilon-greedy action selection policy. Since the best reward for Softmax is higher, we conclude that it is the better policy to use for this experiment.

Thus, the best configuration (from our experiments) is the Softmax action selection policy with $\tau = 0.01, \alpha = 0.5$ and $\gamma = 1$.

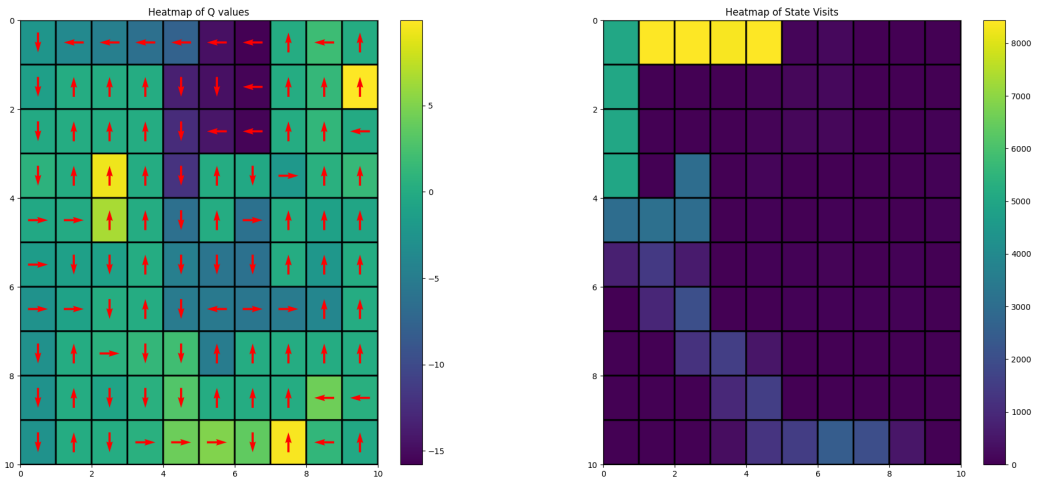
We give the plots of the total reward for an episode (averaged over 5 runs) v/s episode number and number of steps in an episode (averaged over 5 runs) v/s episode number for the best configuration below. We also depict the 1 standard deviation error bars in the figure.



(a) Plot of total reward v/s episode number

(b) Plot of no. of steps v/s episode number

Figure 24: Metric plots for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.



(a) Heatmap of Q values

(b) Heatmap of number of visits per state

Figure 25: Q values and no. of visits per state for Q Learning algorithm with best hyperparameter configuration, averaged over 5 runs. We provide plots only for the best performing action selection policy, which in this case is the Softmax policy.

4.3.3 Observations

- From the best average rewards for both algorithms, we conclude that SARSA and Q Learning perform similarly for this environment setting, with SARSA having a marginally better average reward.
- From the state visitation heatmaps in 23b and 25b, we observe similar optimal path heatmaps.
- Apart from this, there is also a diagonal path from (5,1) to (9,7). We hypothesize that this path is due to the effects of wind. This path is more prominently explored in the

case of Q Learning algorithm, which follows this diagonal to converge to a different terminal state.

- We observe that the two algorithms ultimately converge to the same optimal path, which is coherent with our previous observation of similar average rewards.

5 Conclusion

In all 12 experiments, we observe significant variation in rewards and number of steps taken for the same environment set up and algorithm, depending on the hyperparameters chosen.

We manage to converge to the optimal path within 5000 episodes through hyperparameter tuning for both SARSA and Q Learning algorithms. We also observe that in both algorithms, the restart states are very well avoided, with barely any visits in all 12 to these states in all 12 experiments.

We also observe that the softmax action-selection policy outperforms the epsilon-greedy policy in all our experiments. We hypothesize that this might be because softmax action probabilities are weighted by their Q values. Thus, this policy explores higher Q states more thoroughly and comes up with better estimates for their Q values.