Programming Assignment 1
CS6700

SARSA and Q Learning

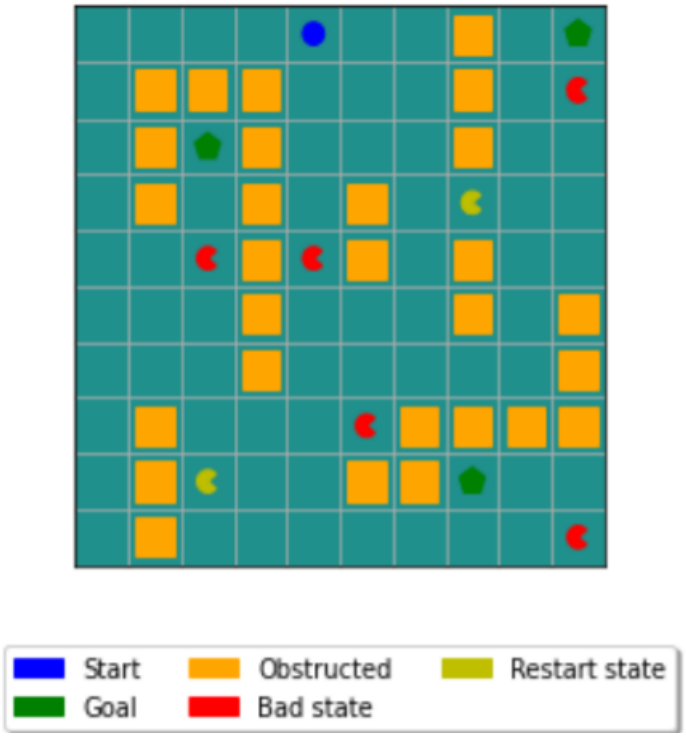
February 28, 2024

Balakumar R ME20B043

Girish Madhavan V ME20B072

Contents

1 Problem Statement



The dimensions of the grid are **10 × 10**. The following types of states exist:

- **Start state:** The agent starts from this state.
- **Goal state:** The goal is to reach one of these states. There are 3 goal states in total. **Reward = +10**
- **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. **Reward = -6**
- **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. **Reward = -100**
- **Normal state:** Entry into these states will incur a small penalty. **Reward = -1**

1.1 Algorithms

Temporal Difference (TD) Learning algorithms represent a powerful class of reinforcement learning techniques that have proven instrumental in addressing problems related to sequential decision-making and prediction tasks. These algorithms bridge the gap between dynamic programming and Monte Carlo methods, offering a computationally efficient and online approach to learning from experience. Two popular TD algorithms:

1.1.1 SARSA

SARSA algorithm learns to estimate the value of state-action pairs by iteratively updating these estimates based on observed transitions in the environment."SARSA", reflects the sequence of information used in the update process: the current state (S), the action taken in that state (A), the reward received as a result of that action (R), the next state reached (S'), and the subsequent action taken (A').

SARSA is an on-policy algorithm, that it updates its action-value estimates while following the same policy that it is trying to improve. The SARSA update rule is based on the Bellman equation for action values and is expressed as follows:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \tag{1}$$

Here, α denotes the learning rate, r_t is the immediate reward, γ is the discount factor, s_{t+1} is the next state and a_{t+1} is the subsequent action.

1.1.2 Q-Learning

Q-learning is to learn an optimal action-value function $Q(s, a)$, where s is the state , a is the action and in order to guide an agent in making decisions that maximize cumulative rewards over time.

The off-policy exploration strategy allows Q-learning to explore the environment using a distinct policy while concurrently learning the optimal policy. This decoupling enhances flexibility, enabling the algorithm to efficiently balance exploration and exploitation.The SARSA update rule is based on the Bellman equation for action values and is expressed as follows:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_t + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)] \tag{2}$$

NOTE :

- **Why Softmax was chosen as best policy in all experiments ?**

  Softmax assigns probabilities to all available actions based on their estimated values, allowing the algorithm to explore different options while still favoring actions with higher values whereas Epsilon-Greedy, makes a random exploration choice with a fixed probability (ϵ), otherwise exploits the current best-known action. This can lead to a more deterministic exploration strategy, potentially missing out on valuable information.

- **How is the best model selected for every experiment ?**

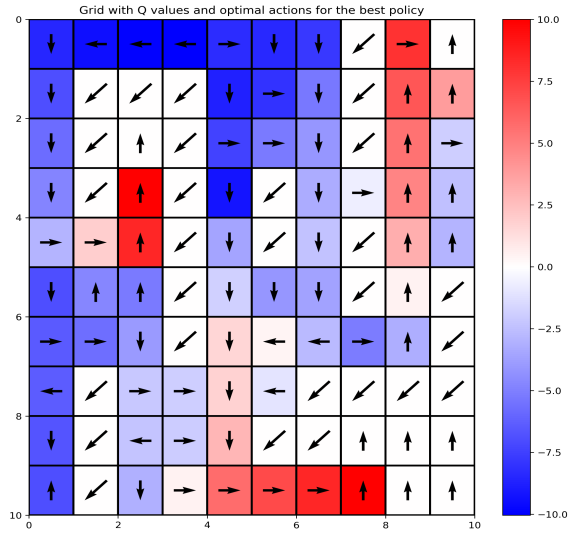
  The Mean Reward is used to choose the best model amongst the various combinations of the hyperparameters

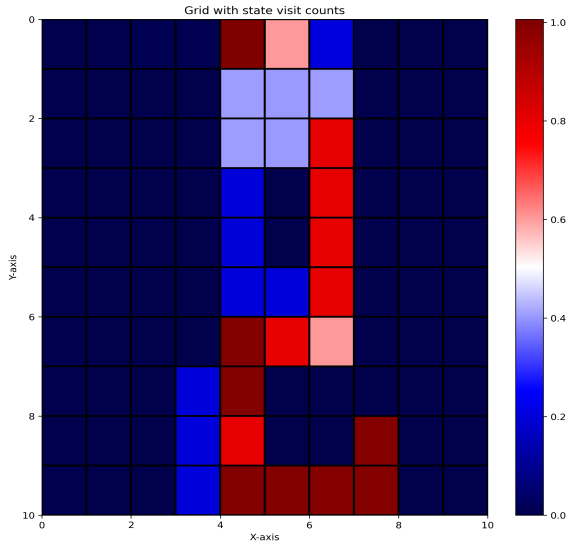
2 Results

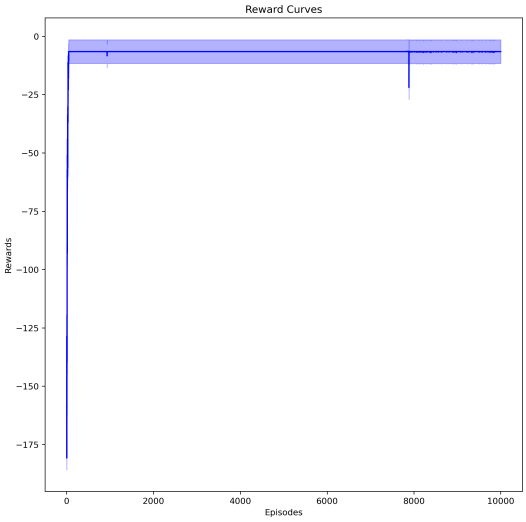
Best Hyperparameters

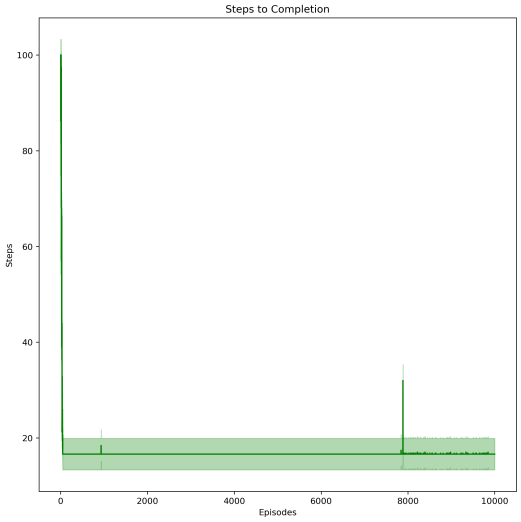
SARSA	Q-Learning
E01: α: 0.91, γ: 0.95, τ: 0.2, *Policy:* Softmax	**E02:** α: 0.28, γ: 0.95, τ: 0.1, *Policy:* Softmax
E03: α: 0.28, γ: 0.95, τ: 0.1, *Policy:* Softmax	**E04:** α: 0.37, γ: 0.95, τ: 0.1, *Policy:* Softmax
E05: α: 0.64, γ: 0.95, τ: 0.2, *Policy:* Softmax	**E06:** α: 0.73, γ: 0.9, τ: 0.1, *Policy:* Softmax
E07: α: 0.73, γ: 0.95, τ: 0.1, *Policy:* Softmax	**E08:** α: 0.28, γ: 0.9, τ: 0.1, *Policy:* Softmax
E09: α: 0.37, γ: 0.95, τ: 0.2, *Policy:* Softmax	**E10:** α: 0.19, γ: 0.95, τ: 0.1, *Policy:* Softmax
E11: α: 0.37, γ: 0.9, τ: 0.1, *Policy:* Softmax	**E12:** α: 0.19, γ: 0.9, τ: 0.2, *Policy:* Softmax

2.1 E01 : Initial State : [0 4], Wind : False, p = 1, Algorithm : SARSA







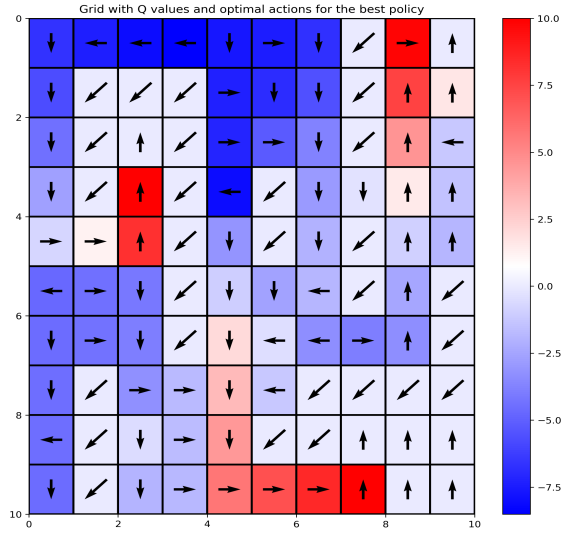


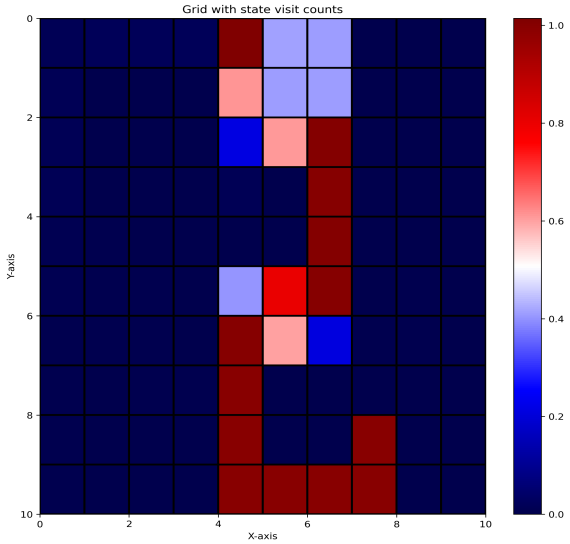
best model $\implies$ mean reward : -6.854, mean steps : 16.7825

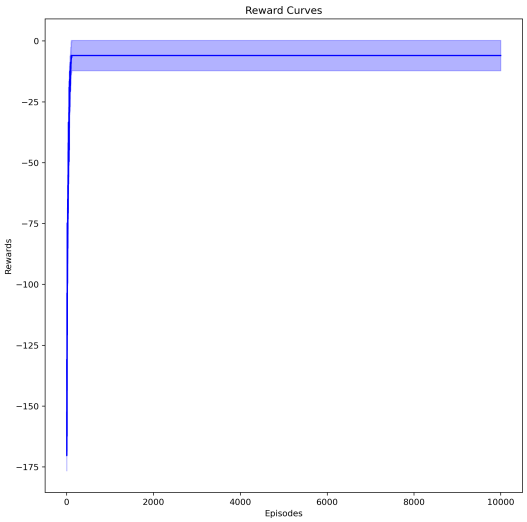
Hyperparameter : $\alpha = 0.91, \gamma = 0.95, \tau = 0.2$& Softmax Policy

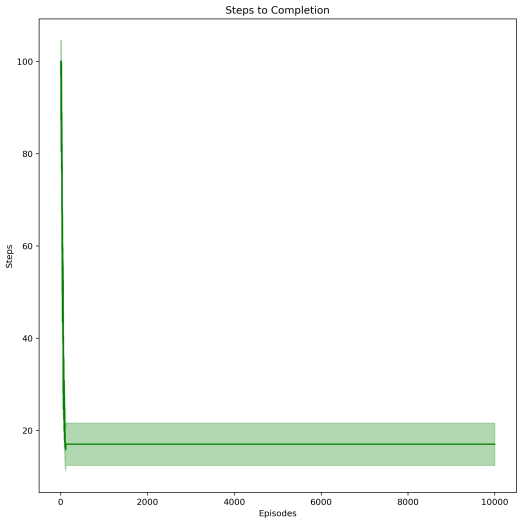
- $\alpha = 0.91$ allows the algorithm to update Q-values more significantly based on new information from each step which leads to faster convergence.
- $\tau = 0.2$ suggest that higher average return will have a higher chance of being chosen in the exploration phase

2.2 E02 : Initial State : [0 4], Wind : False, p = 1, Algorithm : Q-Learning









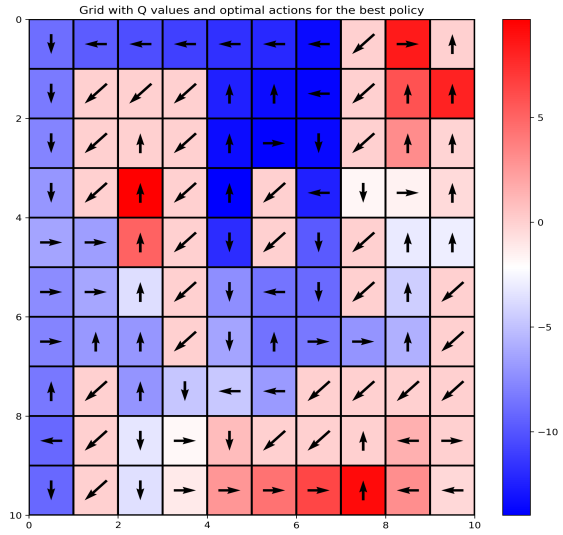
best model $\implies$ mean reward : -6.461, mean steps : 17.361

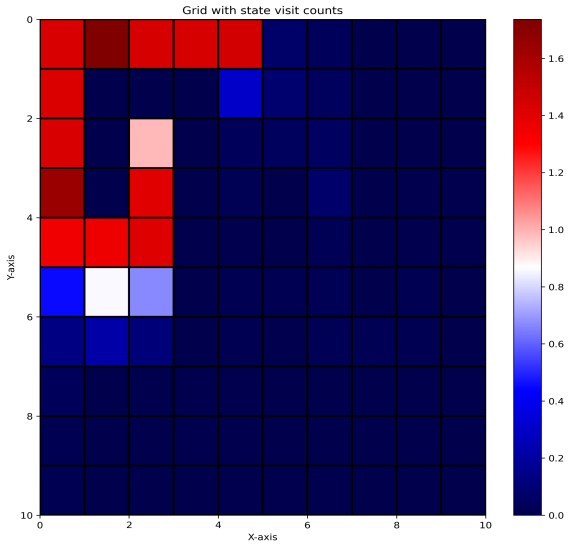
Hyperparameters $\alpha = 0.28, \gamma = 0.95, \tau = 0.1,$ & Softmax Policy end up giving the best model

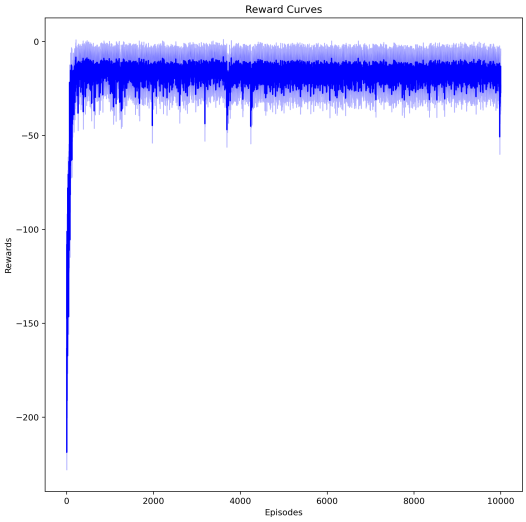
Comparision between E01 and E02

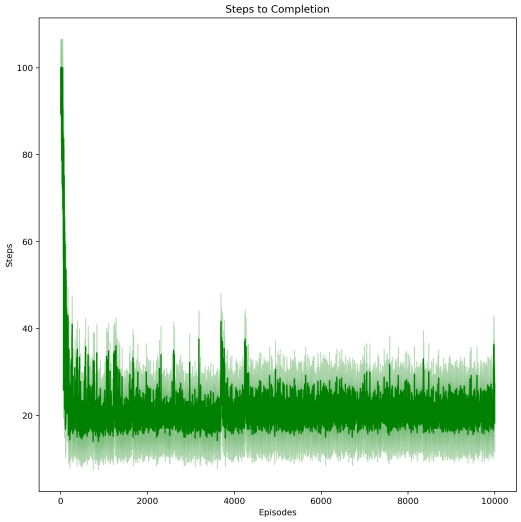
- In this given start state, the optimal path and the safest path for reaching the end state of `[8 7]` are the same. This is due to the presence of the restart state at `[8 2]` which is clearly avoided by the SARSA algorithm, similarly Q-Learning avoids the state leads to the shortest path to the terminal state.
- `SARSA` being on-policy uses higher learning rate than `Q-Learning`, which is an off-policy algorithm, thus the former adapts to its policy quickly while the latter strives to learn the optimal policy.

2.3 E03 : Initial State : [0 4], Wind : False, p = 0.7, Algorithm : SARSA







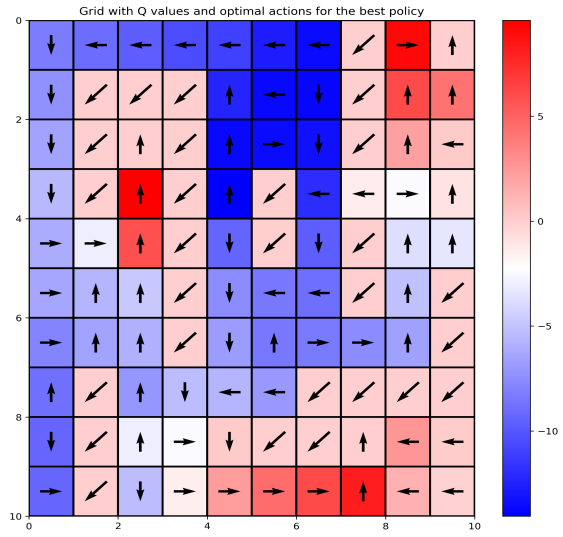


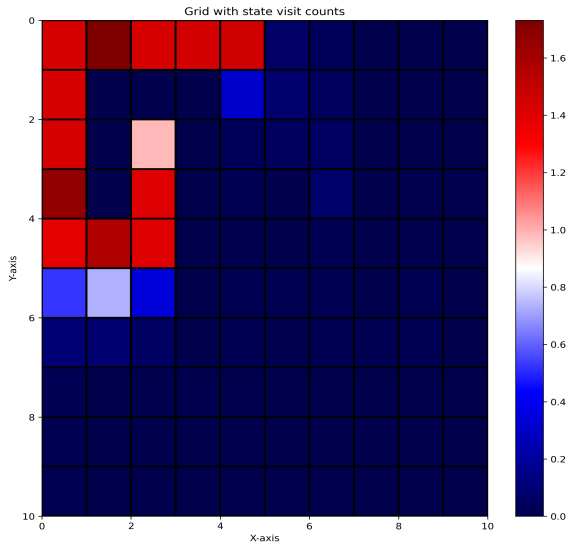
best model $\implies$ mean reward : -17.549, mean steps : 21.207

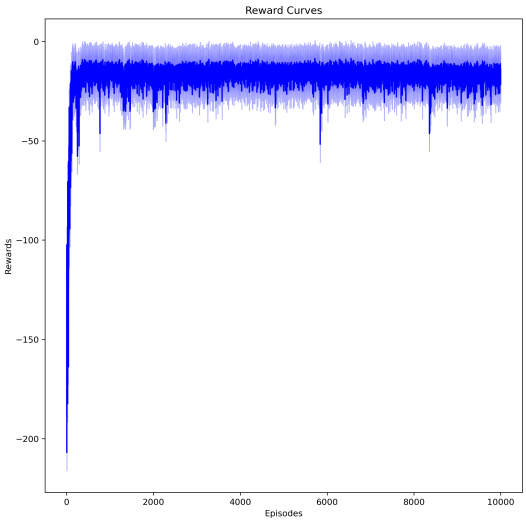
Hyper parameters : $\alpha = 0.28, \gamma = 0.95, \tau = 0.1$& Softmax policy, yielded the best model for this experiment

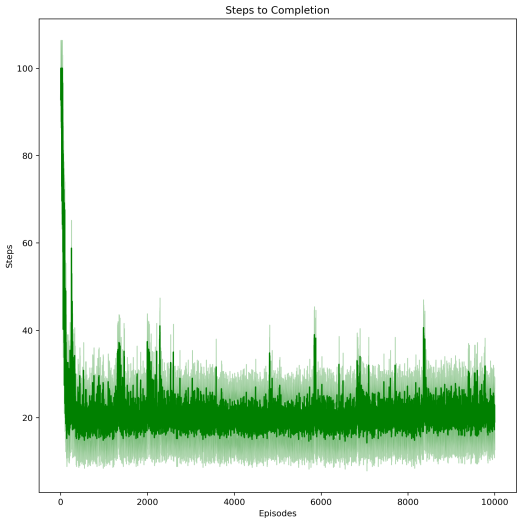
Due to the stochasticity in the actions of the agent, it tries to take a path that squeezes between the obstacles thereby reaching [2 2] and avoiding the chance of stepping on a restart state or bad state accidently.

2.4 E04 : Initial State : [0 4], Wind : False, p = 0.7, Algorithm : Q-Learning









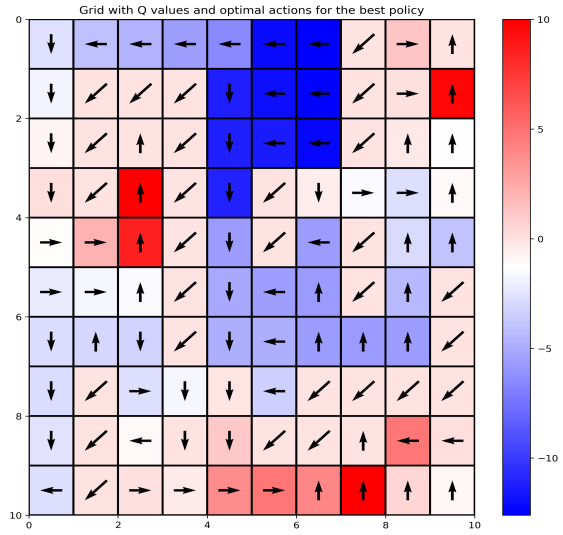
best model $\implies$ mean reward : -17.177, mean steps : 20.844

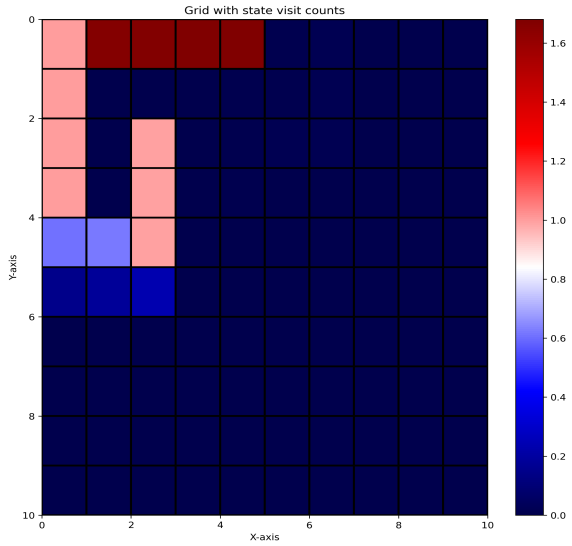
Hyperparameters : $\alpha = 0.37, \gamma = 0.95, \tau = 0.1$& Softmax policy

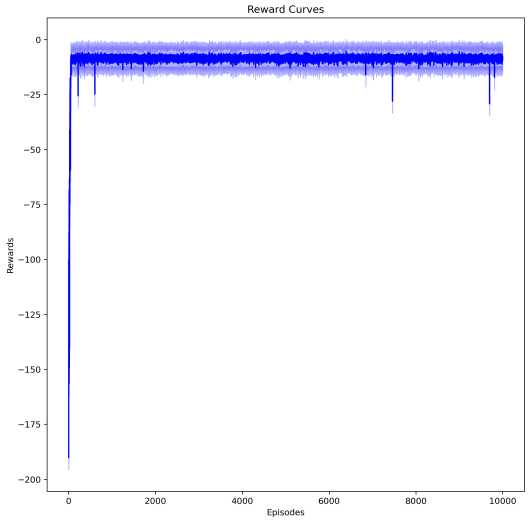
Comparision between E03 and E04 :

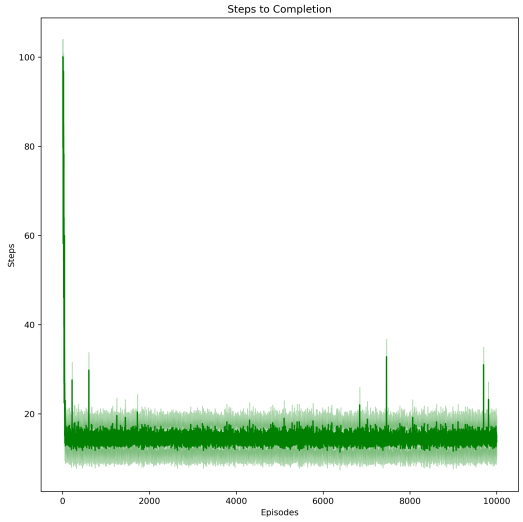
- The learning rate for SARSA is smaller than E01 due to the stochastic nature of the actions while E01 encountered deterministic actions.
- SARSA cautiously updates its estimates due to the stochastic nature of the environment whereas Q-Learning strives to find optimal policy [off-policy algorithm], thus SARSA's learning rate is less than Q-Learning
- The selection policy and discount factor (γ) are similar for both cases

2.5 E05 : Initial State : [0 4], Wind : True, p = 1, Algorithm : SARSA







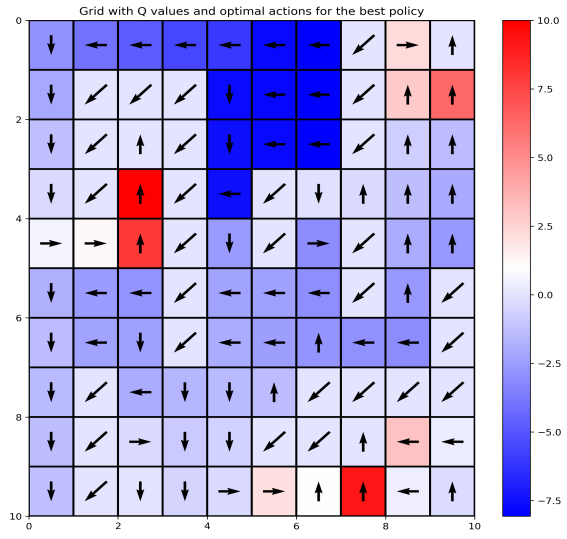


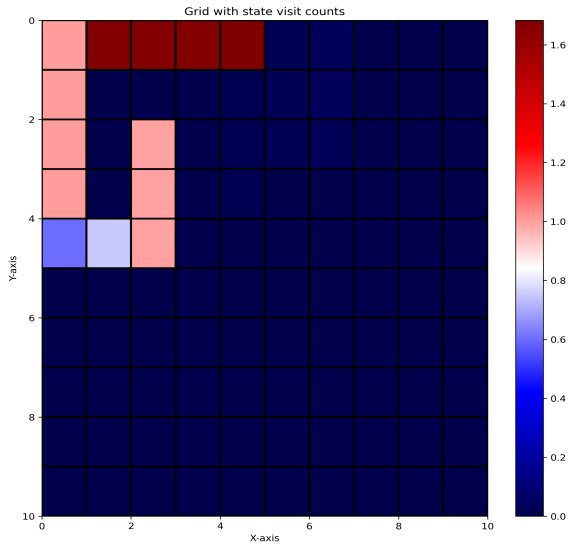
best model $\implies$ mean reward : -8.778, mean steps : 14.701

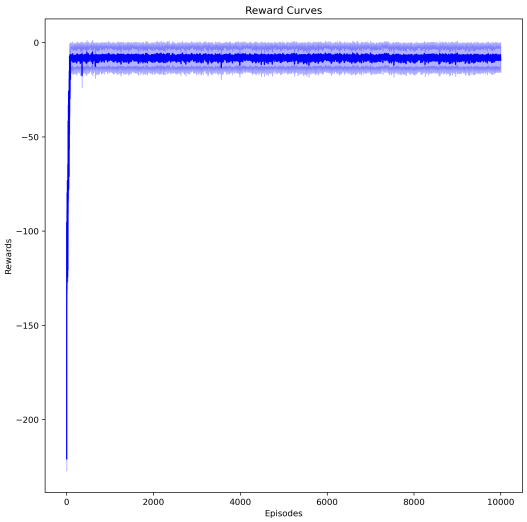
Hyperparameters : $\alpha = 0.64, \gamma = 0.95, \tau = 0.2$& Softmax Policy

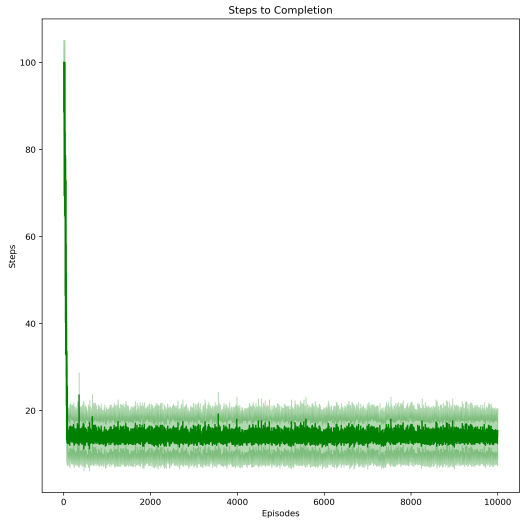
Wind can cause the agent to experience delayed rewards or changes in state due to unpredictable transitions. `SARSA`, being an on-policy algorithm, directly updates its Q-values based on the observed transitions.
Thus in this case, it is more probable to fall to restart sate if it takes other goal paths and so the agent is biased towards goal state `[2 2]`.

2.6 E06 : Initial State : [0 4], Wind : True, p = 1, Algorithm : Q-Learning









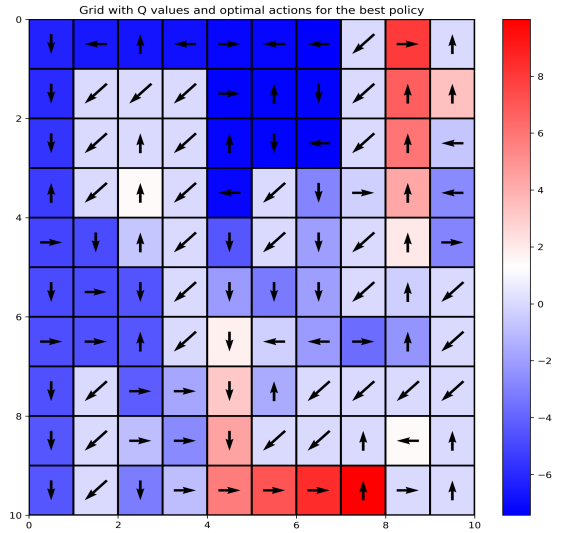
best model $\implies$ mean reward : -8.472, mean steps : 14.392

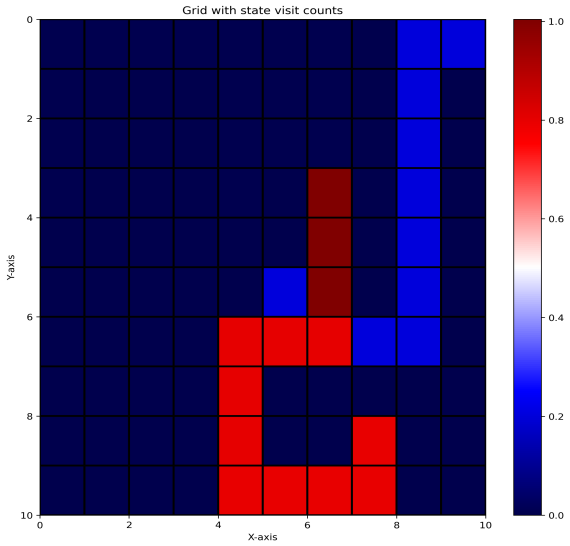
Hyperparameters : $\alpha = 0.73, \gamma = 0.9, \tau = 0.1$& Softmax Policy

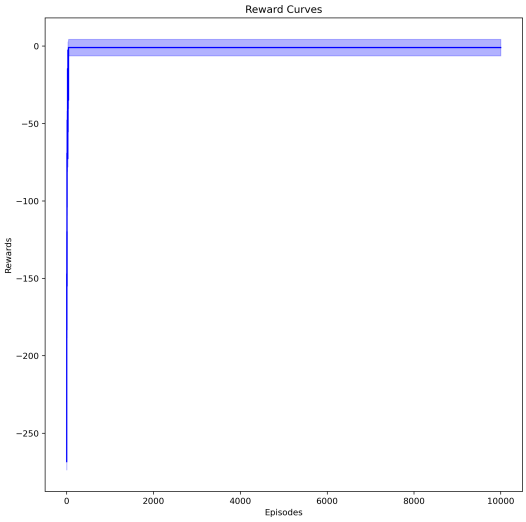
Comparision between E05 and E06 :

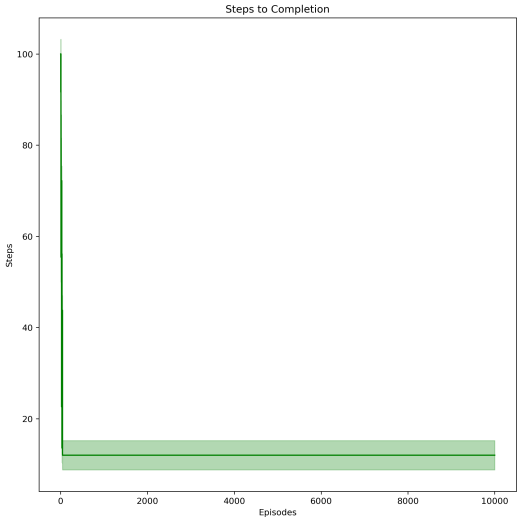
- We can observe that SARSA has lower learning rate than Q-Learning
- The wind in environment has affected the learning dynamics of the model, this has resulted in changes in leanring rate as it induced some stochaticity
- Due to the wind effect, the optimal policy is biased toward a particular goal state [2 2]

2.7 E07 : Initial State : [3 6], Wind : False, p = 1, Algorithm : SARSA







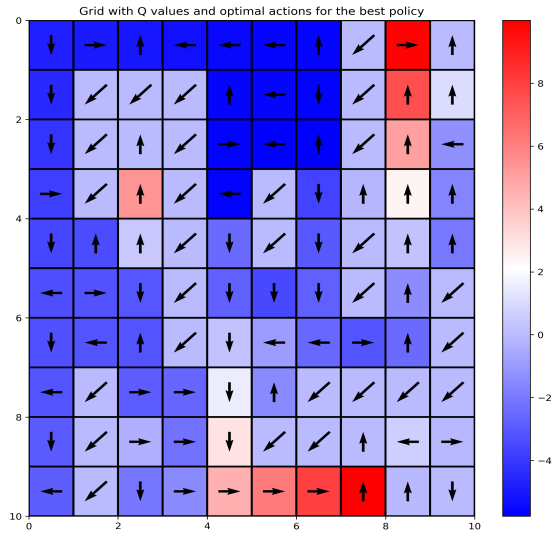


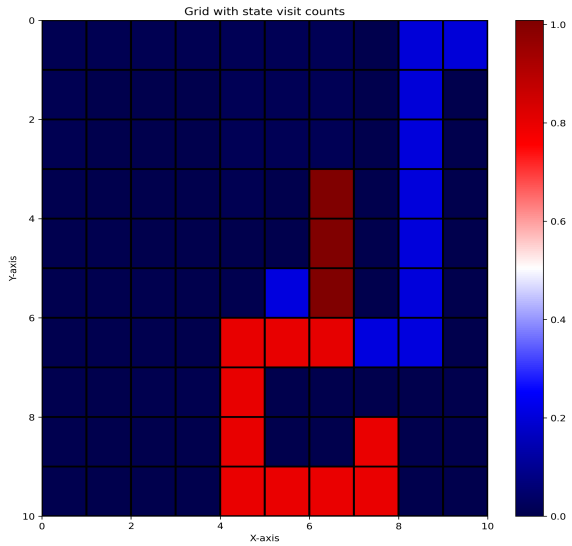
best model $\implies$ mean reward : -1.249, mean steps : 12.173

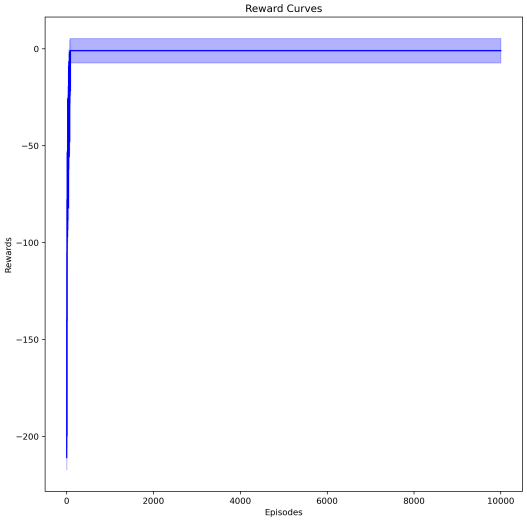
Hyperparameters $\alpha = 0.73, \gamma = 0.95, \tau = 0.1$&Softmax policy end up giving the best model

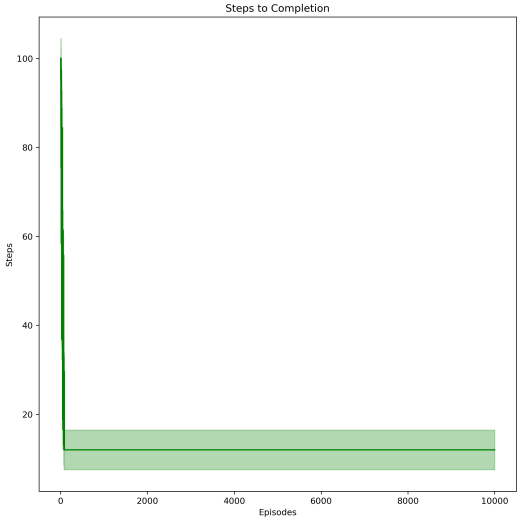
- $\alpha = 0.73$ allows the algorithm to update Q-values more significantly based on new information from each step which leads to faster convergence.
- $\tau = 0.1$ suggest that higher average return will have a higher chance of being chosen in the exploration phase

2.8 E08 : Initial State : [3 6], Wind : False, p = 1, Algorithm : Q-Learning









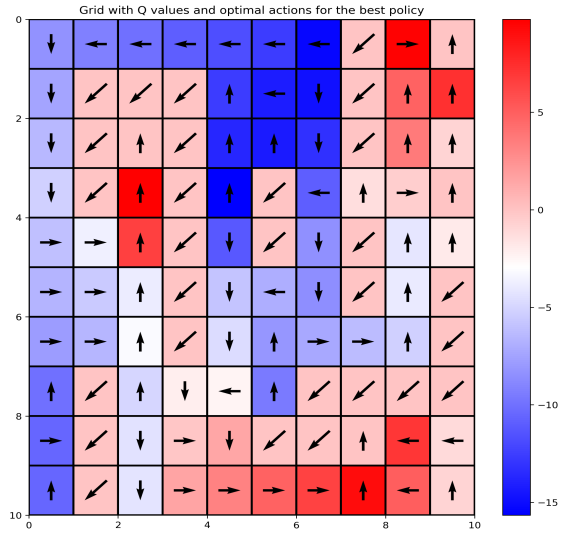
best model $\implies$ mean reward : -1.431, mean steps : 12.336

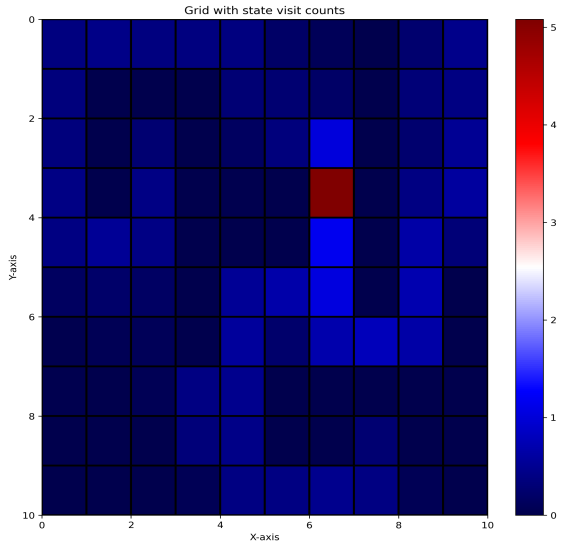
Hyperparameters $\alpha = 0.28, \gamma = 0.9, \tau = 0.1$&Softmax policy end up giving the best model

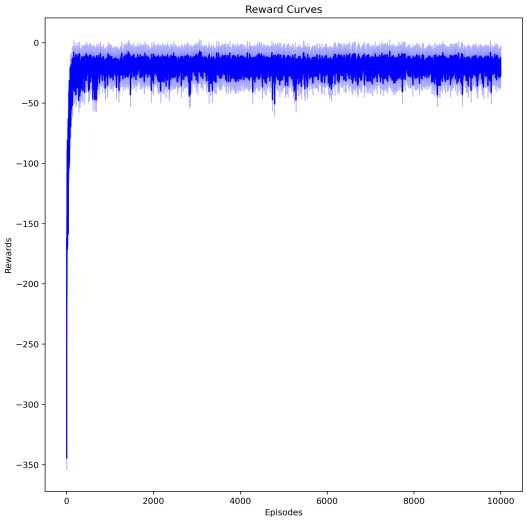
Comparision between E07 and E08

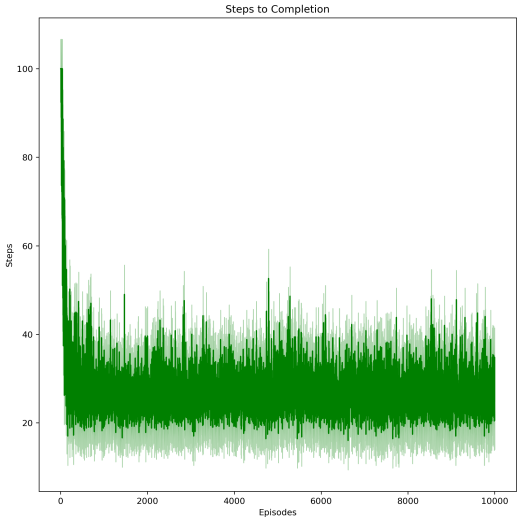
- `SARSA` being on-policy uses higher learning rate than `Q-Learning`, which is an off-policy algorithm, thus the former adapts to its policy quickly while the latter strives to learn the optimal policy.
- In this given start state, the optimal path and the safest path for reaching the end state of `[8 7]` are the same.
- Due to the difference in the start states as compared to E01 and E02, other optimal paths could be calculated to different goal states. One such state here is `[0 9]` but due to the proximity of bad states and restart states on the path to reach this goal state, this path is rarely taken.

2.9 E09 : Initial State : [3 6], Wind : False, p = 0.7, Algorithm : SARSA







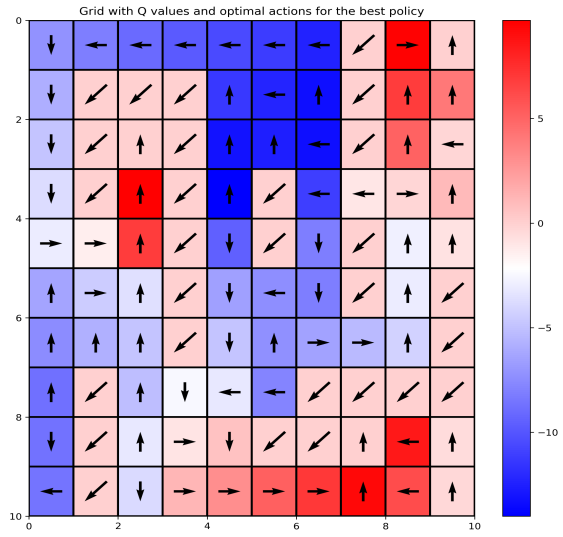


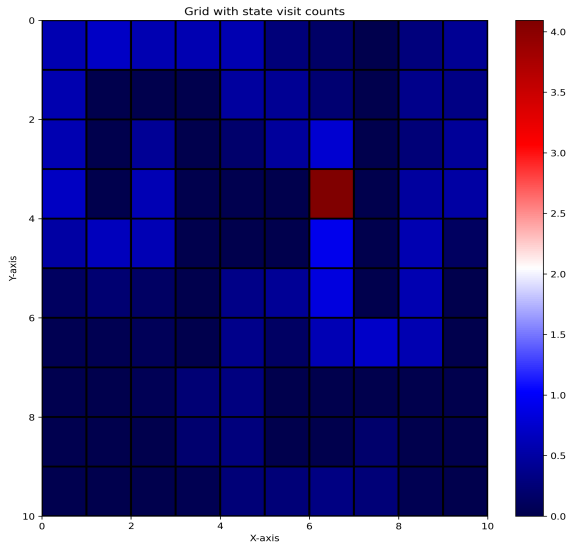
best model $\implies$ mean reward : -20.406, mean steps : 27.182

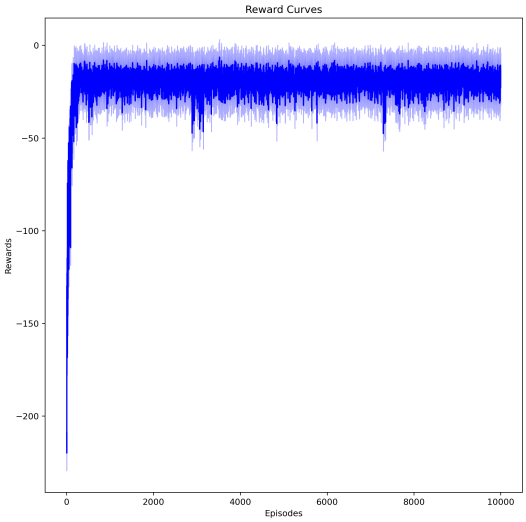
Hyperparameters $\alpha = 0.37, \gamma = 0.95, \tau = 0.2$&Softmax policy end up giving the best model

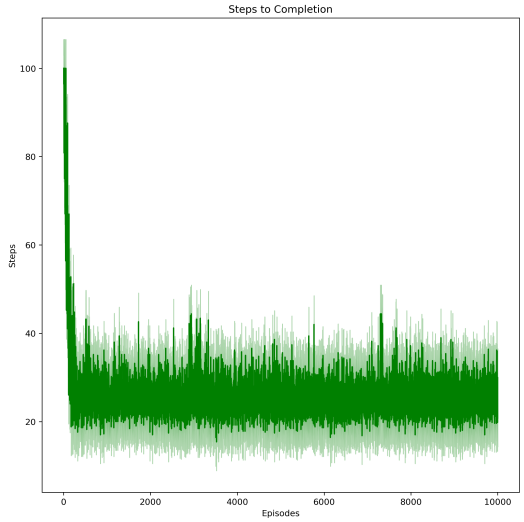
Due to the small learning rate, multiple optimal paths exist in the environment

2.10 E10 : Initial State : [3 6], Wind : False, p = 0.7, Algorithm : Q-Learning









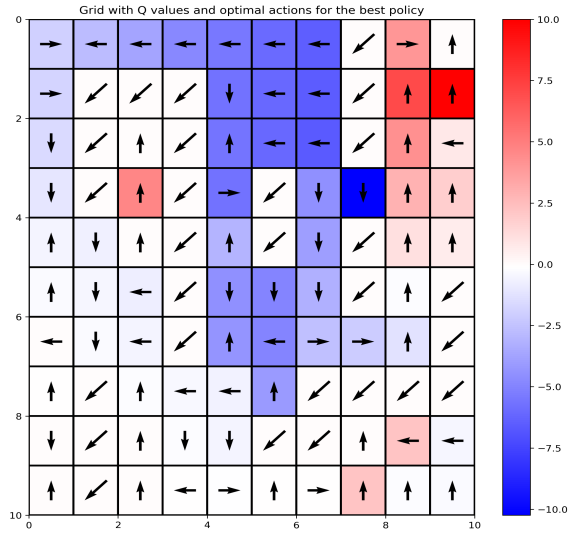
best model $\implies$ mean reward : -19.935, mean steps : 26.011

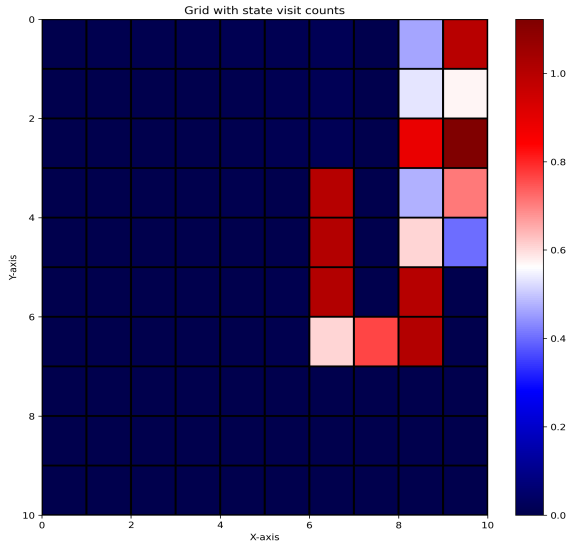
Hyperparameters $\alpha = 0.19, \gamma = 0.95, \tau = 0.1$&Softmax policy end up giving the best model

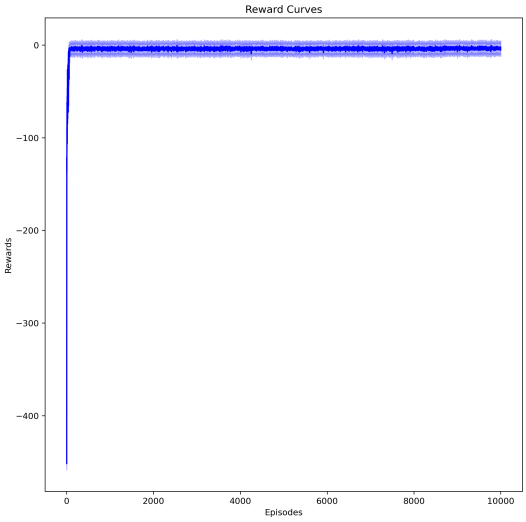
Comparision between E09 and E10 :

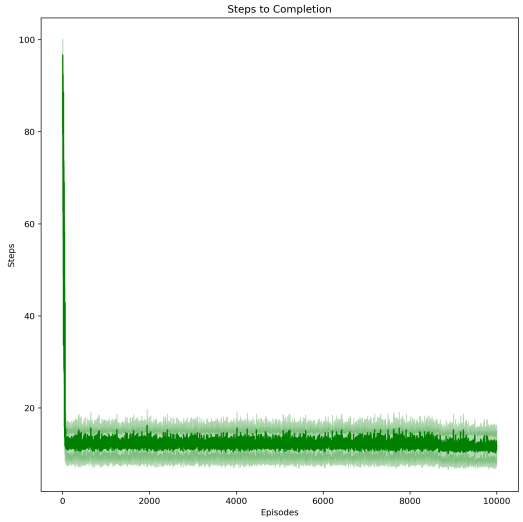
- The learning rate for SARSA is smaller than E07 due to the stochastic nature of the actions while E07 encountered deterministic actions.
- SARSA cautiously updates its estimates due to the stochastic nature of the environment whereas Q-Learning strives to find optimal policy [off-policy algorithm].
- Due to the difference in the start states, multiple optimal policies have been computed as result of lower learning rate.
- The selection policy and discount factor (γ) are similar for both cases

2.11 E11 : Initial State : [3 6], Wind : True, p = 1, Algorithm : SARSA







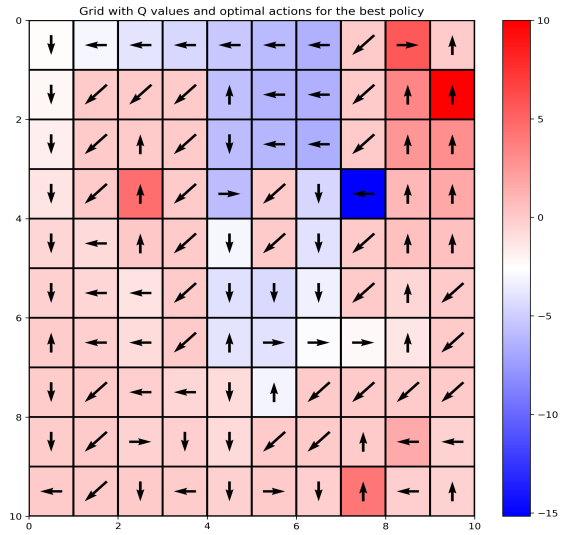


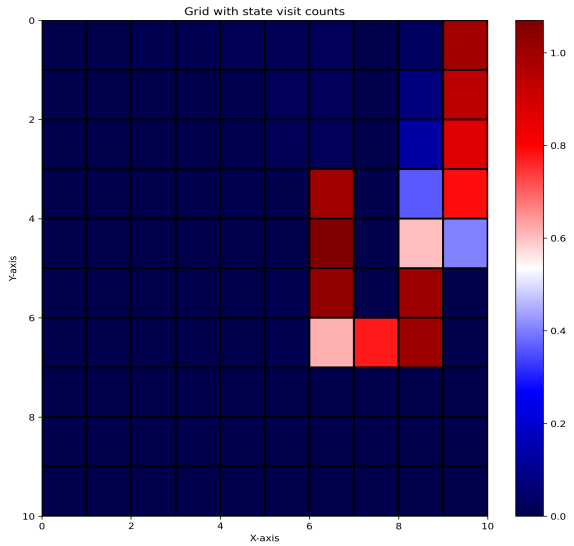
best model $\implies$ mean reward : -4.324, mean steps : 12.344

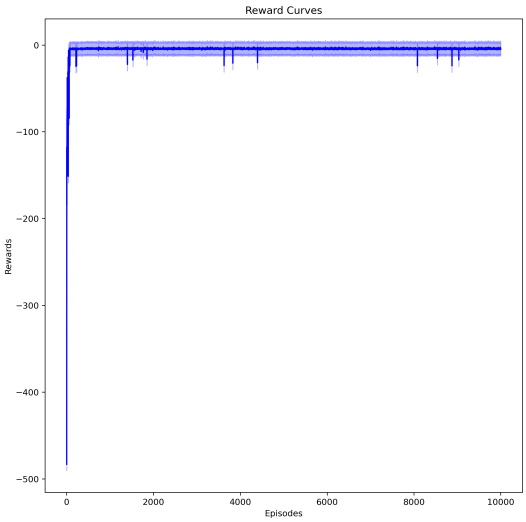
Hyperparameters : $\alpha = 0.37, \gamma = 0.9, \tau = 0.1$& Softmax Policy

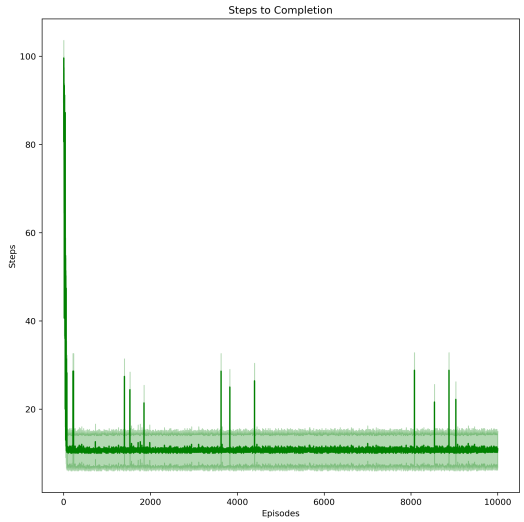
Due to the presence of wind in the environment, `SARSA` chooses a slower learning rate (α) just to be cautious

2.12 E12 : Initial State : [3 6], Wind : True, p = 1, Algorithm : Q-Learning









best model $\implies$ mean reward : -4.847, mean steps : 10.968

Hyperparameters : $\alpha = 0.19, \gamma = 0.9, \tau = 0.2$& Softmax Policy **Comparision between E11 and E12 :**

- The wind in environment has affected the learning dynamics of the model, this has resulted in changes in leanring rate as it induced some stochaticity.
- Due to the wind effect, the optimal policy is biased toward a particular goal state [0 9] as the wind keeps blowing towards the the right of grid world.