Report

CS6700 Spring 2024

Assignment 1
Prof Balaram Ravindran

Submitted by:
Akranth Reddy me20b100@smail.iitm.ac.in
Hemanth me20b045@smail.iitm.ac.in

Date: 28-02-2024

The report is organized as follows
TheExperiment result section has 12 subsections for each section (3 grid world variants x 2 start stages x Learning Algorithms)

Note:

1. The hyperparameters reported here are the best for the given experiments.
2. Each experiment ran for 10,000 Episodes.
3. b = 0.5 is used for all the experiments

Common Observations

- More or less the main idea of a policy is to explore in the beginning and then exploit the experience it learns, the softmax policy tends to do this and we have a control of how deterministic and how much should it explore, using a low temperature value will allow more exploration and a higher temperature will lead to more exploitations. Using a plain epsilon greedy policy the agent learnt slower than softmax policy as observed because we have a fixed probability of exploring new states, even when the agent has gained good experience it still explores this leads to lower rewards than softmax, but we can improvise here by changing the epsilon value dynamically like after half the number of episodes reduce the eposion value by some amount and at the end we have a policy that goes with the best Q value for the action, state.
- Q-Learning converges faster and easier to finetune the hyperparameters than the SARSA Learning rule.
- Lower learning rate is better for stability, since choosing a **higher learning rate leads to dependency on new information** and if the agent explores then Q-value changes a lot which is bad, since all the collected old information is lost.
- The problem we had was to teach the agent to go from the starting state to the goal state, and we incur a large positive reward when we reach the goal so we have a lot of

dependency in the future for reward so a higher gamma = 1.0 (discount factor) is chosen. This is also confirmed by experimentation with different gamma values.

Q-Learning vs SARSA

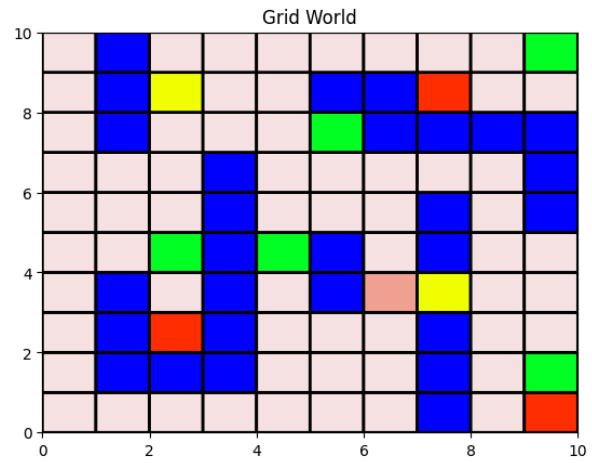
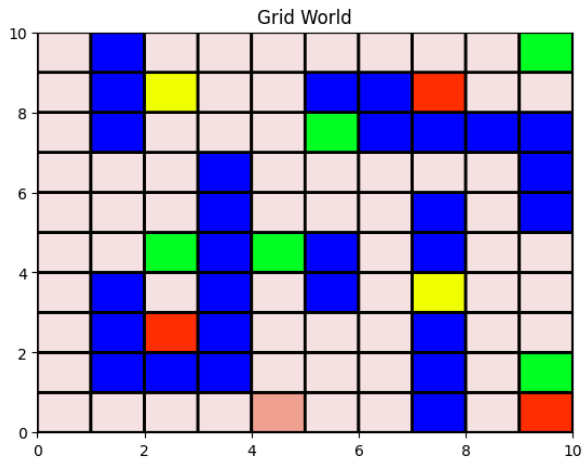
- We had observed that in the case of agents stochastically shifting to east or west (P!=1) or in the case of wind, these situations enabled the agent to explore more, so using a lower epsilon value or lower tau value would be sufficient.
- Q-Learning updates the value function by choosing the maximum value at the next state irrespective of action taken whereas SARSA updates its value function with the value of the state we reach by following the action. Because of this Q-learning reduces the randomness in which an agent learns and hence it converges faster. But the trade off is we may not find the optimal path.
- Consider the Experiments for scenario - 1, deterministic state (P=0, no wind) Experiment -1 Q-Learning was able to find the best paths faster than SARSA, where as if you consider the Scenario -2 (P=1) SARSA is better as it explores more and finds the optimal path, Experiment - 7 (SARSA) has reward -19, for the similar setting Experiment - 8 (Q-Learning) has -42 reward.

Takeaways

- SARSA is better for stochastic environments than Q-Learning.
- Use a higher discount factor if rewards are far away in future.
- Lower learning rate is better for stability.
- SARSA inherently does more exploration than Q-Learning.

Environment

Green - Bad states
Yellow - Restart states
Red - Goal states
Blue - Obstructed states
Peach - Start state

Start State = [3, 6]	Start State = [0, 4]
	

Experiment Results

Objective was to make the agent learn to go from start to goal state such that it incurs less reward. Experiments were conducted to understand the learning algorithm, policy used to see what makes the agent learn faster.

Experiment 1

Setting:

Start state: [3,6]
Learning Algorithm: SARSA
Wind = False
p = 1.0 (deterministic step)

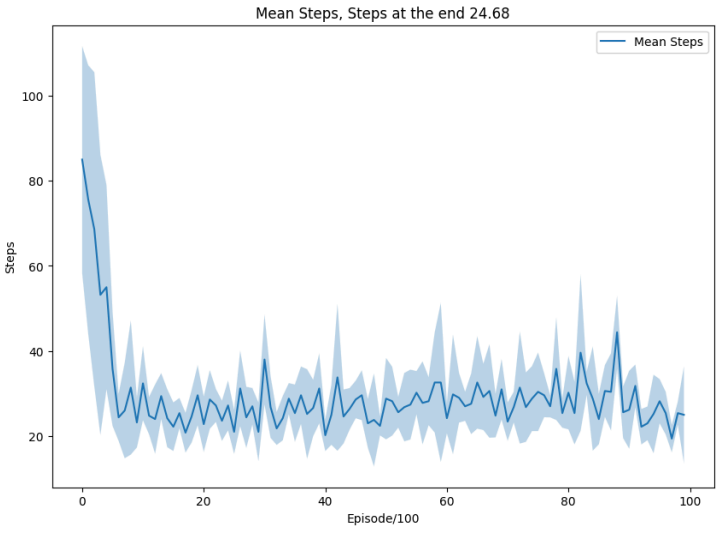
Hyperparametes:

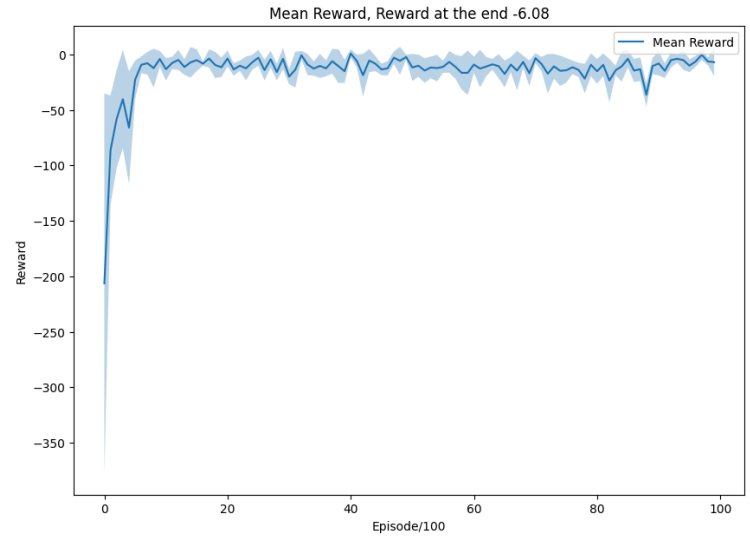
Alpha = 0.1
Gamma = 1.0
Policy = Softmax
Tau = 1.0

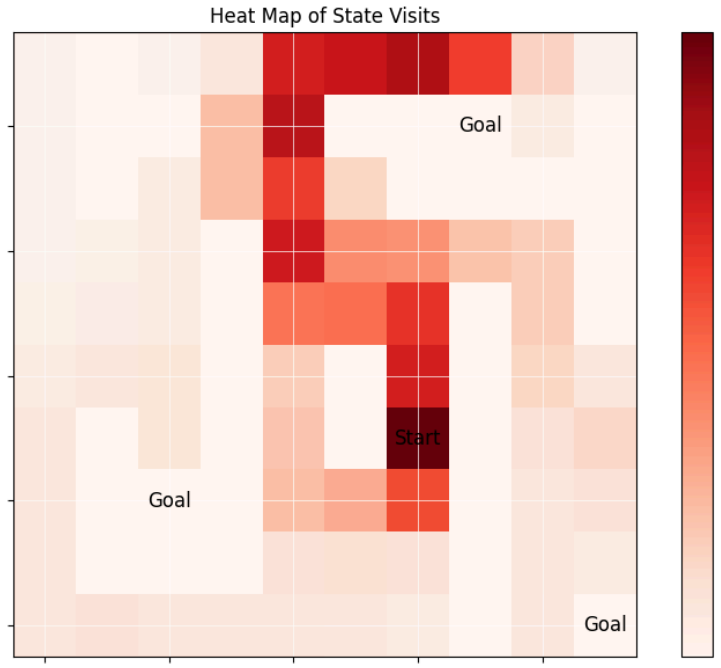
Design Choices:

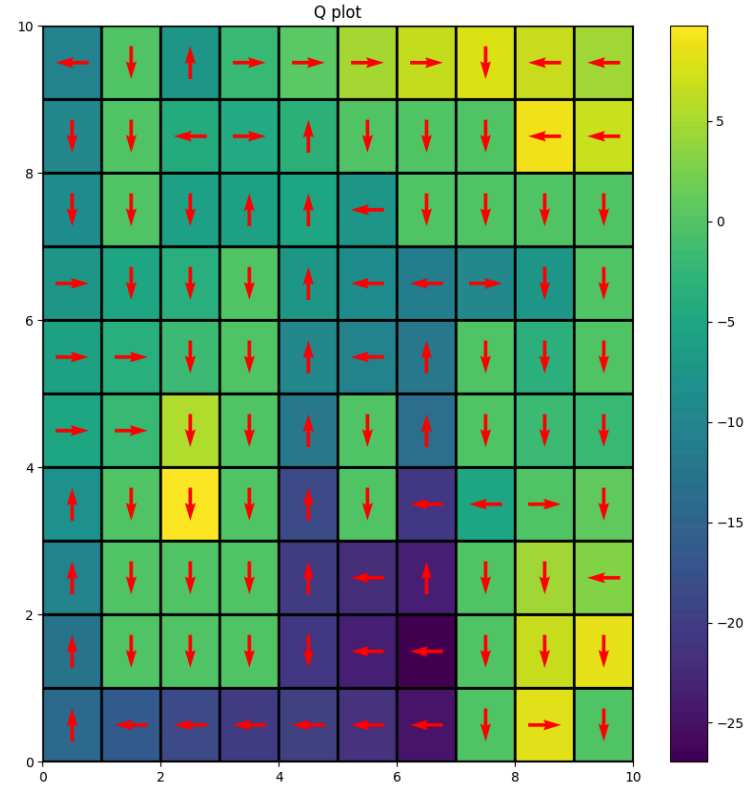
- A higher discount factor (gamma) is used to account for future rewards, learning rate (alpha) is set to be 0.1 for stability.
- Softmax policy was found to be better than epsilon greedy policy.

Plots:


Mean Steps, Steps at the end 24.68
Mean Steps
Steps
Episode/100


Mean Reward, Reward at the end -6.08
Mean Reward
Reward
Episode/100


Heat Map of State Visits
Goal
Start
Goal
Goal


Q plot

Experiment 2

Setting:

Start state: [3,6]
Learning Algorithm: Q-Learning
Wind = False
p = 1.0 (deterministic step)

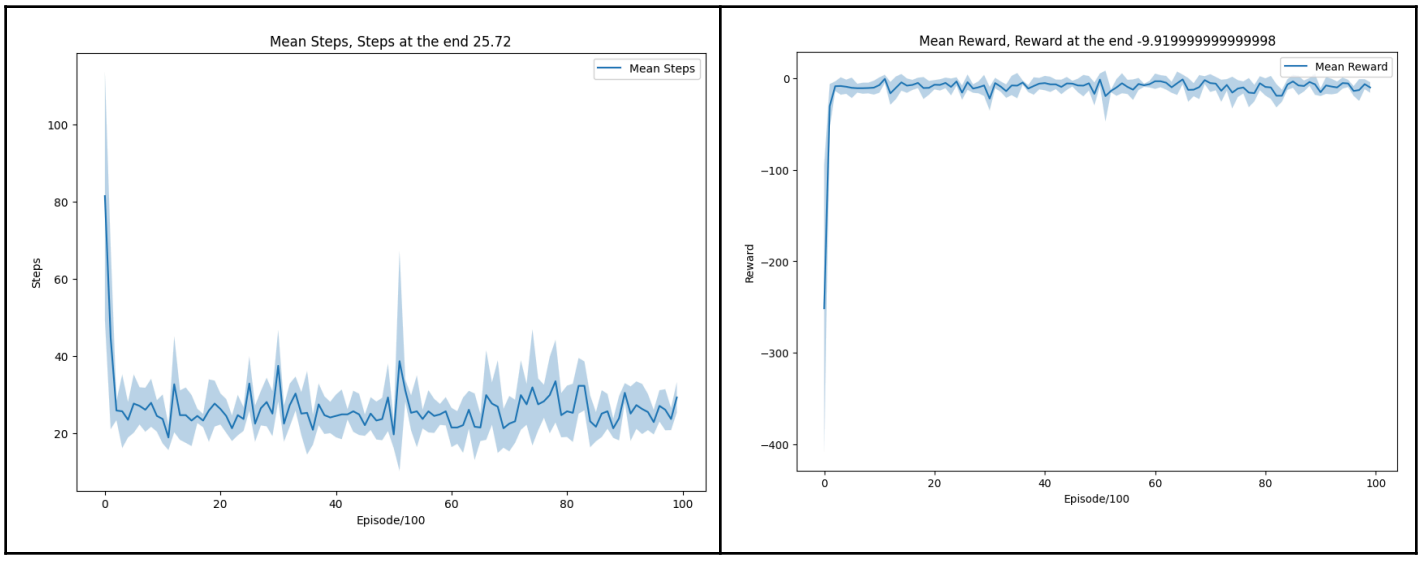
Hyperparametes:

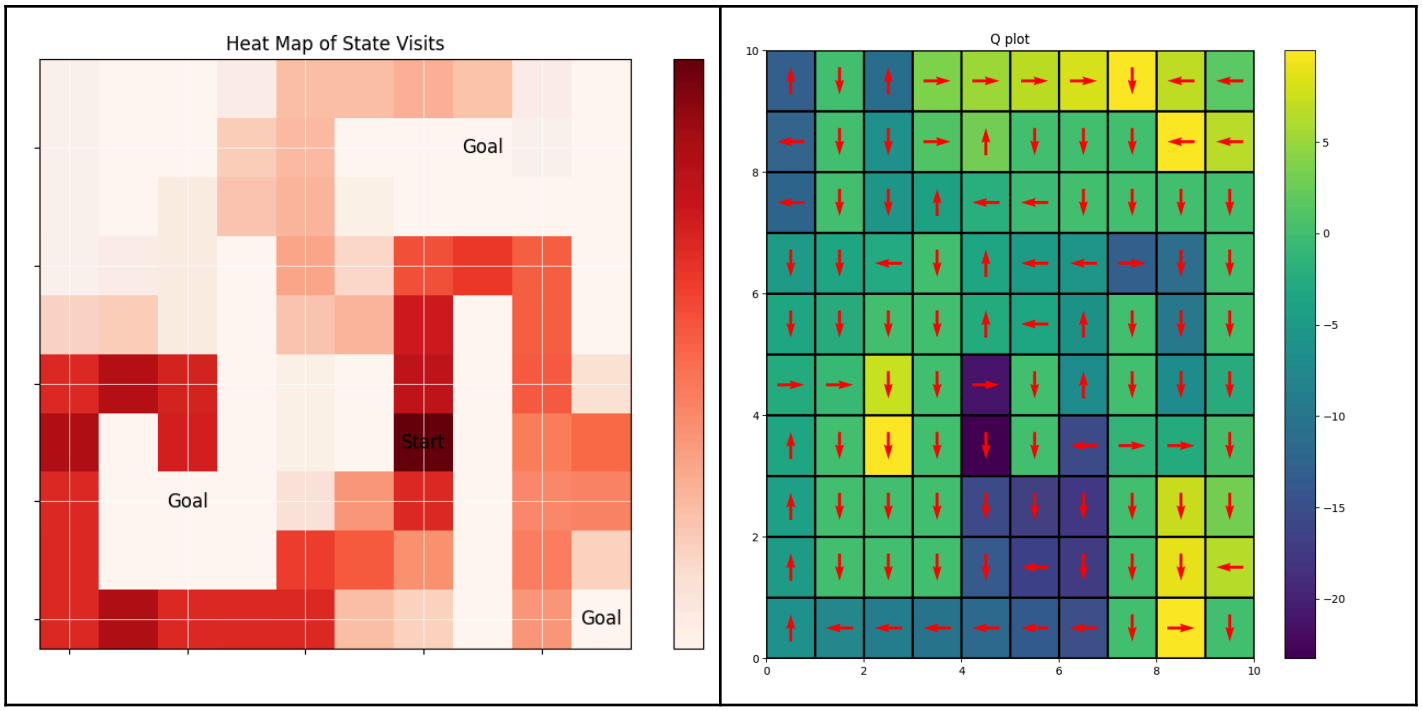
Alpha = 0.5
Gamma = 1.0
Policy = Softmax
Tau = 1.0

Design Choices:

- A higher tau value is chosen for this case to enable the agent to explore more since Q-Learning has less exploration in it.
- Higher learning rate (alpha) is used because Q-learning converges faster and we don't have random exploring in the end so Q-values do not change and we achieve stability.

Plots:





Experiment 3

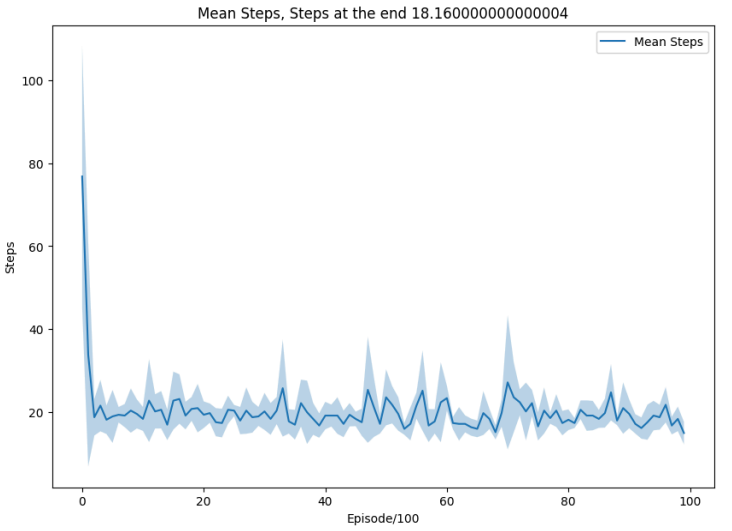
Setting:

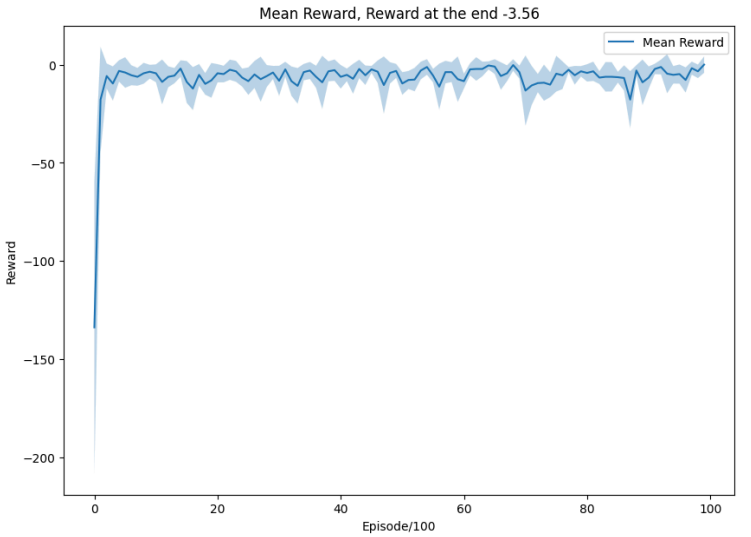
- Start state: [0,4]
- Learning Algorithm: SARSA
- Wind = False
- p = 1.0 (deterministic step)

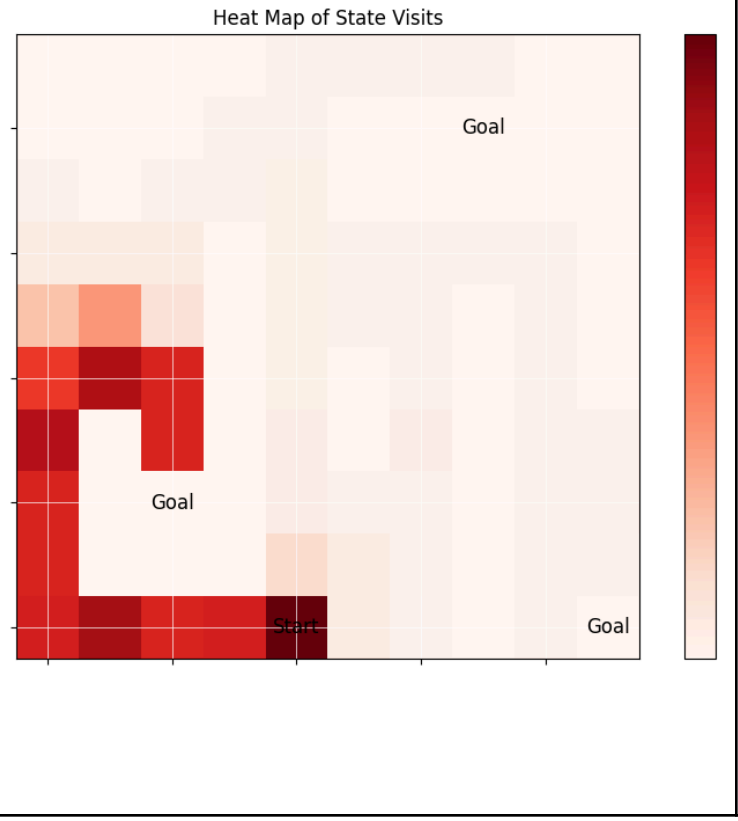
Hyperparametes:

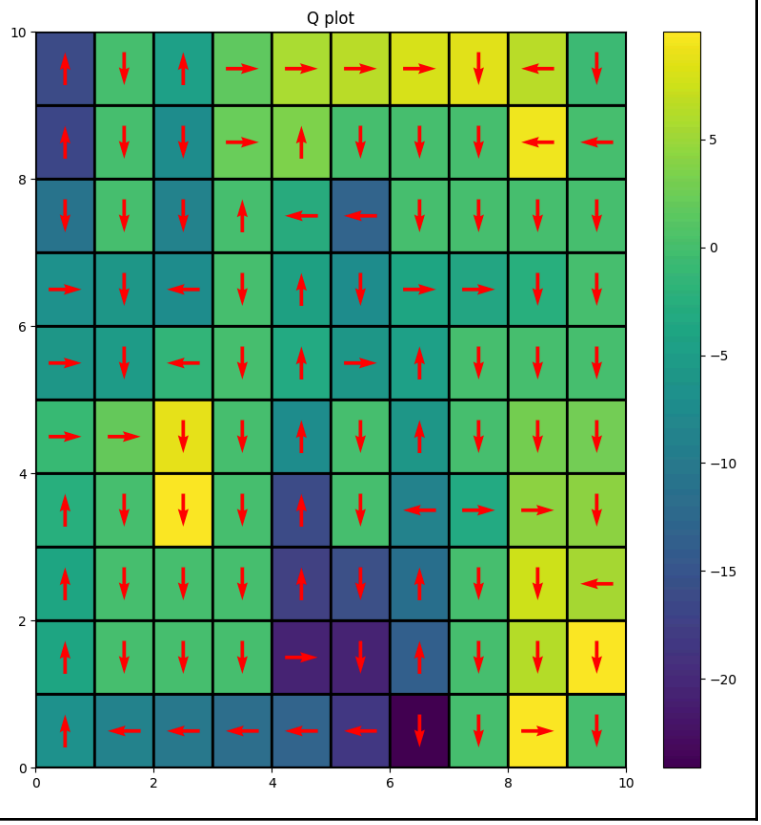
- Alpha = 0.5
- Gamma = 1.0
- Policy = Softmax
- Tau = 0.5

Plots:


Mean Steps, Steps at the end 18.160000000000004
Mean Steps
Steps
Episode/100


Mean Reward, Reward at the end -3.56
Mean Reward
Reward
Episode/100


Heat Map of State Visits
Goal
Goal
Start
Goal


Q plot

Experiment 4

Setting:

Start state: [0,4]
Learning Algorithm: Q-Learning
Wind = False
p = 1.0 (deterministic step)

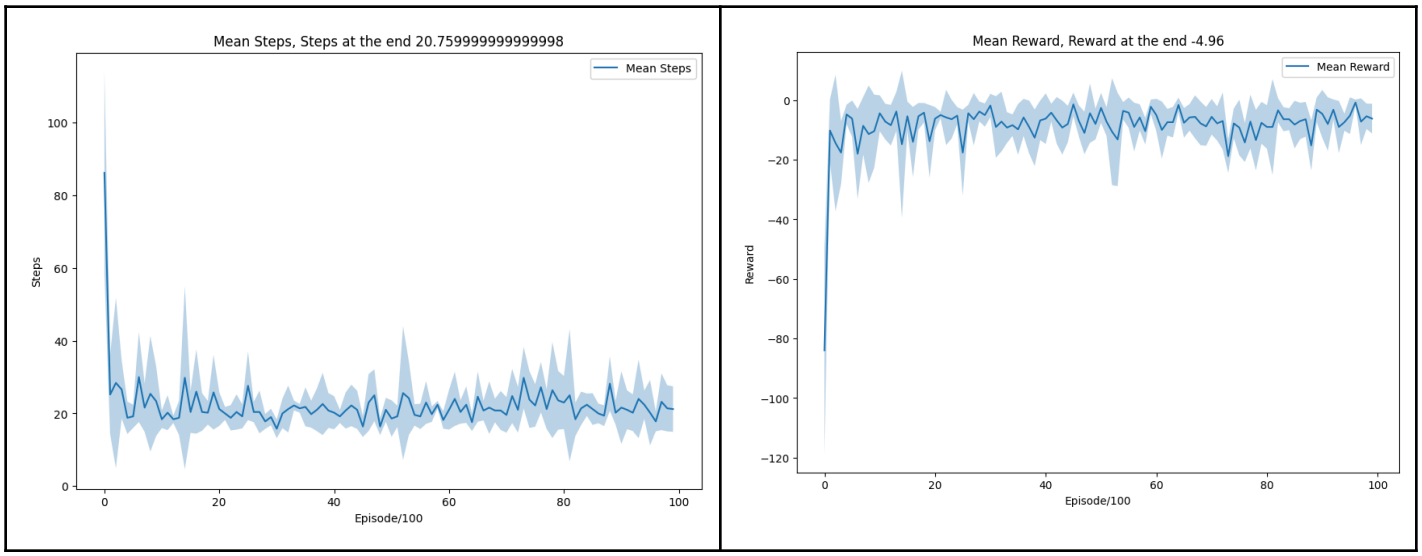
Hyperparametes:

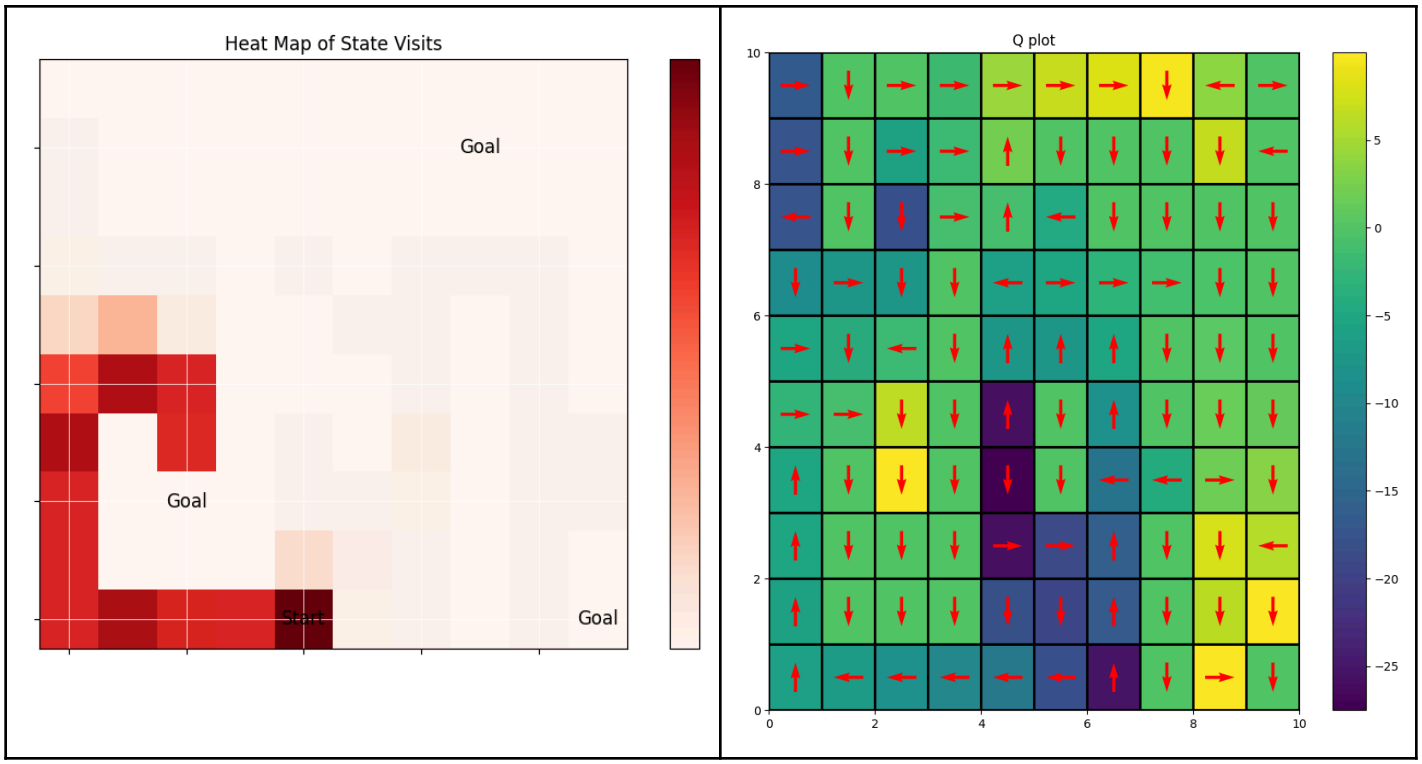
Alpha = 0.5
Gamma = 1.0
Policy = epsilon_policy (with decay of 0.999)
starting epsilon = 0.5

Design Choices:

Since the optimal path is near the starting point there, exploration is not needed much so a lower lau value is chosen. Epsilon greedy policy is chosen with epsilon value starting at 0.5 and reduced by 0.999. Since not much exploration is needed (optimal path is near) so a lower epsilon value is chosen.

Plots:





Experiment 5

Setting:

Start state: [3,6]
Learning Algorithm: SARSA
Wind = False
p = 0.7 (Stochastic step)

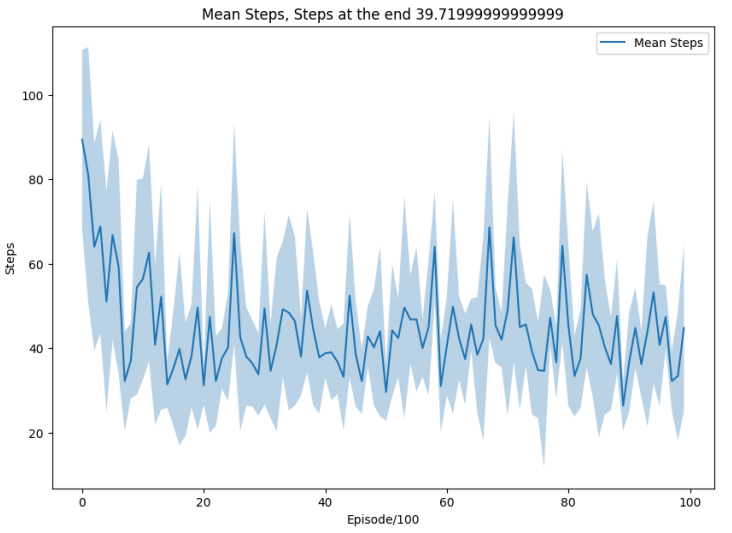
Hyperparametes:

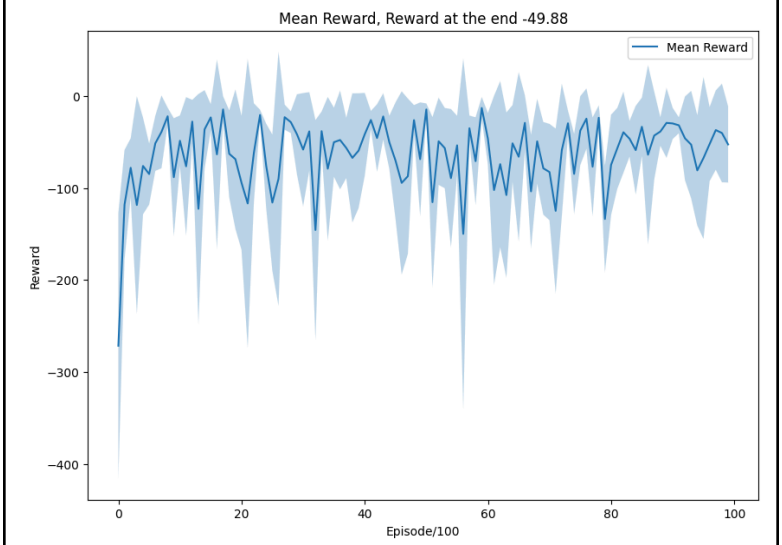
Alpha = 0.1
Gamma = 1.0
Policy = Softmax
Tau = 0.5

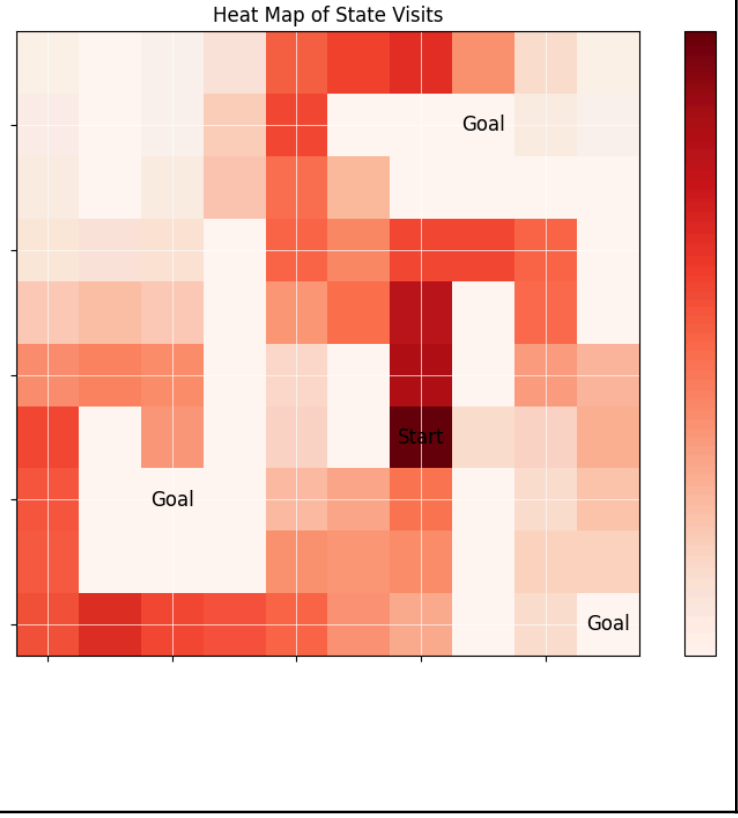
Design Choices:

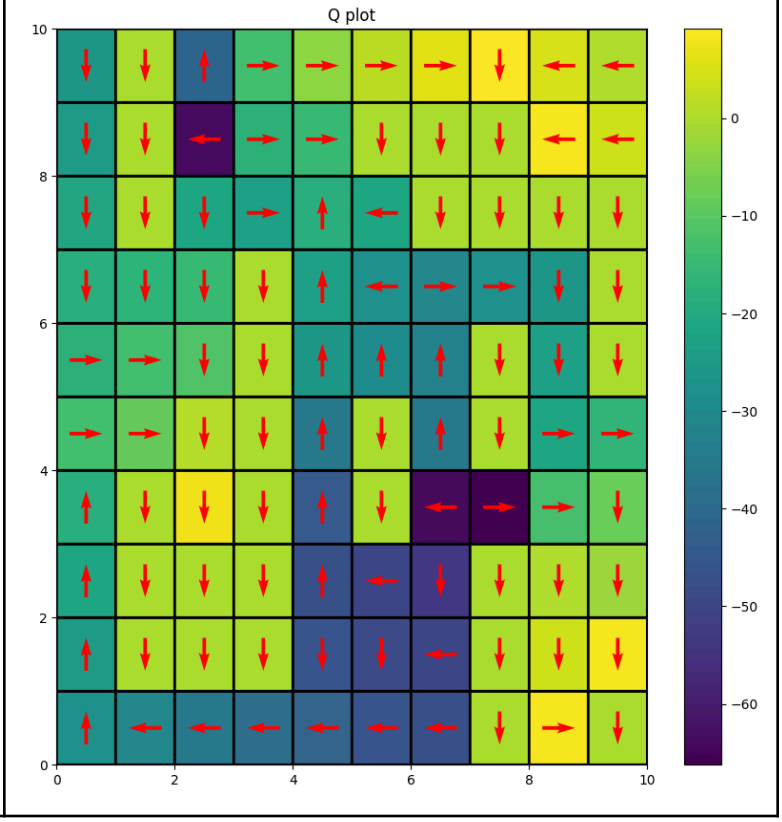
- Because stochasticity comes from randomly jumping to other states, epsilon greedy policy is not used, rather softmax is used which eliminates the randomness in actions taken.
- Because of high stochasticity (exploration) a lower learning rate would make the value function stable.
- Similarly a lower tau value is used.
- Because of a lot of randomness, the variance of the reward distribution is high.

Plots:


Mean Steps, Steps at the end 39.71999999999999
Mean Steps
Steps
Episode/100


Mean Reward, Reward at the end -49.88
Mean Reward
Reward
Episode/100


Heat Map of State Visits
Goal
Start
Goal
Goal


Q plot

Experiment 6

Setting:

Start state: [3,6]
Learning Algorithm: Q-Learning
Wind = False
p = 0.7 (Stochastic step)

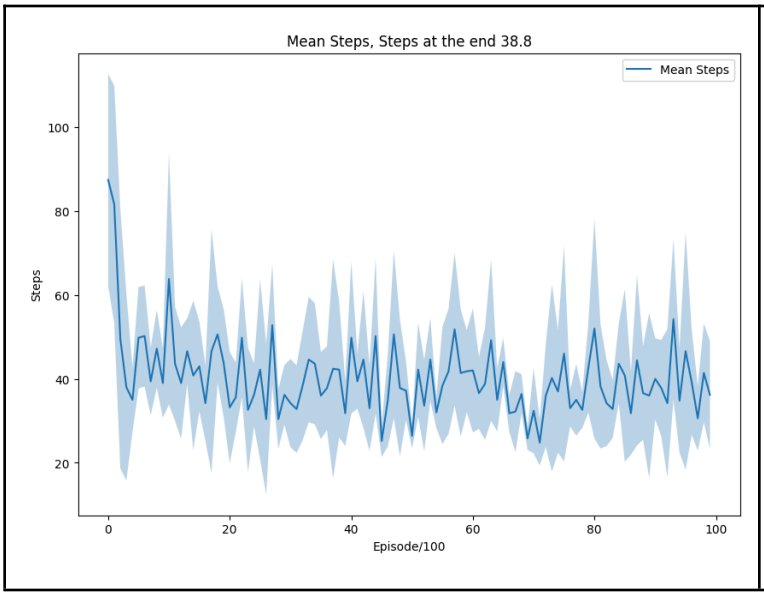
Hyperparametes:

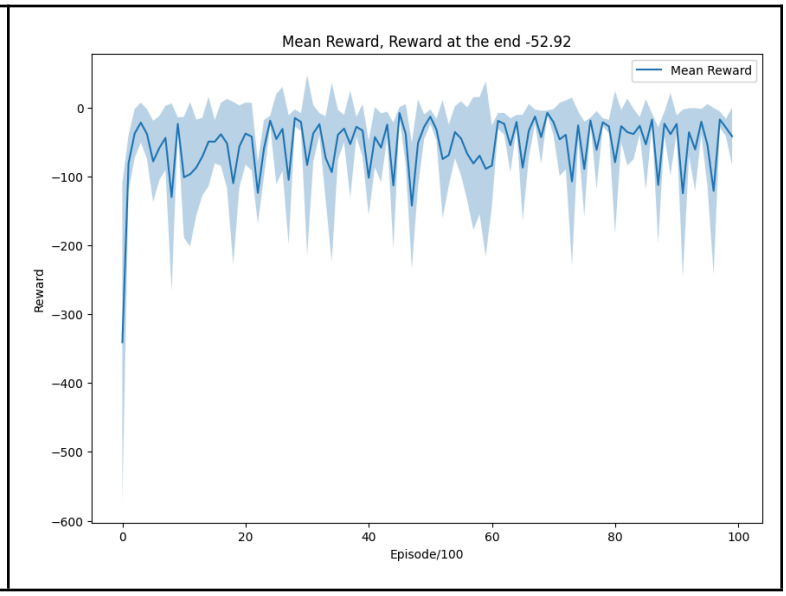
Alpha = 0.1
Gamma = 1.0
Policy = Softmax
Tau = 0.5

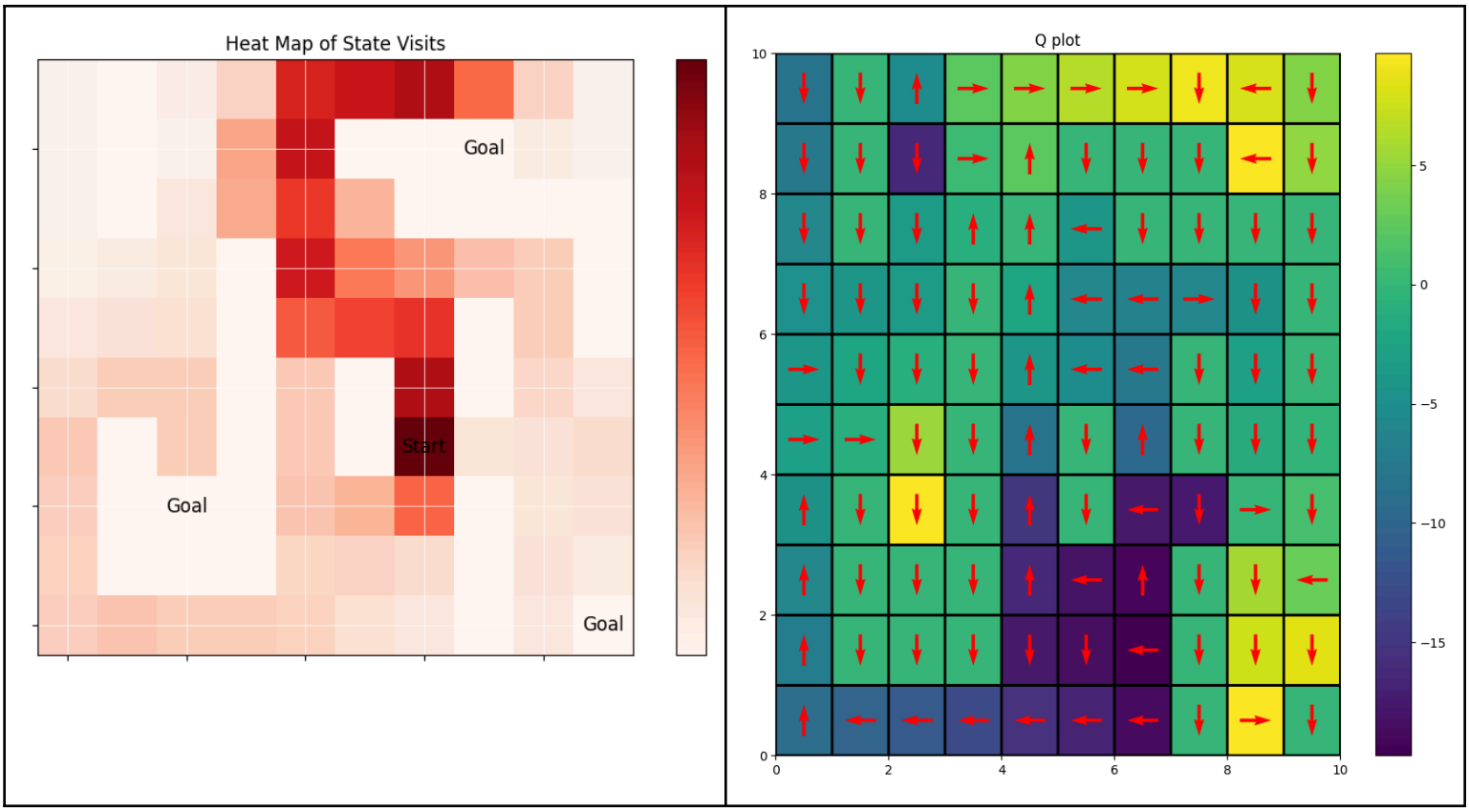
Design Choices:

- Because of high stochasticity (exploration) a lower learning rate would make the value function stable.
- As compared to the above experiment SARSA is better in stochastic environments as it explores more to find the optimal path.
- Because of a lot of randomness, the variance of the reward distribution is high.

Plots:







Experiment 7

Setting:

Start state: [0,4]
Learning Algorithm: SARSA
Wind = False
p = 0.7 (Stochastic step)

Hyperparametes:

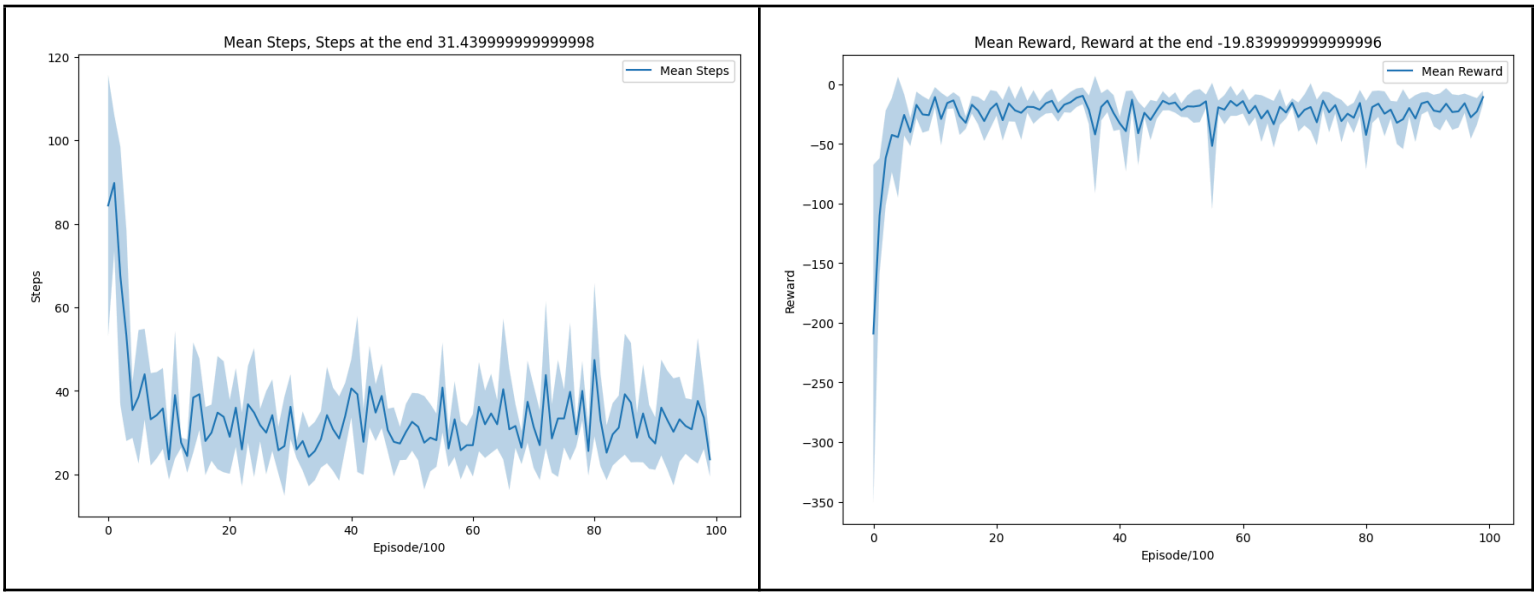
Alpha = 0.1
Gamma = 1.0
Policy = Softmax
Tau = 0.5

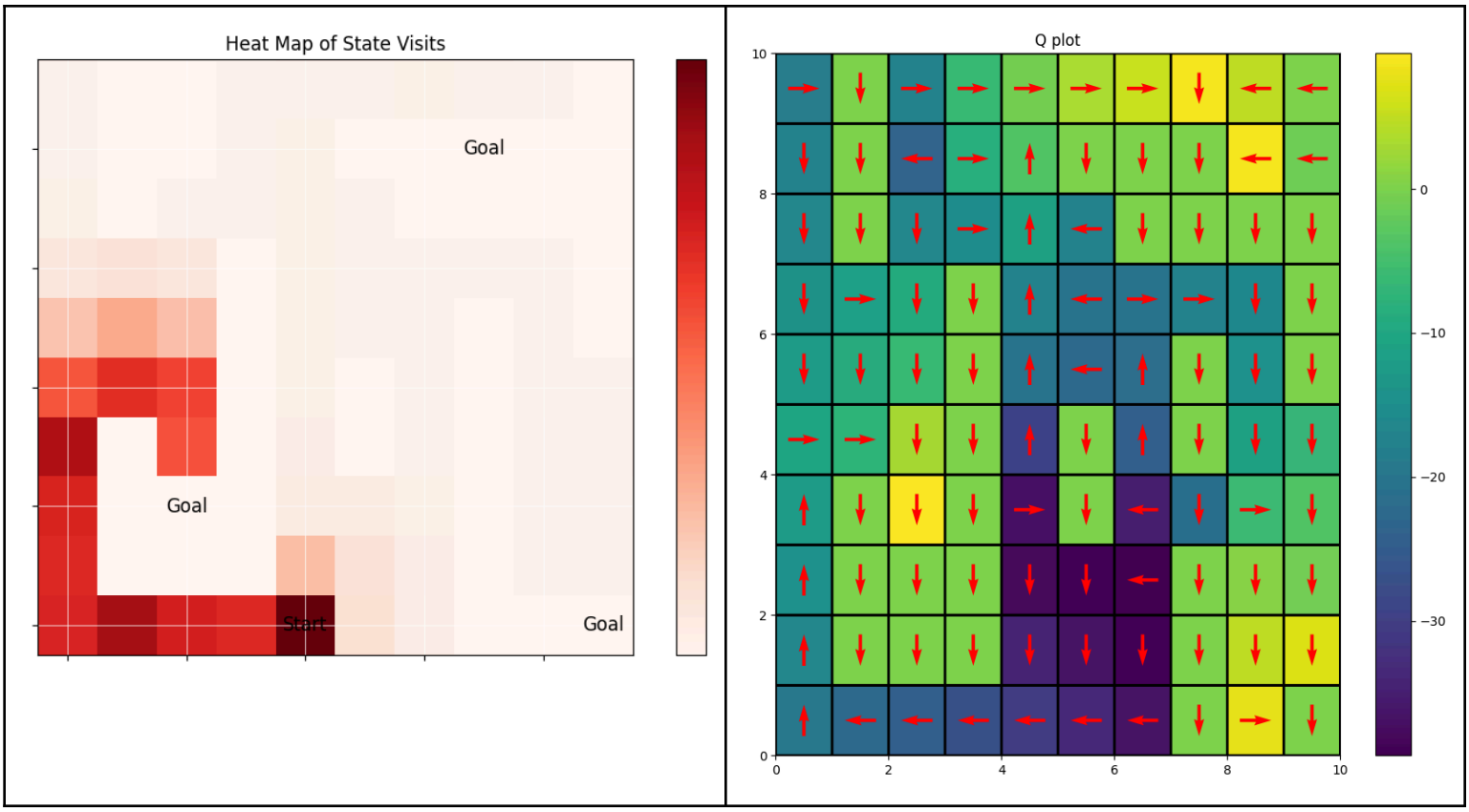
Design Choices:

- A higher discount factor (gamma) is used to account for future rewards, learning rate (alpha) is set to be 0.1.

- Because stochasticity comes from randomly jumping to other states, epsilon greedy policy is not used, rather softmax is used which eliminates the randomness in actions taken.

Plots:





Experiment 8

Setting:

Start state: [0,4]
Learning Algorithm: Q-Learning
Wind = False
p = 0.7 (Stochastic step)

Hyperparametes:

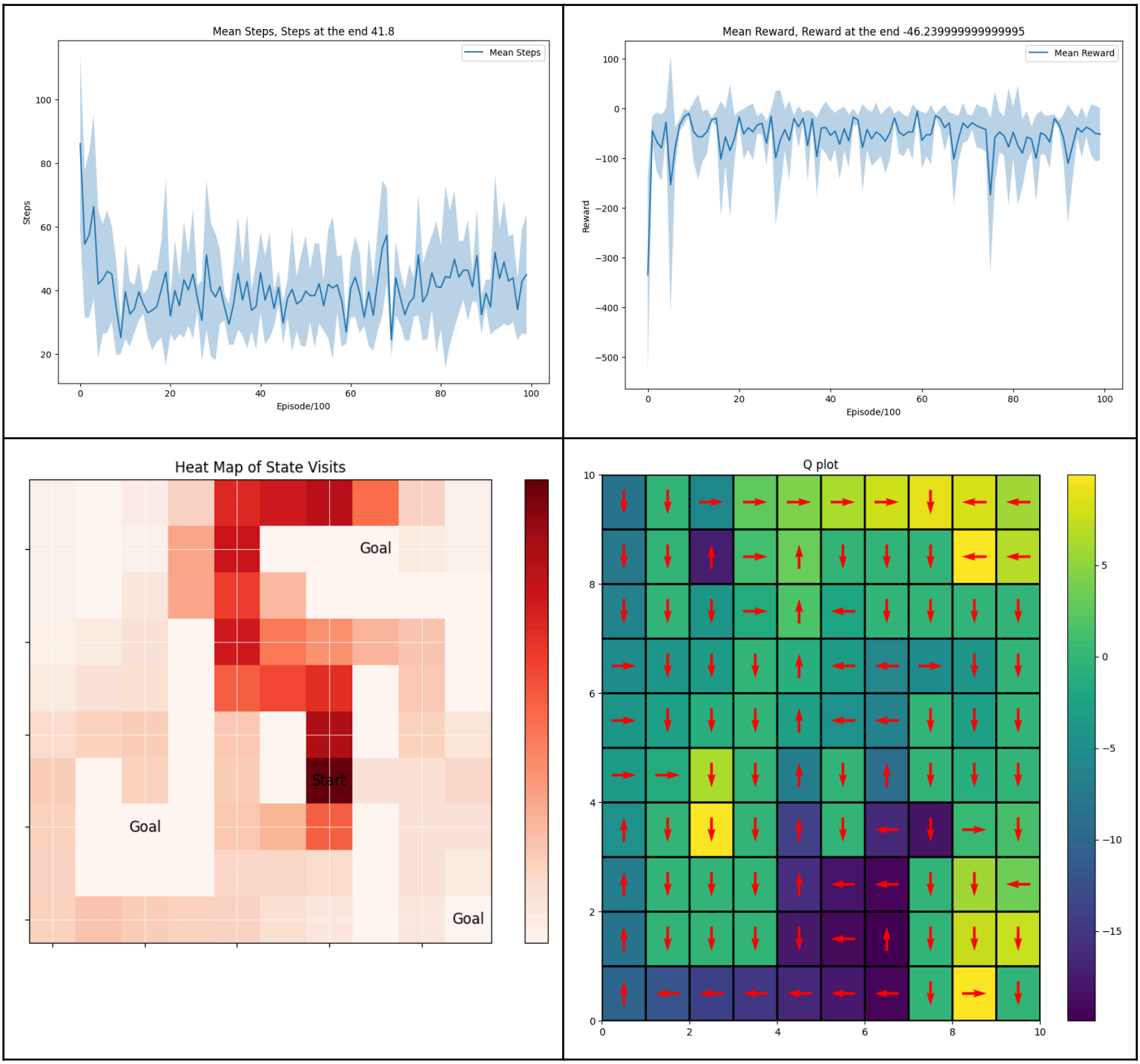
Alpha = 0.1
Gamma = 1.0
Policy = Softmax
Tau = 0.5

Design Choices:

- Since the environment is stochastic SARSA is better.

- Because stochasticity comes from randomly jumping to other states, epsilon greedy policy is not used, rather softmax is used which eliminates the randomness in actions taken.

Plots:



Experiment 9

Setting:

Start state: [0,4]
Learning Algorithm: Q-Learning
Wind = False
p = 0.7 (Stochastic step)

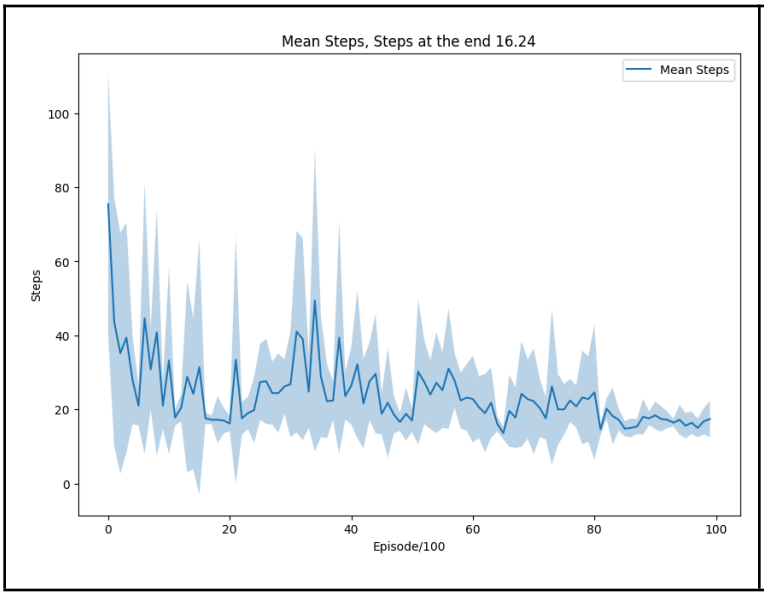
Hyperparametes:

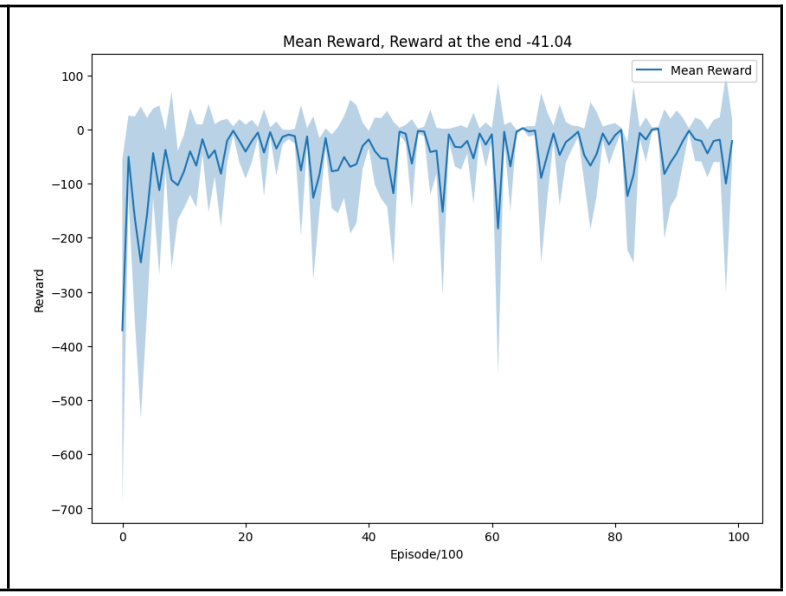
Alpha = 0.1
Gamma = 1.0
Policy = Softmax
Tau = 0.5

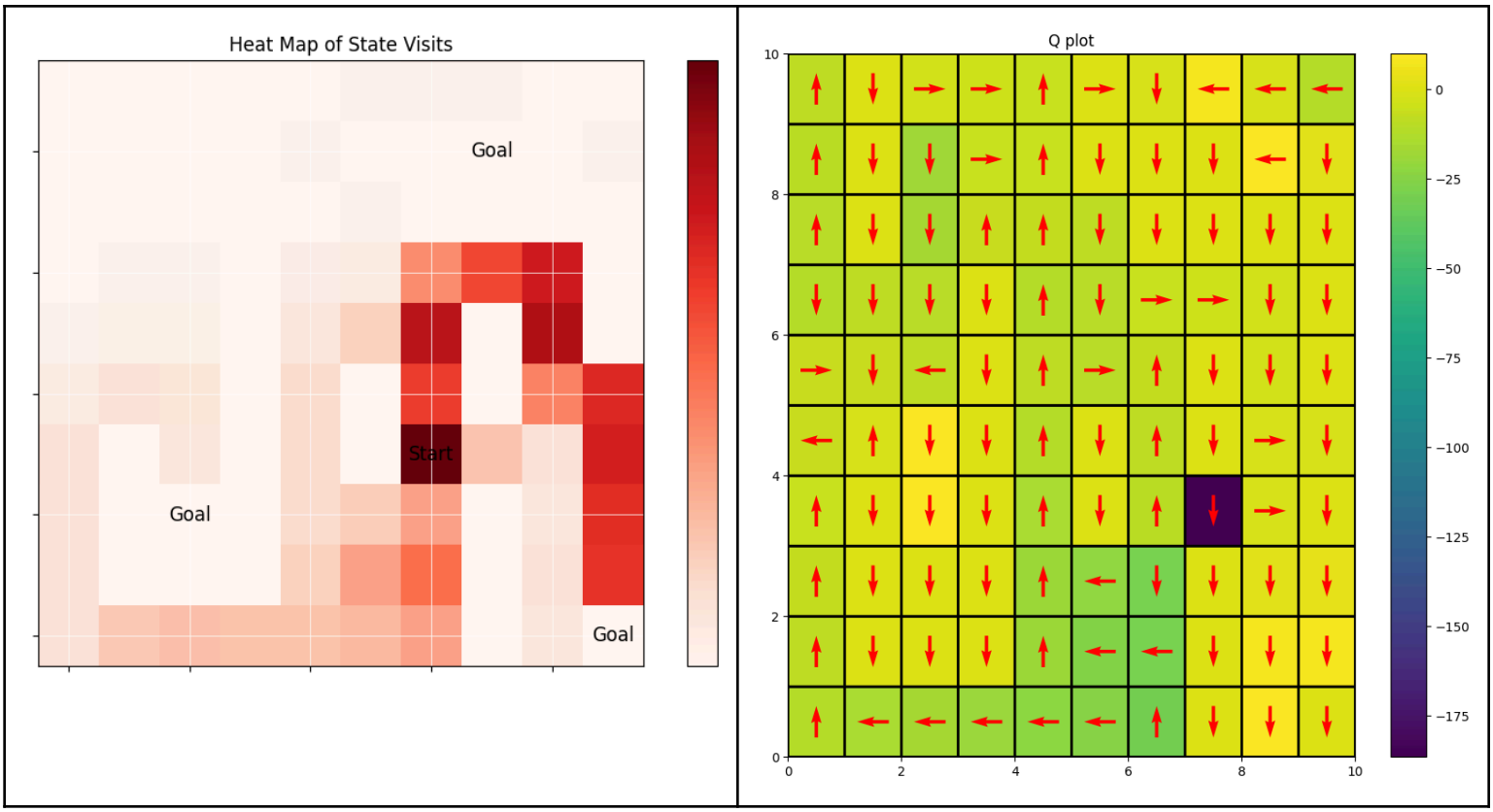
Design Choices:

- Because stochasticity comes from the wind, epsilon greedy policy is not used.
- Since the environment has inbuilt stochasticity via the wind, a lower tau value (a policy that explores less) is sufficient. And lower alpha for stability.
- Because of a lot of randomness, the variance of the reward distribution is high.

Plots:







Experiment 10

Setting:

Start state: [3,6]
Learning Algorithm: Q-Learning
Wind = True
p = 1.0

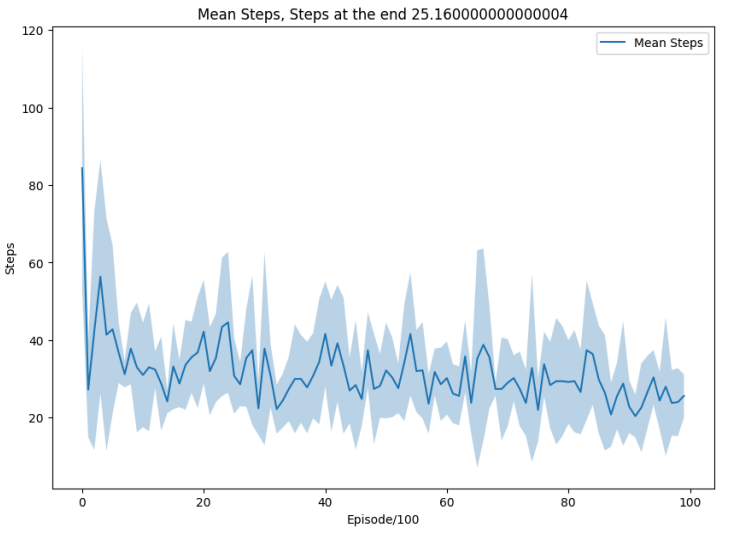
Hyperparametes:

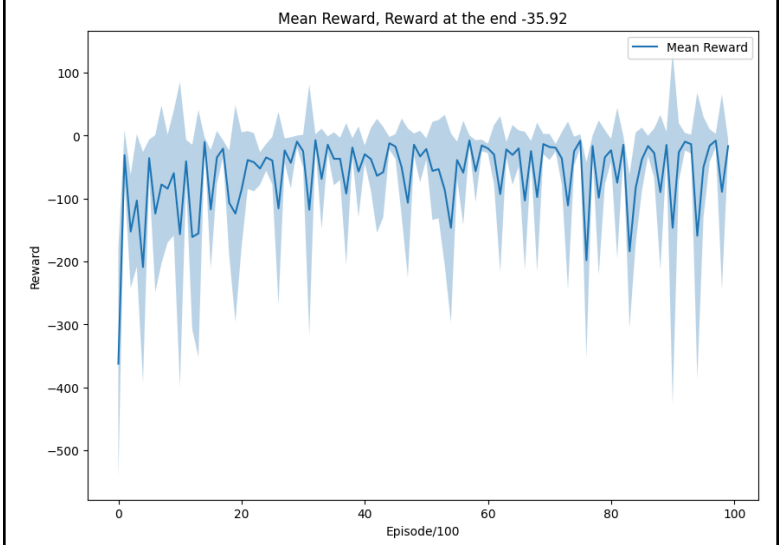
Alpha = 0.5
Gamma = 1.0
Policy = epsilon policy
Epsilon = 0.5

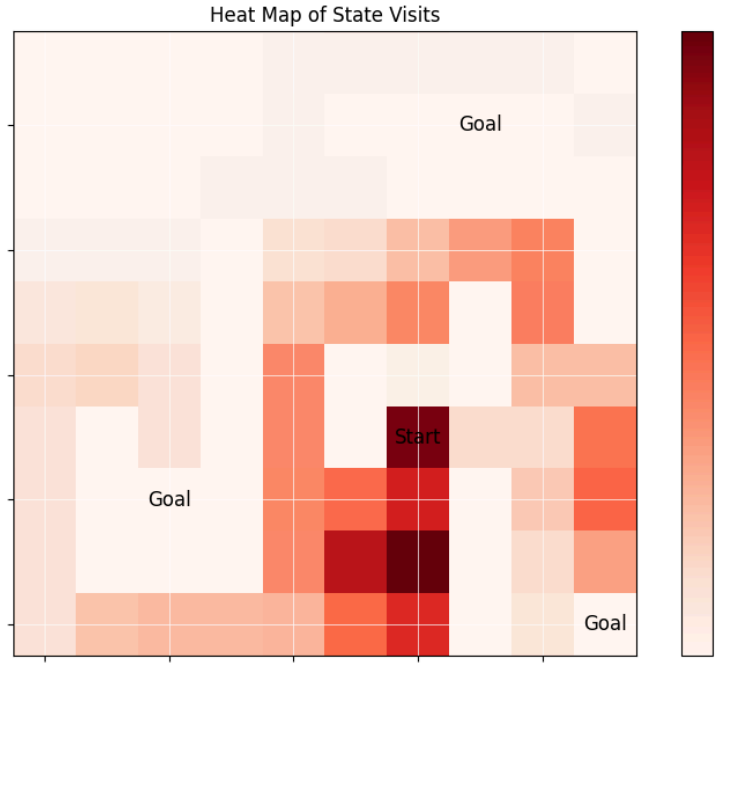
Design Choices:

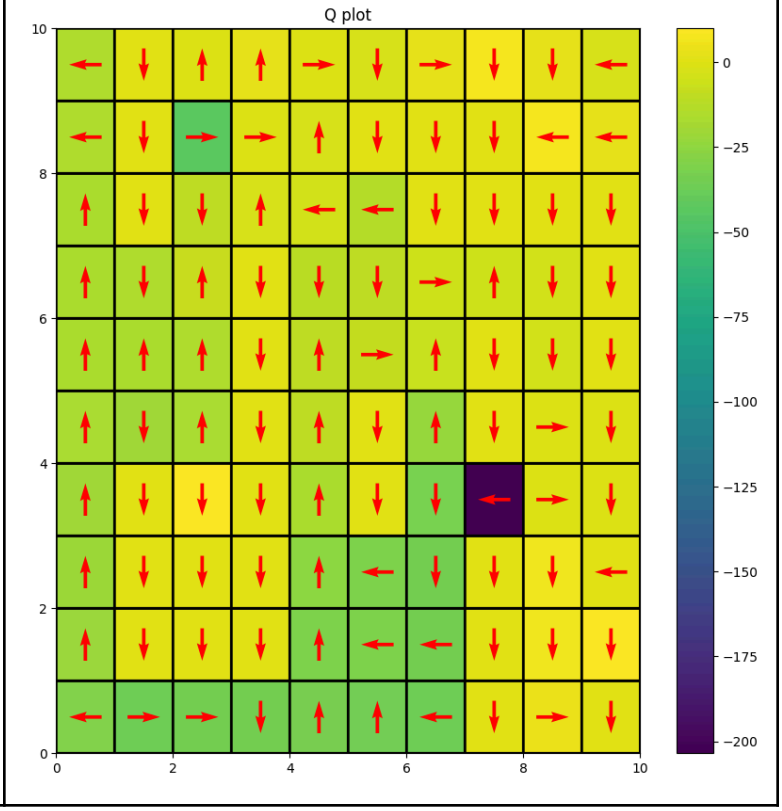
- Because stochasticity comes from the wind, epsilon greedy policy is not used.
- Since Q-learning has less implicit ability to explore, a learning rate of 0.5 is used.

Plots:


Mean Steps, Steps at the end 25.160000000000004
Mean Steps
Steps
Episode/100


Mean Reward, Reward at the end -35.92
Mean Reward
Reward
Episode/100


Heat Map of State Visits
Goal
Start
Goal
Goal


Q plot

Experiment 11

Setting:

Start state: [0,4]
Learning Algorithm: SARSA
Wind = True
p = 1.0 (deterministic step)

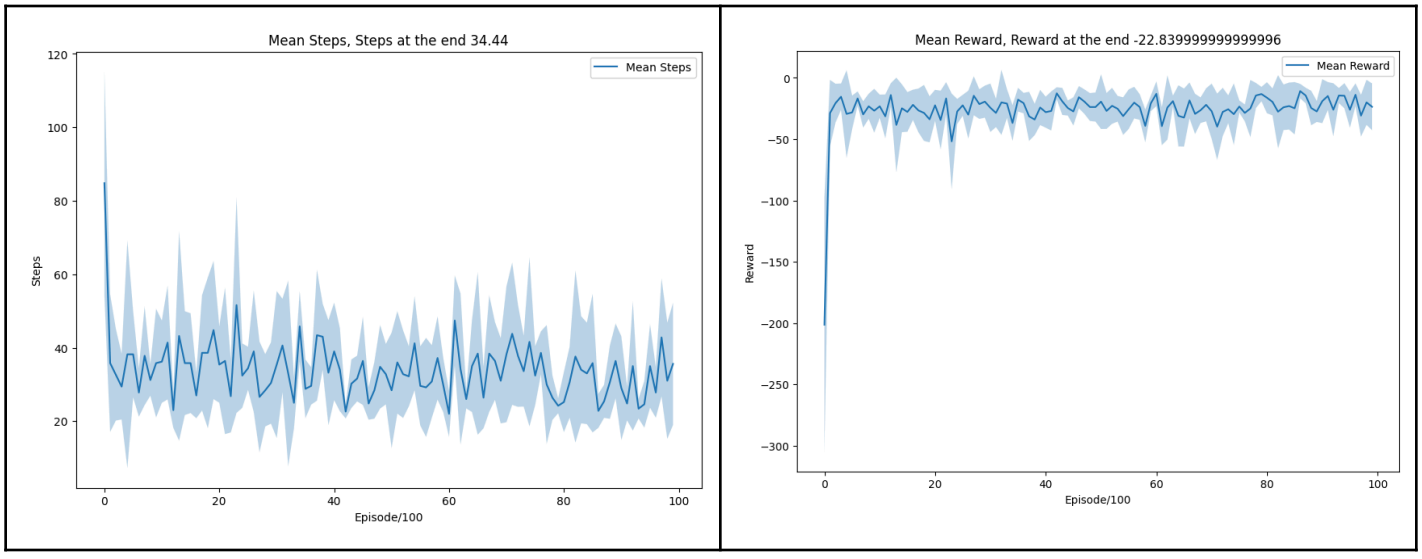
Hyperparametes:

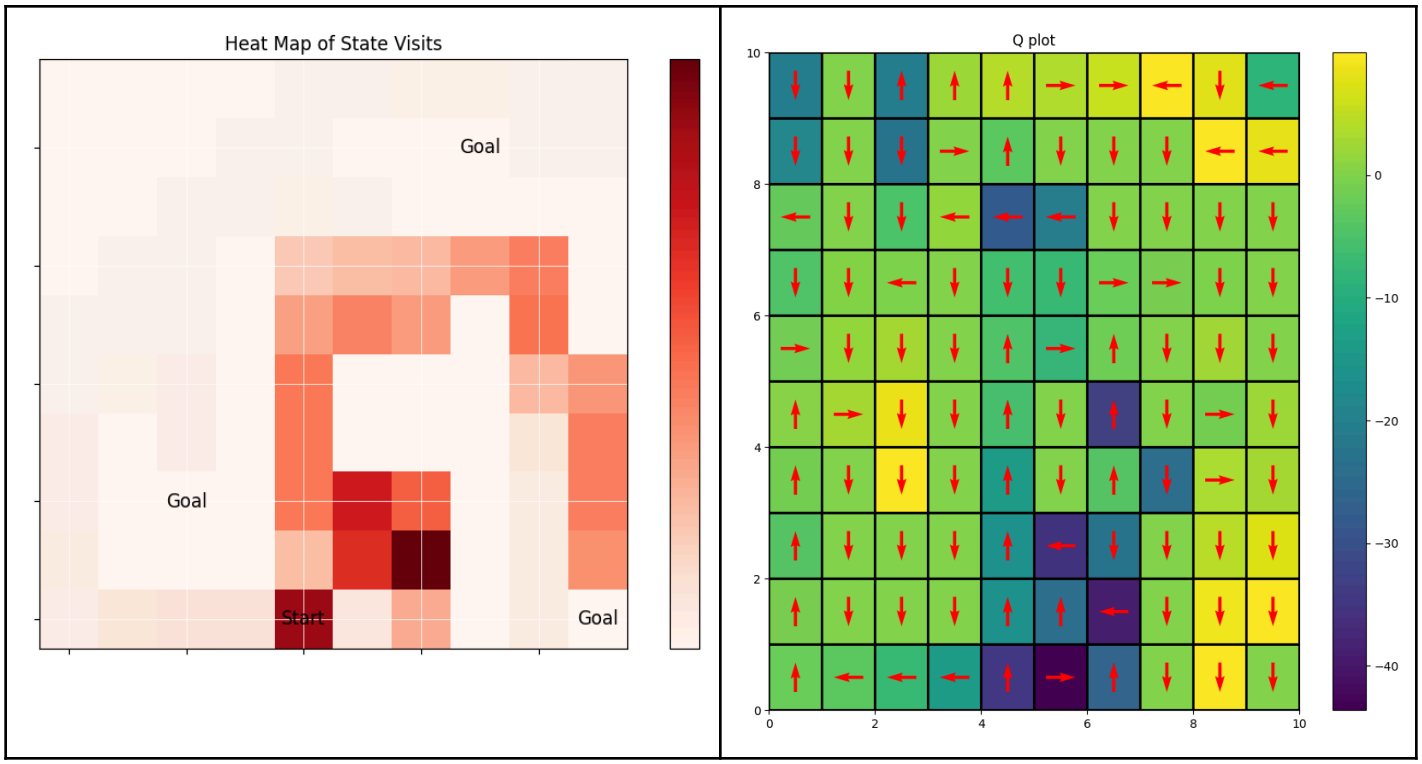
Alpha = 0.5
Gamma = 1.0
Policy = Softmax
Tau = 0.5

Design Choices:

- Because stochasticity comes from the wind, epsilon greedy policy is not used.
- Since Q-learning has less implicit ability to explore, a learning rate of 0.5 is used.

Plots:





Experiment 12

Setting:

Start state: [0,4]
Learning Algorithm: Q-Learning
Wind = True (windy)
p = 1.0 (deterministic step)

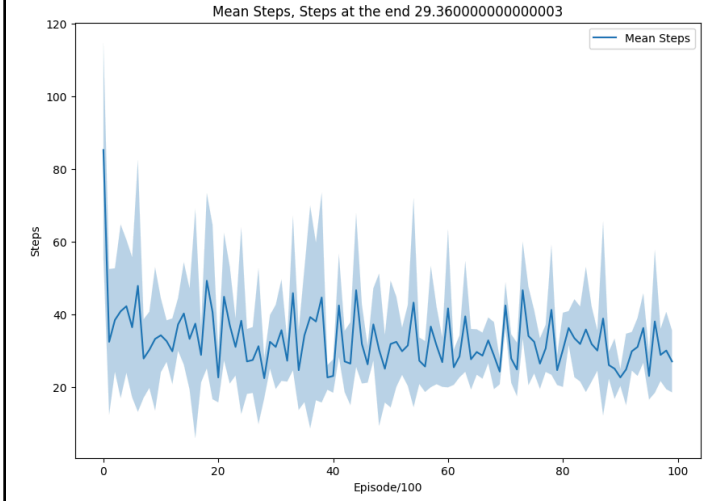
Hyperparametes:

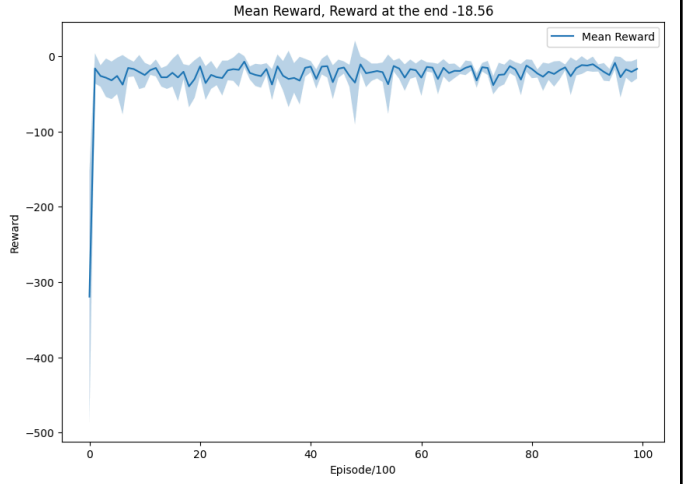
Alpha = 0.5
Gamma = 1.0
Policy = Softmax
Tau = 0.5

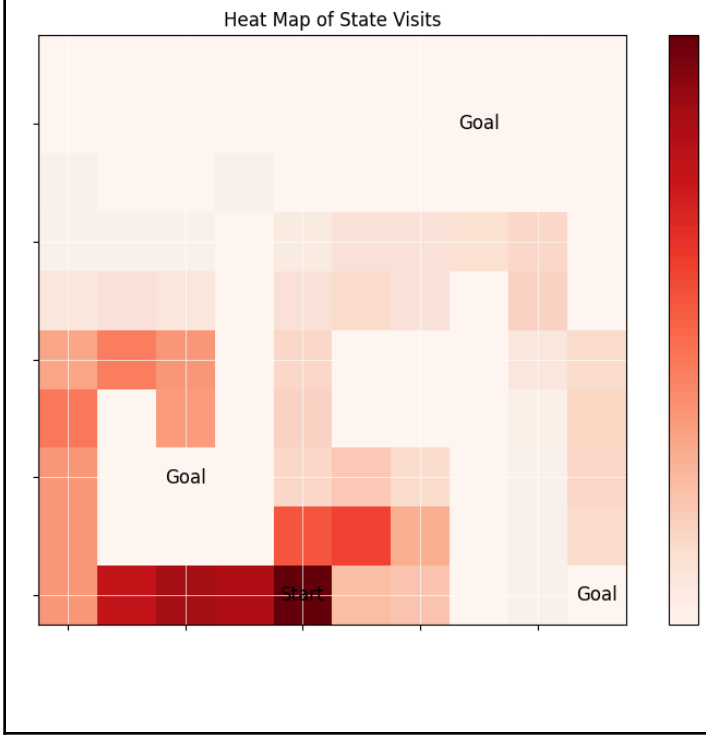
Design Choices:

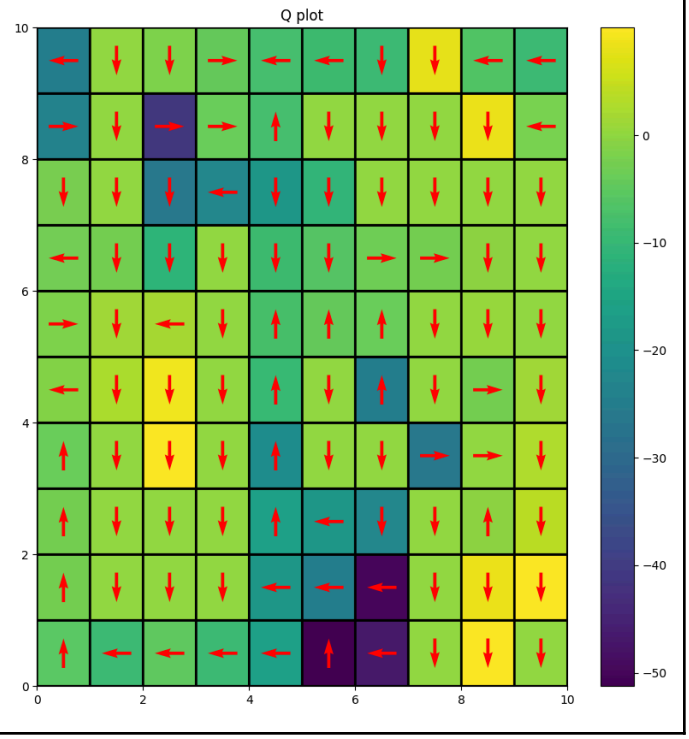
- Because stochasticity comes from the wind, epsilon greedy policy is not used.
- Since Q-learning has less implicit ability to explore, a learning rate of 0.5 is used.

Plots:


Mean Steps, Steps at the end 29.360000000000003
Mean Steps
Steps
Episode/100


Mean Reward, Reward at the end -18.56
Mean Reward
Reward
Episode/100


Heat Map of State Visits
Goal
Goal
Start
Goal


Q plot