Programming Assignment2

Prakhar Thakuria(EE16B061)

15 February 2019

Question1

Solution:

Puddle World

1. There is a wall all around the the grid and if and action tries to go in the wall there is no change that is remains in the same state as before.

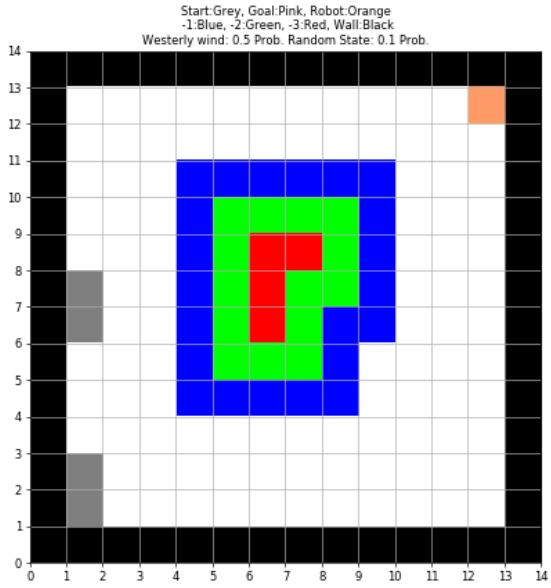
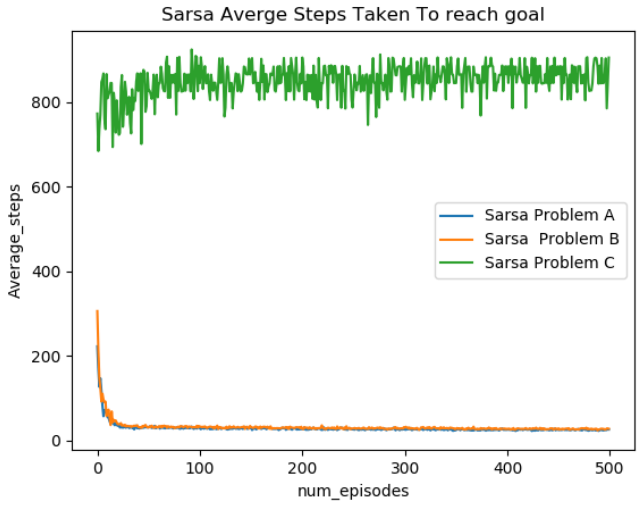
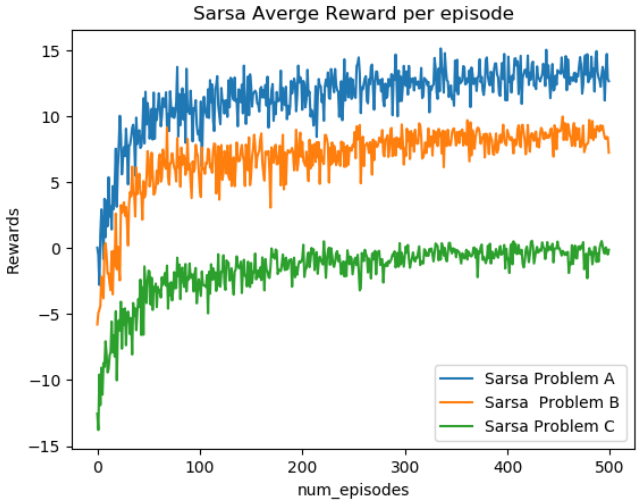


Figure 1: My Puddle World

Puddle

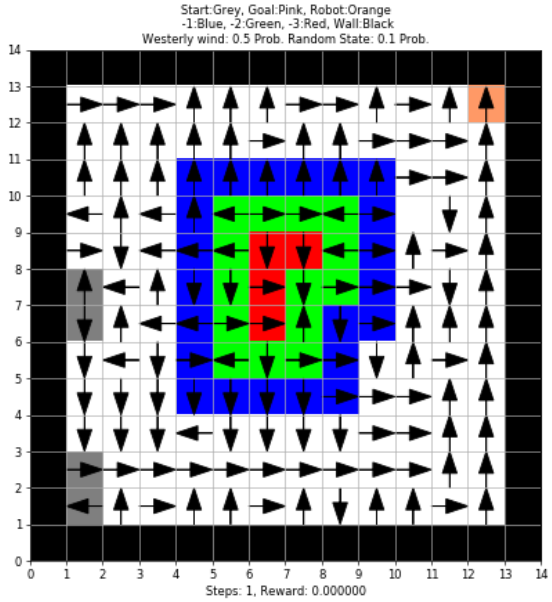


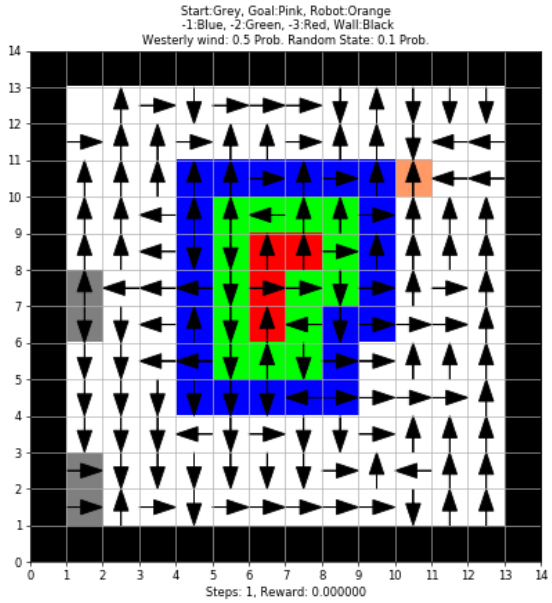
(a) Steps To Goal

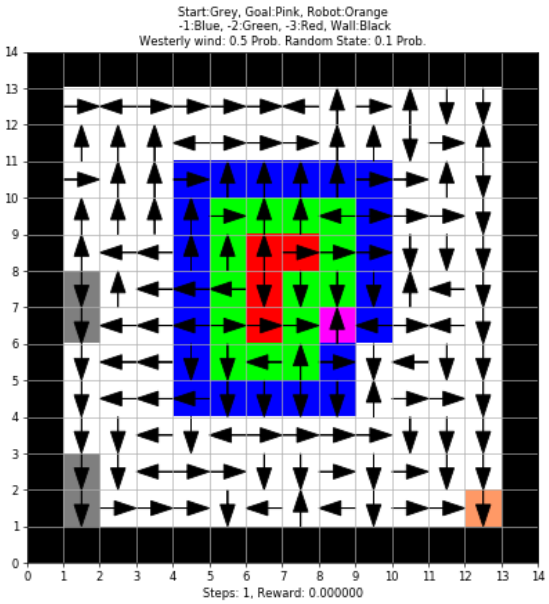


(b) Average Reward

Figure 2: Sarsa All Goals


Start:Grey, Goal:Pink, Robot:Orange
-1:Blue, -2:Green, -3:Red, Wall:Black
Westerly wind: 0.5 Prob. Random State: 0.1 Prob.
Steps: 1, Reward: 0.000000


Start:Grey, Goal:Pink, Robot:Orange
-1:Blue, -2:Green, -3:Red, Wall:Black
Westerly wind: 0.5 Prob. Random State: 0.1 Prob.
Steps: 1, Reward: 0.000000



Sarsa

1. The update rule used was

$$Q(S, A) = Q(S, A) + \alpha * (R + \gamma * Q(S', \pi(S') - Q(S, A))$$

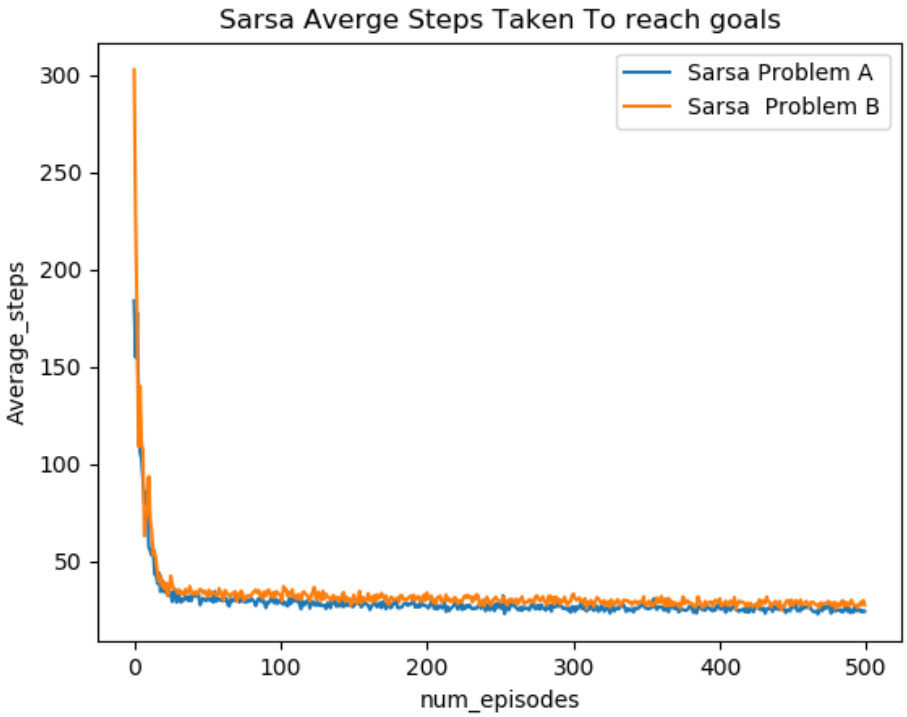
2. I made in a change for the last problem i.e. Goal == C that if does not reach the goal within 1000 steps, it truncates.

Observation

1. It take a lot of time to reach the goal state in the case when goal is C it takes an average of over 10,000 steps to converge to the goal state while for others it is around 250.

This is due the fact Goal state C is covered by negative reward and hence if initialization 0 it tries to look at the possibilities before reaching the goal state.

2. Since the goal state is A relatively far away from the puddle hence average reward is higher, while in the case of goal B as it is very near to the puddle the



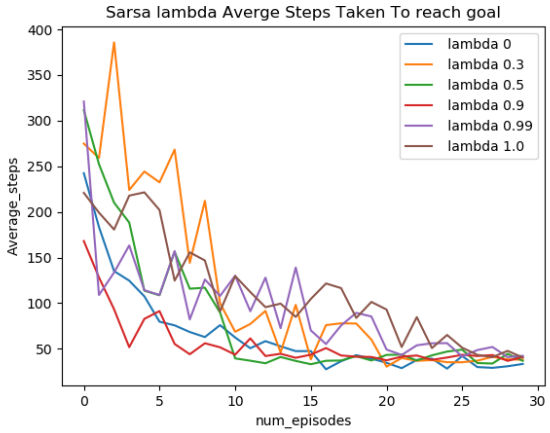
reward is relatively lower as it in the neighbourhood of the puddle. Similar explanation can be derived for goal state C

3.For the optimal policy in the case A the puddle is , but in the case B also tries to move out of the puddle but to an extent it can also take one or two step in the puddle if ti take it closer to the goal through the puddle. but in the case of goal state C the optimal policies also go through the whole puddle for some states.

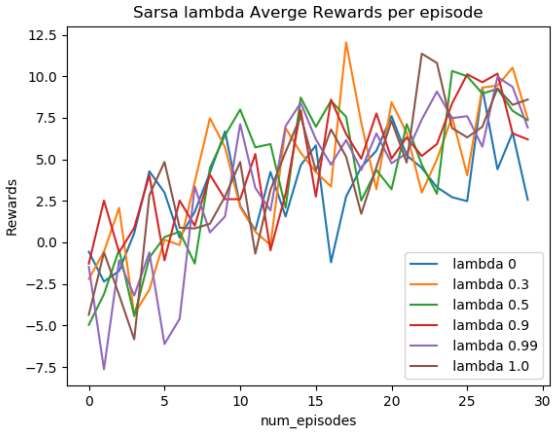
4. As the the steps for goal states were not quite clear in the previous figure because of goal state C the above is a better view of the problem. 5. Since there is no negative rewards for the steps taken to reach goal it does not learn the shortest path to the goal.

Question3

Solution:



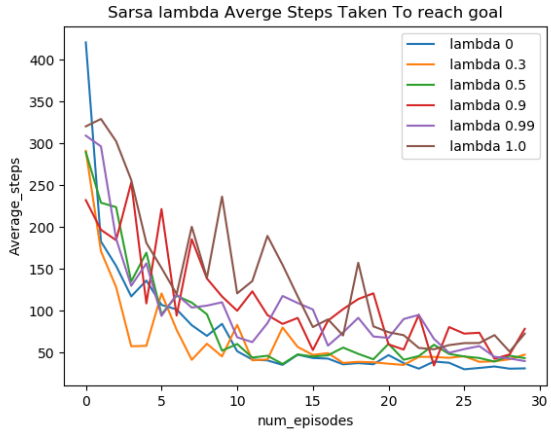
(a) Steps to Goal



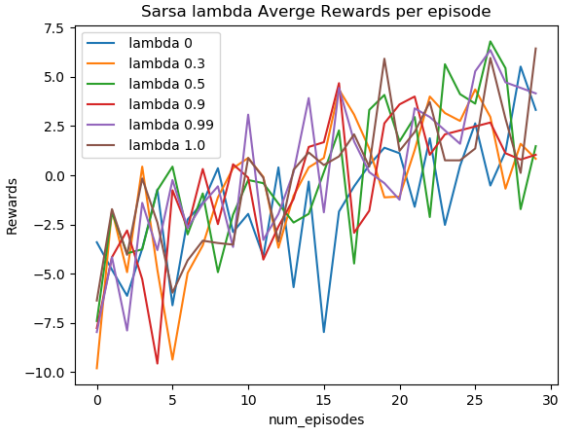
(b) Average Reward

Figure 3: Sarsa Lambda goal A

Go to te last few pages to see the optimal policy for sarsa-lamda



(a) Steps to Goal

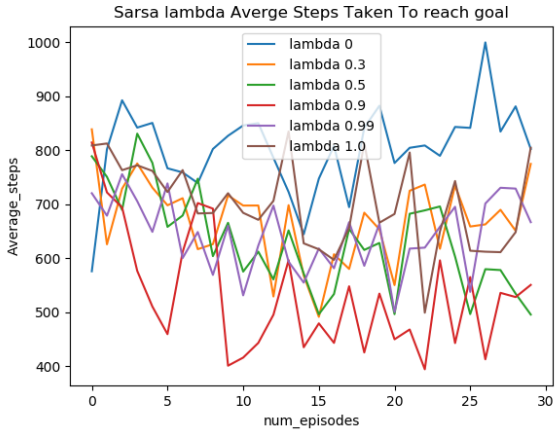


(b) Average Rewards

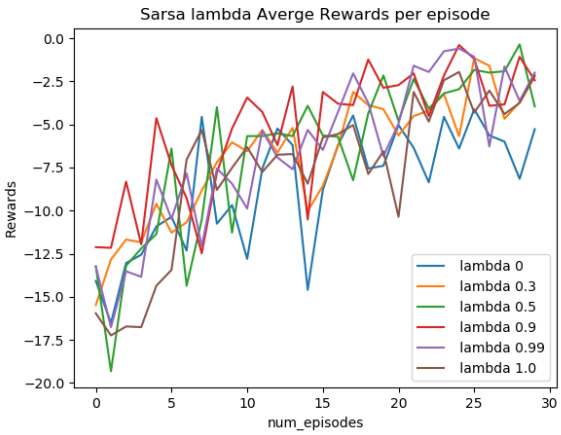
Figure 4: Sarsa Lambda Goal B

Sarsa(λ)

1. Used value of Lambda were: 0,0.3,0.5,0.9,0.99,.1.
2. Accumulating traces were used for the development of the following graphs, as it seemed to have a little less variance than replacing traces.
3. There was a cap at of 1000 steps for termination.



(a) Steps to Goal



(b) Average Reward

Figure 5: Sarsa Lambda Goal C

Observation

1. Sarsa Lambda takes a lot time to complete as the number of updates is lot more than the Sarsa and hence a lot less number of updates can be done and hence the all the possibilities are not explored. The number of iteration don't seem to be enough rather to make meaning full conclusions .
2. For goal state == A, as the goal state is much far way from the puddle every lambda seems to be doing almost same. but in the case of goal state

B returns tend to increase in the start and then decrease and lambda = 0.99 or 0.9 seams to have taken the lead but still the variance is just too much to say anything for sure.
3. Sarsa lambda takes care that the first step is i the right direction, which is not in the case of sarsa and the direction goes on for avoiding puddle as the lambda increases.(Refrence to last page where optimal policies are given)
3. There is a definite increase in the average reward in a lot less episodes than sarsa lambda over sarsa. What sarsa-lambda was able to achieve in 30 episodes sarsa took more than 100 episodes to do.
4. Again the rewards as in the case for the goal C are very low but they increase as move along the episodes 5. Since there is no negative rewards for the steps taken to reach goal it does not learn the shortest path to the goal.

Question4

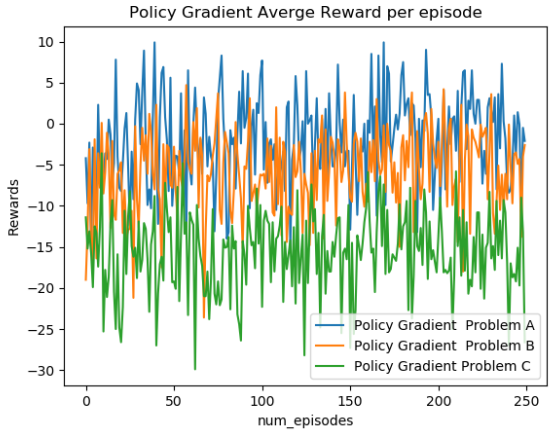
Solution:

Learning parameters used

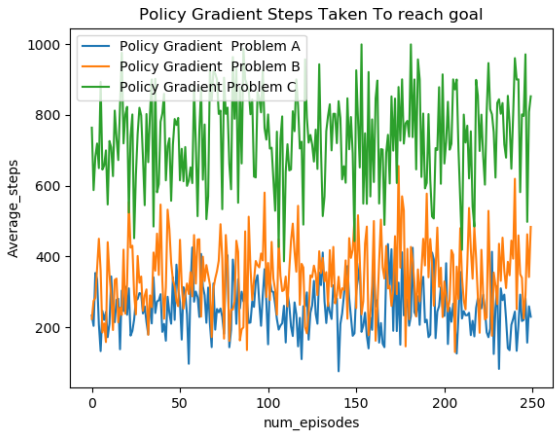
$$\alpha = .001 \gamma = 0.9$$

Policy Gradient Monte Carlo

Since Monte Carlo Policy Gradient uses episodes were generated and the limit was 1000 steps before that it the goal is not achieved then the the episode it terminated. And the the iteration used were 10 , and episodes used were 250



(a) Average Reward



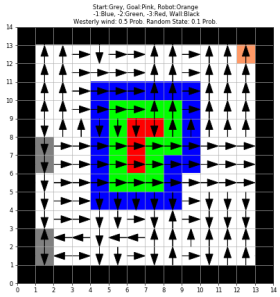
(b) Steps To Goal

Figure 6: Policy Gradient all problems

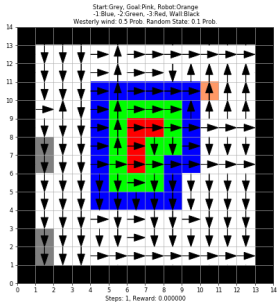
Observation

1. In this case the average steps used for the terminating at the goal don't change that much as there is no compensation used for the number of steps in this case.

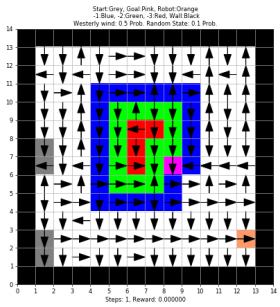
$$\alpha = .001 \gamma = 0.9$$



(a) Problem A



(b) Problem B

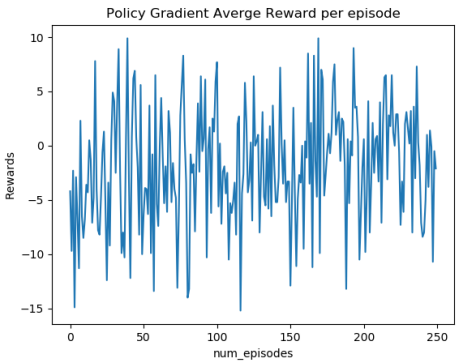


(c) Problem C

Figure 7: Optimal Policies learnt $\alpha = 0.001 \gamma = 0.9$

2. The average reward increases a little bit although there is not much change(as compared to sarsa-lamda or even Sarsa). It still remains quite random, which just goes to show that it not a very good representation of the state space. This might be because the discount factor is to much and

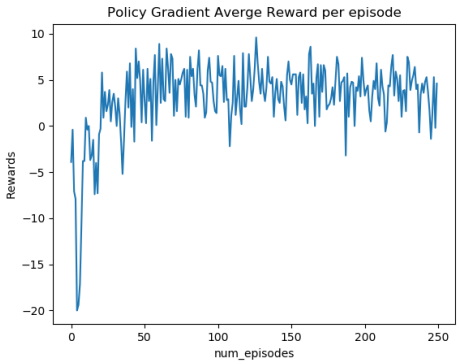
alpha is too low(As the number of iteration and episodes are limited we don't the luxury to use such a small alpha and gamma), hence I increased that for the next iteration.



goal a.png

Figure 8: Policy Gradient Reward goal A

$\alpha = 0.1 \gamma = 0.99$



goal a.png

Figure 9: Policy Gradient rewards A $\alpha = 0.1 \gamma = 0.99$

3.This seems to be doing quite good even for the other cases as well as there is a constant increase in the reward but the the optimal policy learnt depend highly on the westerly wind in this case.

4. As we can see the policy learnt for this is

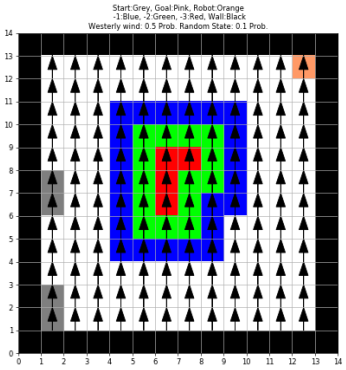


Figure 10: Optimal Policy Goal A $\alpha = 0.1 \gamma = 0.99$

5. The complete picture for the parameters $\alpha = 0.1 and \gamma = 0.99$:

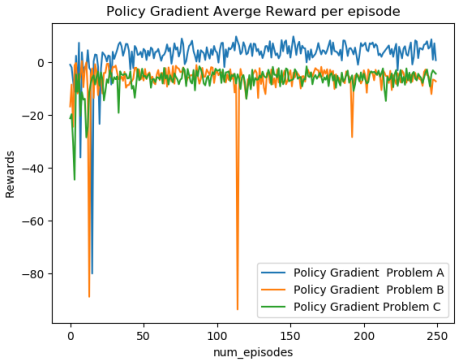


Figure 11: Average reward all goals $\alpha = 0.1\gamma = 0.99$

Question 5

Solution:

Update Equations

$$\theta = \theta + \alpha\gamma G\Delta_\theta ln(\pi(A_t|S_t,\theta))$$

For SoftMax

$$\Delta_\theta ln(\pi(A_t|S_t,\theta)) = 1 - \frac{1}{\Sigma(exp(\theta_x(x,a) + \theta_y(y,a))}$$

As

$$\pi(A_t|S_t,\theta) = \frac{exp(\theta_x(x,A_t) + \theta_y(y,A_t))}{\Sigma(exp(\theta_x(x,a) + \theta_y(y,a))}$$

Comments On the suitability

1. For the given problem this is a not suitable at all as we can see we would get much better returns and much better policy in the with Sarsa.
2. But in general it is quite use full to have such an parametrisation as it has limited number of parameters but still is able to have diff rent action probabilities in diff rent states. If the state space was to blow up we would not be able to maintain Q for all the state action pair and updating them would become even harder. Already given the episodes were less than 100 were not able to update all the state-action pairs. Hence, in such case the parametrization would help a lot if the states were to blow up.

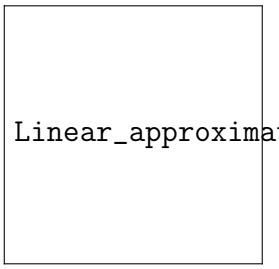
Question6

Solution:

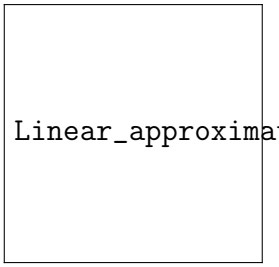
No we won't be able to run the same code for the as the number of states hane now blown it won't be possible to run the same as the number of parameters have also grown exponentially.

Results

This being time taking process i ran it for just Goal A and the Average return results are:



(a) Steps to Goal



(b) Average Rewards

Figure 12: Linear approximator

Sarsa lamda optimal policy

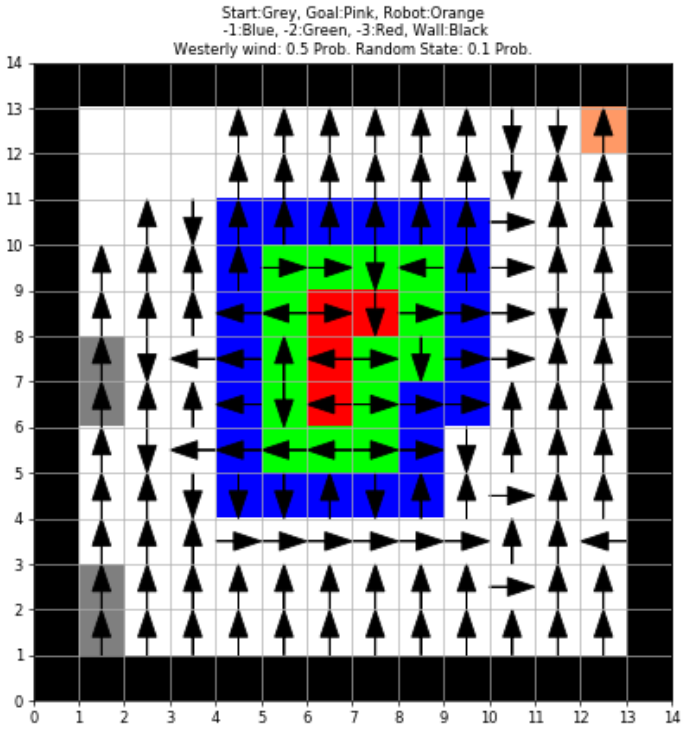


Figure 13: Sarsa lamnda(0) A

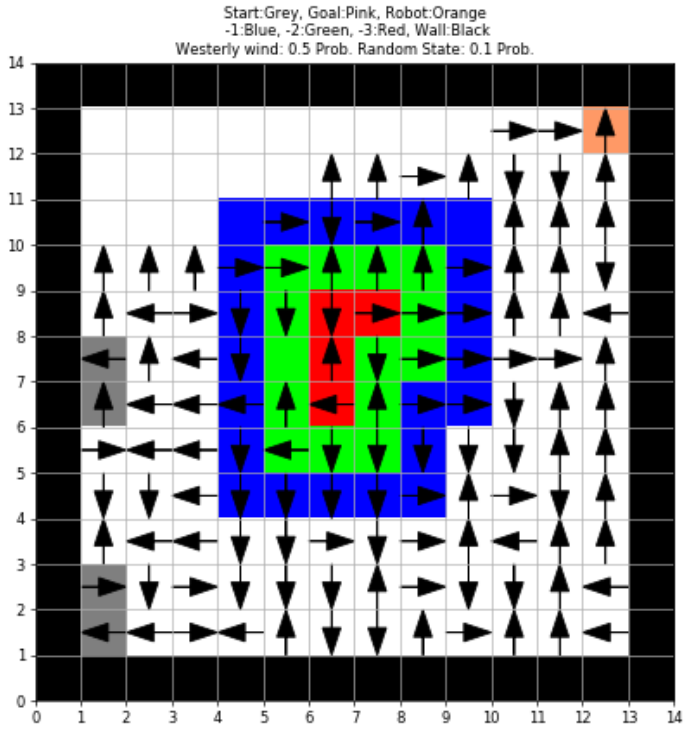


Figure 14: Sarsa lamnda(0.3) A

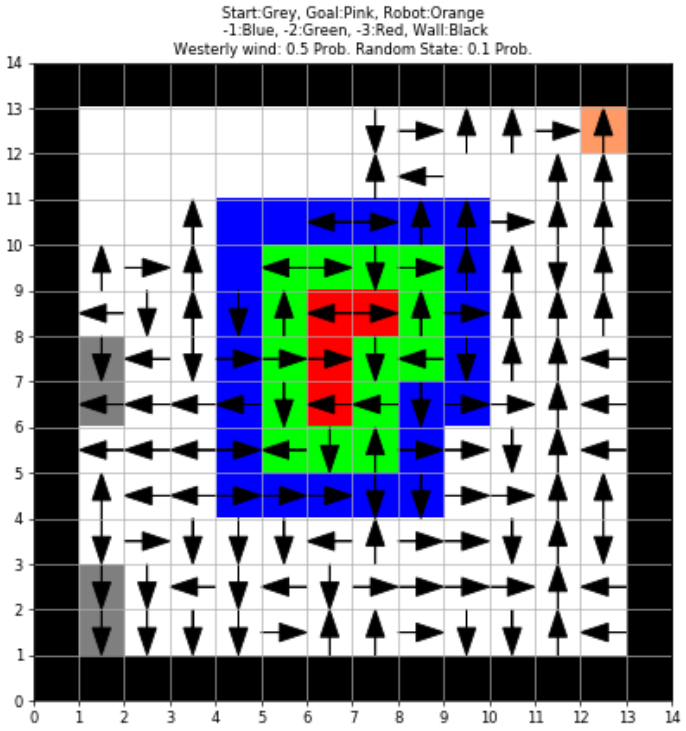


Figure 15: Sarsa lamnda(0.5) A

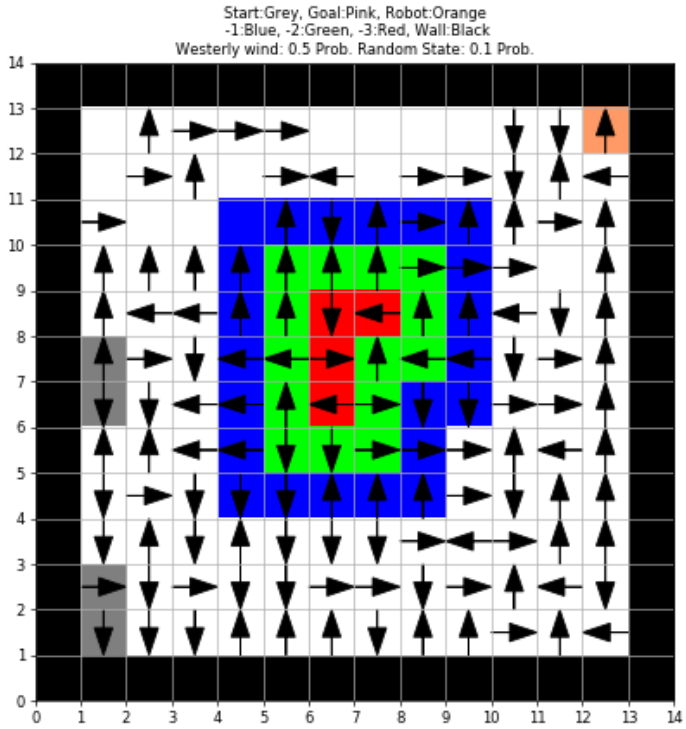


Figure 16: Sarsa lamnda(0.9) A

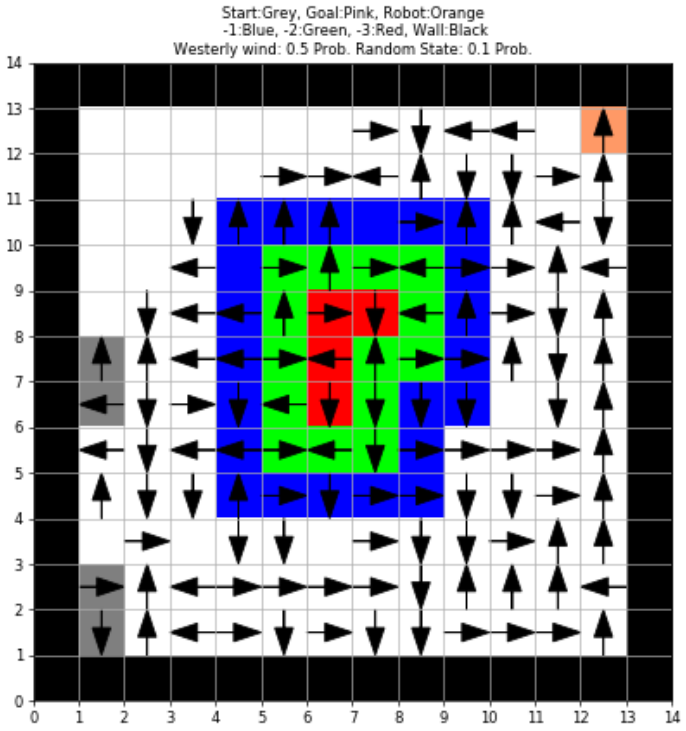


Figure 17: Sarsa lamnda(0.99) A

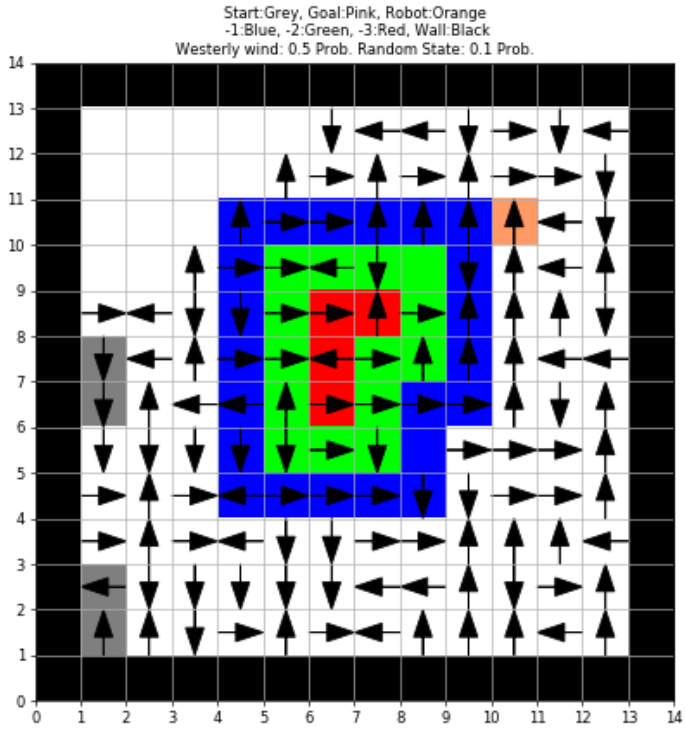


Figure 18: Sarsa lamnda(1) B

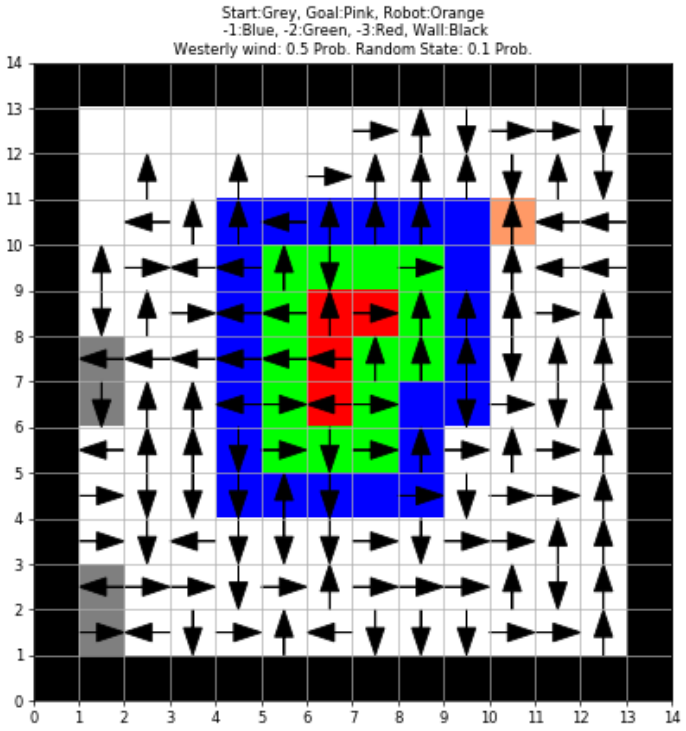


Figure 19: Sarsa lamnda(0.3) B

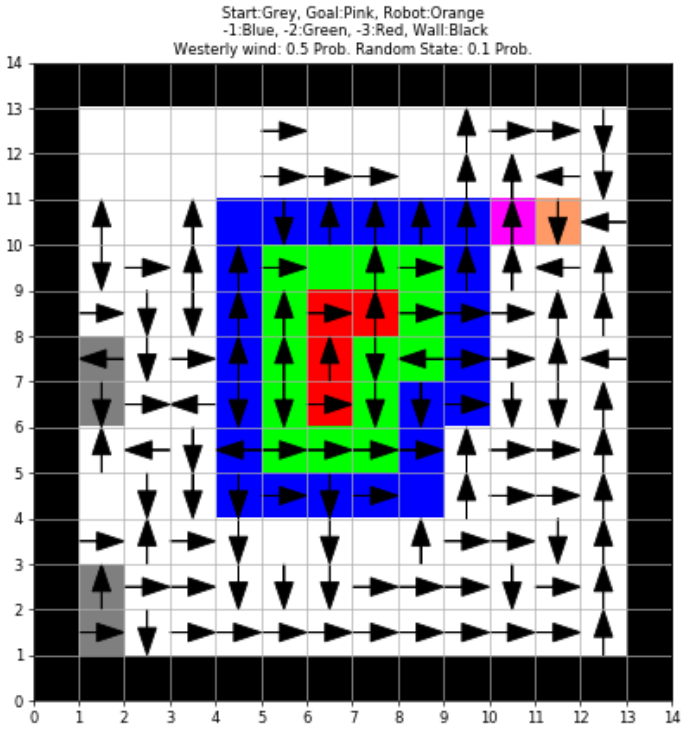


Figure 20: Sarsa lamnda(0.5) B

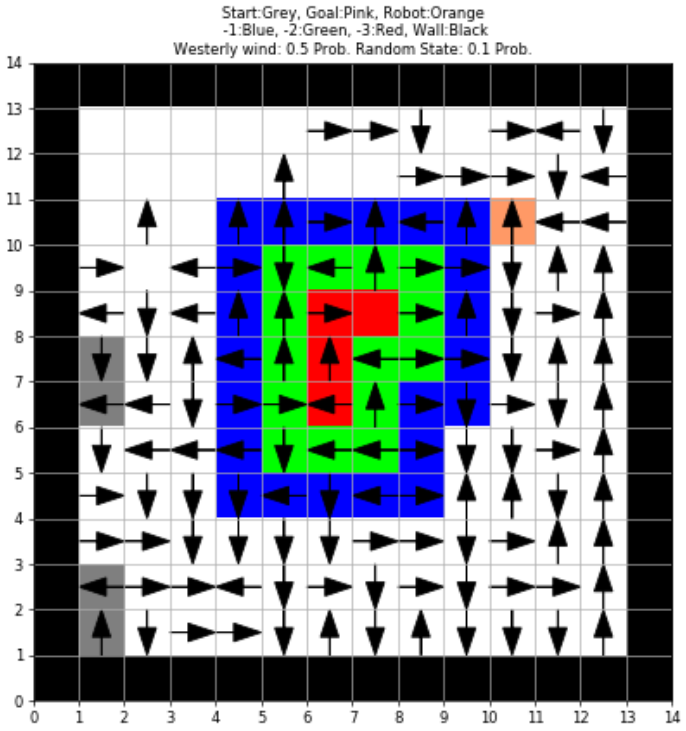


Figure 21: Sarsa lamnda(0.9) B

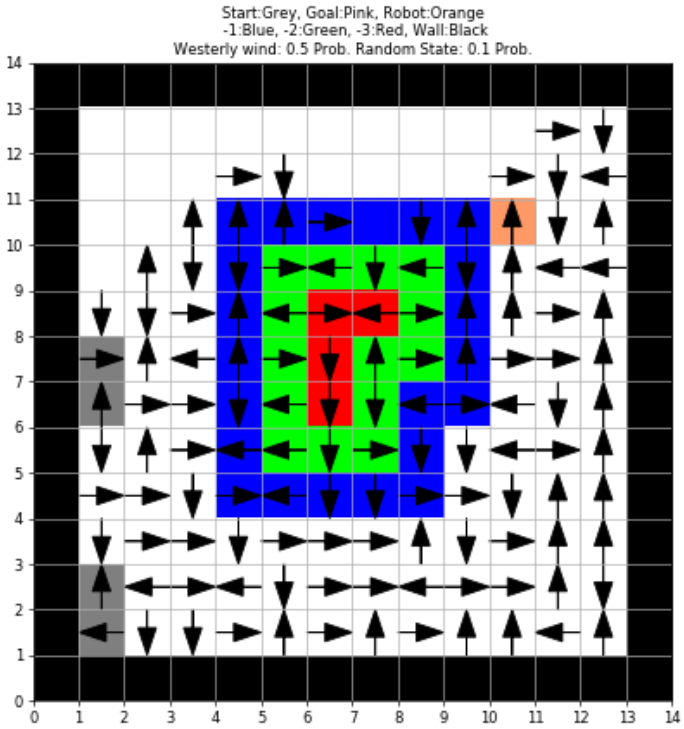


Figure 22: Sarsa lamnda(0.99) B

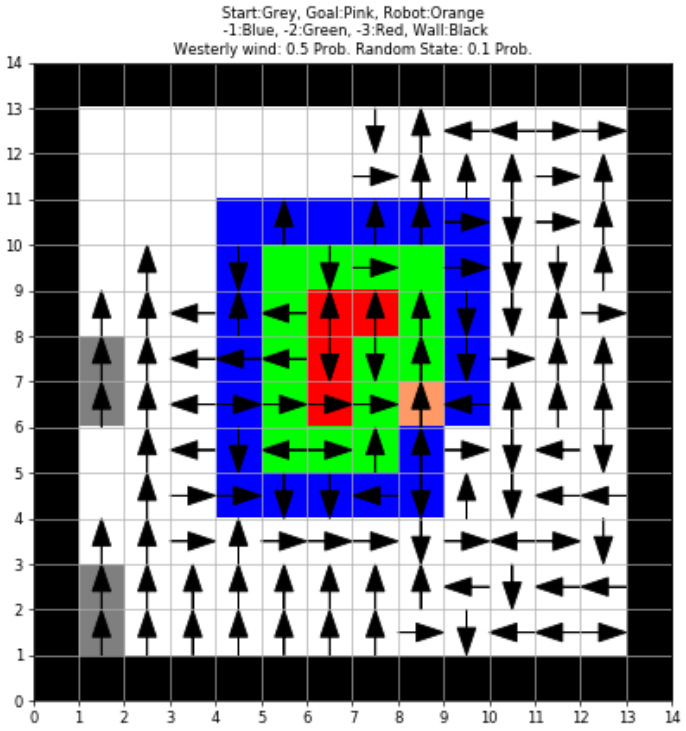


Figure 23: Sarsa lamnda(0) C

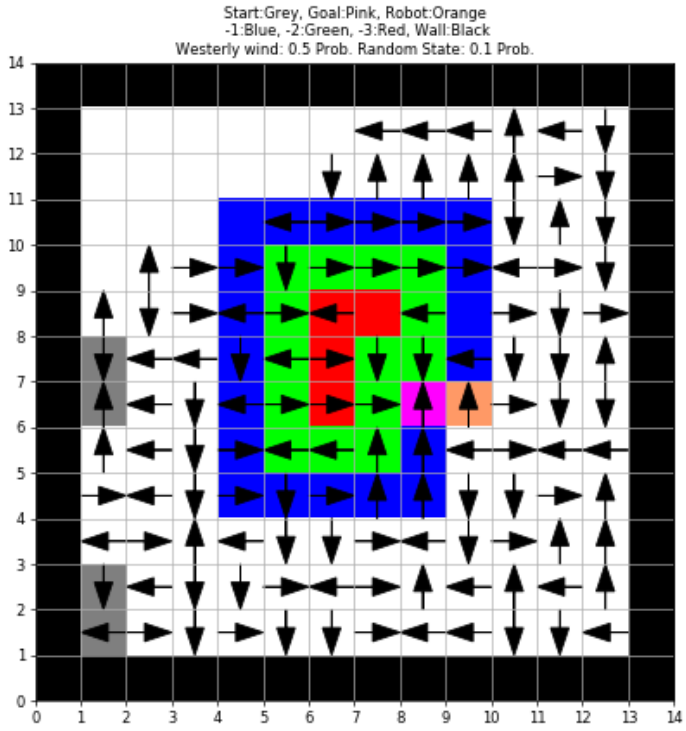


Figure 24: Sarsa lamnda(0.3) C

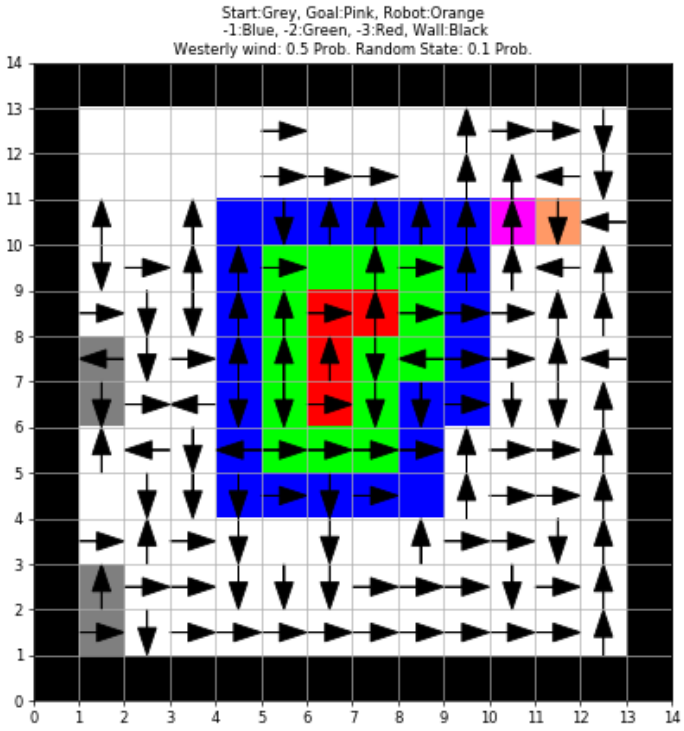


Figure 25: Sarsa lamnda(0.5) C

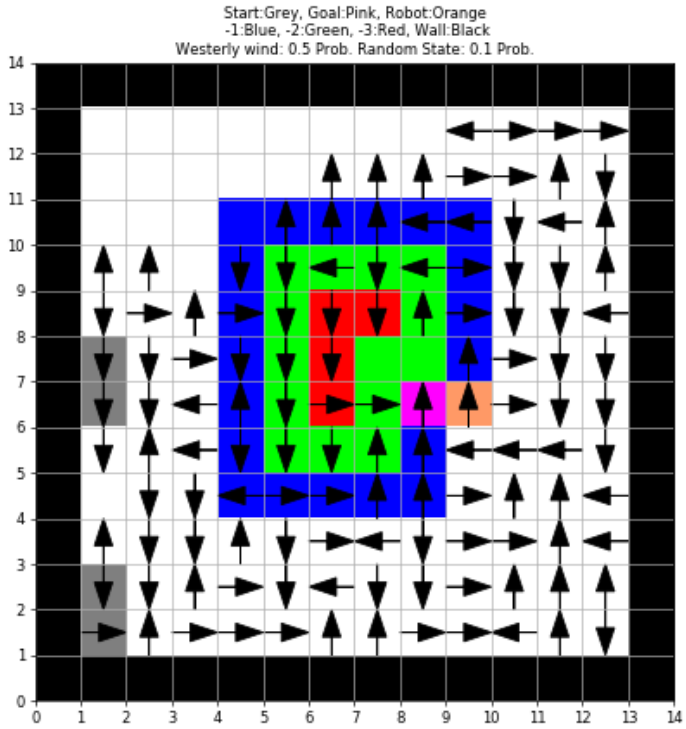


Figure 26: Sarsa lamnda(0.9) C

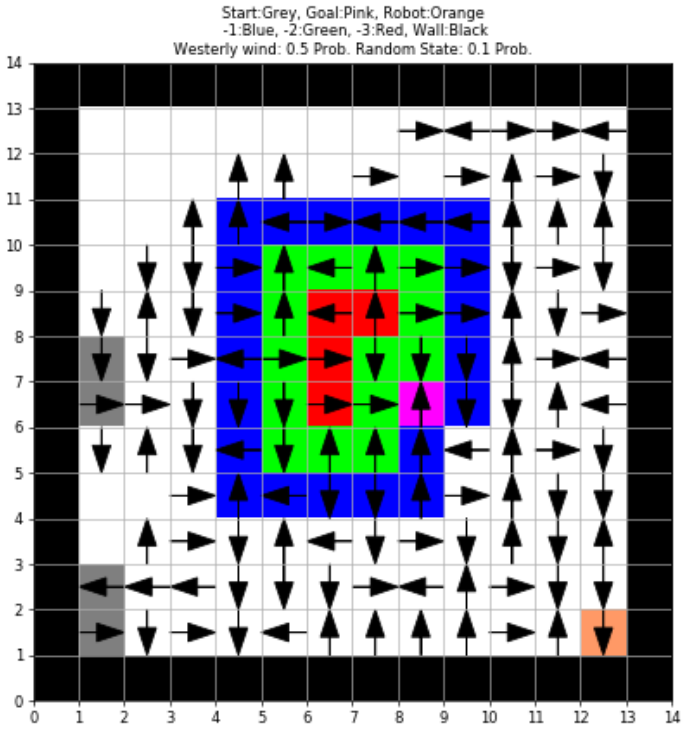


Figure 27: Sarsa lamnda(0.99) C