COL864 ASSIGNMENT 2 REPORT

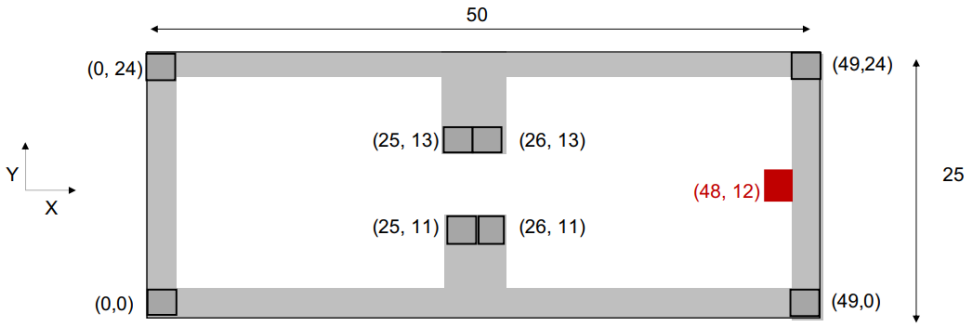
Sagar Sharma (2018CS10378)

1 SOLVING AN MDP

We will solve the given MDP using value iteration
We will use bellman update/backup equations

$$V_{k+1}(s) \longleftarrow max_a \Sigma_{s',r} T(s,a,s')(R(s,a,s') + y * V_k(s')) \tag{1.1}$$

Initialization



- Each grid cell contains $\Longrightarrow$ type(wall,cell), reward(-1,0,100), value, policy action (North,East,South,West)
- Policy is intialized to all North
- Value all are initialized to 0
- Wall cells value are never used.

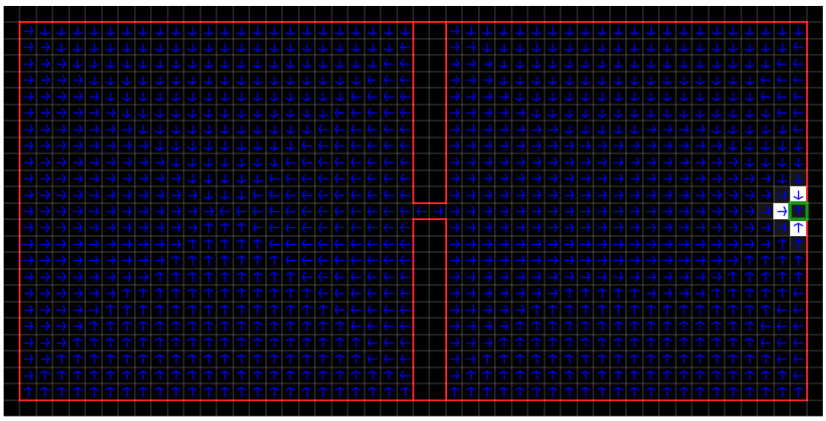
1.1 SOLVE USING VALUE ITERATION

Using the bellman update above we get the algorithm

procedure solve_MDP

1. Initialise grid world
2. for k in range(100):
3. for each cell/state in grid:
4. $V_{k+1}(s) \longleftarrow max_a \Sigma_{s',r} T(s,a,s')(R(s,a,s') + y * V_k(s'))$
5. $\Delta = max(\Delta, |V_{k+1}(s) - V_k(s)|)$
6. if $\Delta < 0.1$ break

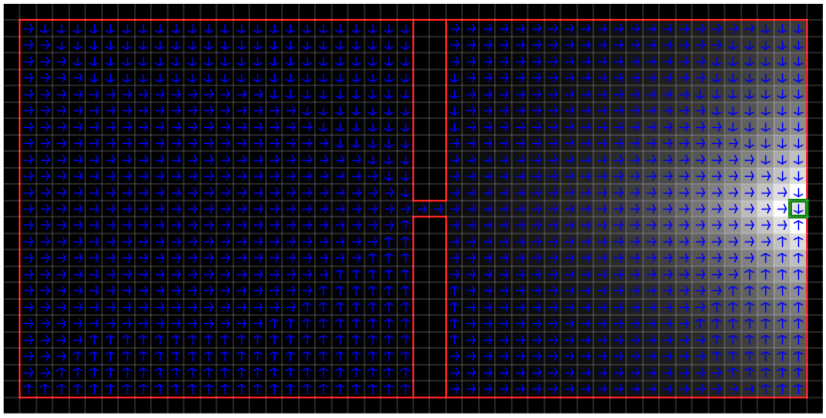
Using the above algorithm with y=0.1 we get



Note: we have ignored theta = 0.1 and run for 100 iterations only as given in question
Observation :

- We can clearly see from policy that the mdp solver pays little attention to distant rewards
- The policy pays more attention to not colliding with wall and get negative reward.
- In first half of the grid the policy indicates that there is almost no effect of distant 100 reward and all policy is trying to do is bring agent to the center of the first half.
- In second half there is effect of goal reward, but still the priority is first given to not colliding with wall as that is more immediate threat than distant reward.
- value of goal cell is lower than adjacent because we have treated it like any other cell.

Using the above algorithm with y=0.9 we get

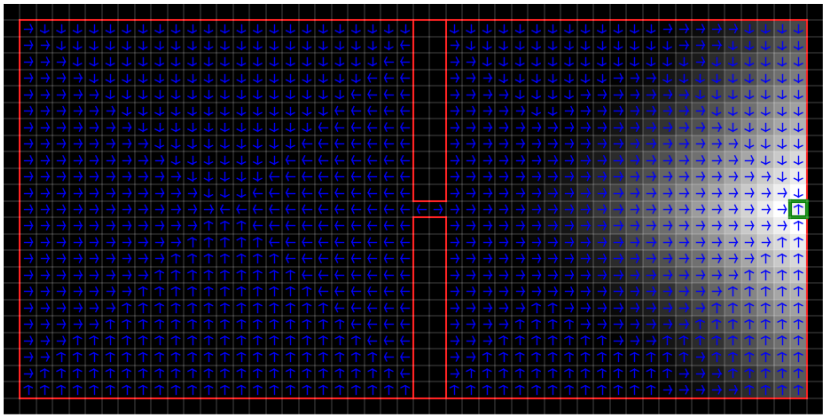


Observation :

- The policy pays a lot more attention to distant reward
- We can observe in first half middle walls, instead of avoiding to collide with the walls the policy is to reach goal as quickly as possible and thus the arrows run towards the opening
- Though around rest of the walls in first half, policy is still to avoid collision first. But policy at opening to get to the goal.
- In second half there is effect of goal reward, there is no worries of colliding and policy is giving more priority to reach goal state for large reward.

1.2 **Increasing y value to 0.99**

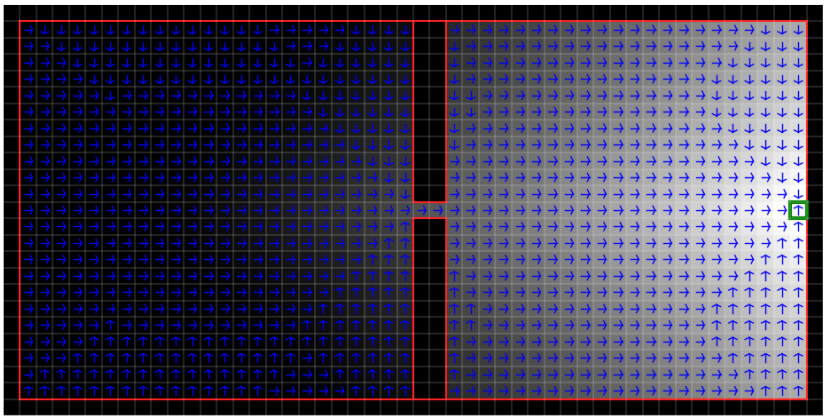
Snapshot at t= 20



Observation :

- The first half, has not yet realised the large reward at goal and thus the policy in first half is more about avoiding the walls and go towards the center of first half.
- In second half effect of goal is more profound, the grey region prioritises the goal over collision. The policy in grey region is all about reach goal cell.
- In black region second half there is less effect of goal reward, around boundaries the policy is still to avoid collision

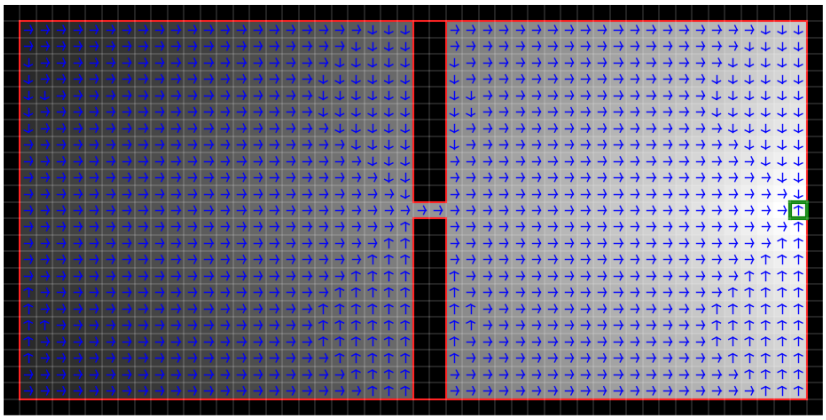
Snapshot at t= 50



Observation :

- The first half, has realised the large reward at goal and thus the policy in first half is a mixture of avoiding left, up and bottom walls and rest of the time to get to the opening.
- Even around the walls of opening the priority is given to reaching the opening rather than avoiding collision.
- In second half all the priority is given to reaching goal cell, the policy is not trying to avoid the walls but rather reach goal state quickly.

Snapshot at t= 100

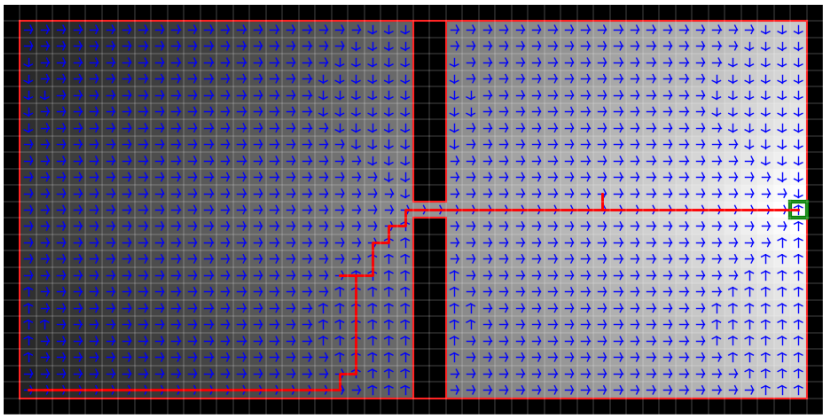


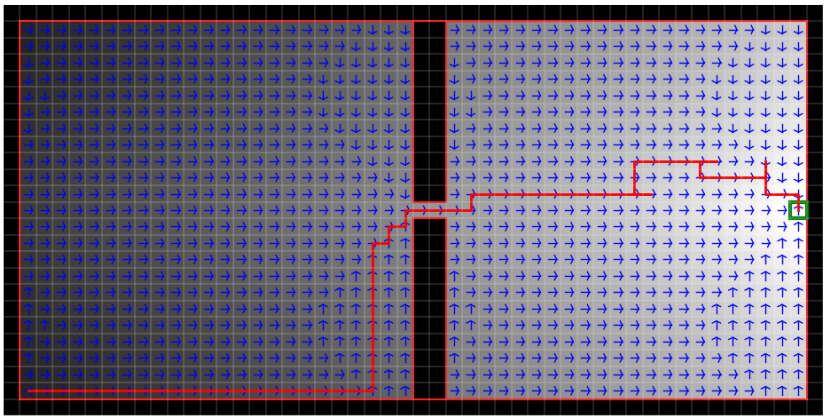
Observation :

- Whole grid has realised large reward, and thus all priority is towards reaching it.
- In first half, the policy is to reach the opening
- In second half all the priority is given to reaching goal cell, the policy is not trying to avoid the walls but rather reach goal state quickly.
- Anywhere in the grid there is not attempt to avoid the walls but rather they are trying to maximise their distant rewards.

1.3 RUNNING THE POLICY

For sample run we get

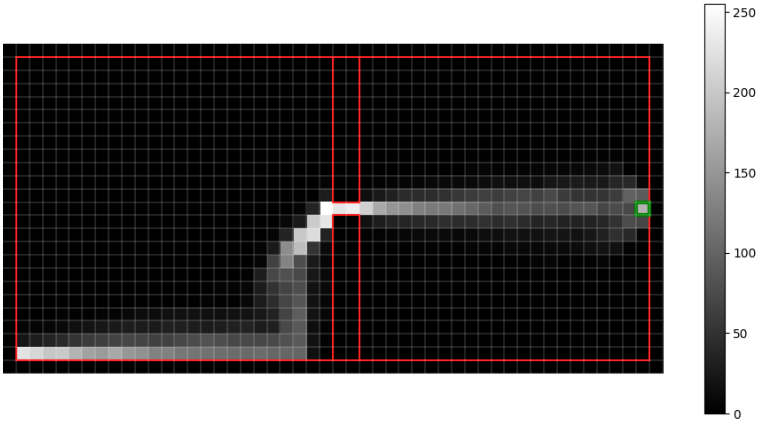




Note : We kept episode length such that once it reaches goal state it finishes. **Observation** :

- We can see from the above two runs that policy is correct and take the agent towards the goal state quickly.
- We have treated goal cell just like any other cell with large reward, if episode was longer the agent reaches it and keeps circling around it.
- We can also observe that actions are stochastic, and the agent ends up moving against the policy but it comes right back in after 2,3 steps.

For episode length 200 and running policy execution 200 times for y=0.99 we get

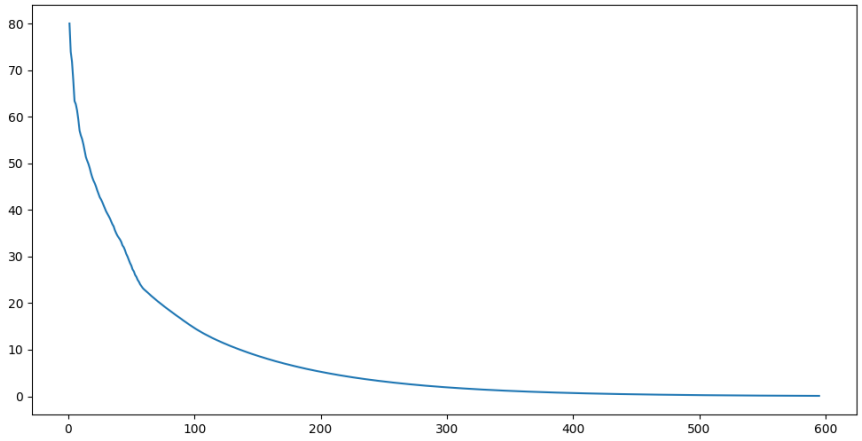


Observation :

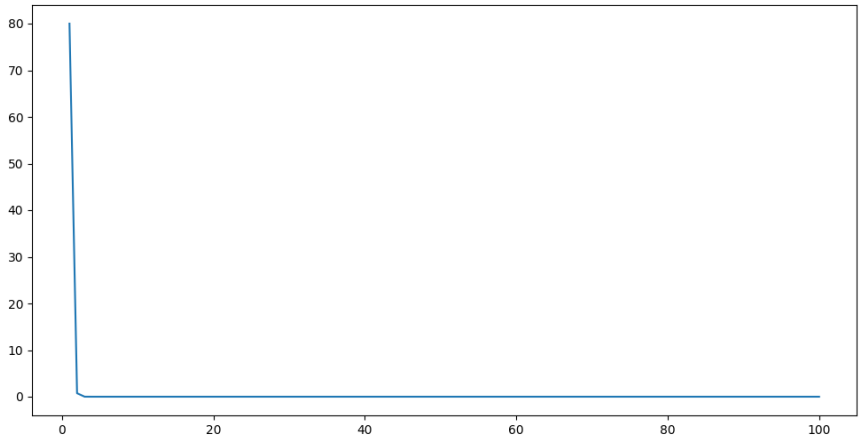
- We can see from the above that highest state count is around the opeining.
- First of all all runs will have to pass the opening, therefore area of approach towards opening will keep getting smaller and therefore we observe similar count just before opening.
- Next due to walls and stochastic actions, the agent does not go stright through opeing but might collide several time hovering around the opening, this raises state count further.
- The grey region in second half tells us the approach agent takes every time, go stright towards the goal.
- In the beginning there is higher statecount than goal because it is a corner, giving more probability of collision.

1.4 MAX NORM VARIATION OVER ITERATIONS

Theta vs Iterations y=0.99 Convergence around 595



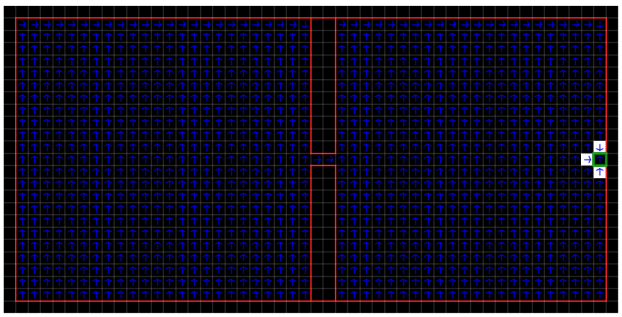
Theta vs Iterations y=0.01 Convergence around 3



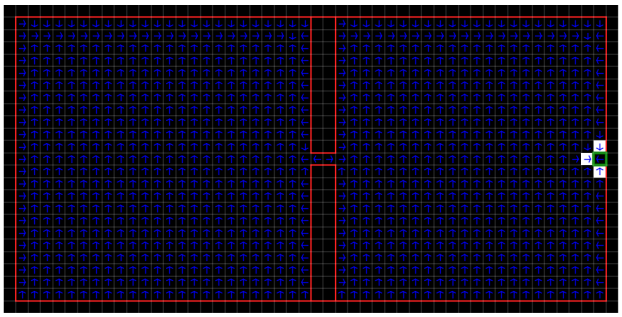
Observation :

- For y=0.99 the max norm decreases smoothly to 0
- For y=0.01 the max norm decreases suddenly to 0 in just 2 or 4 iterations

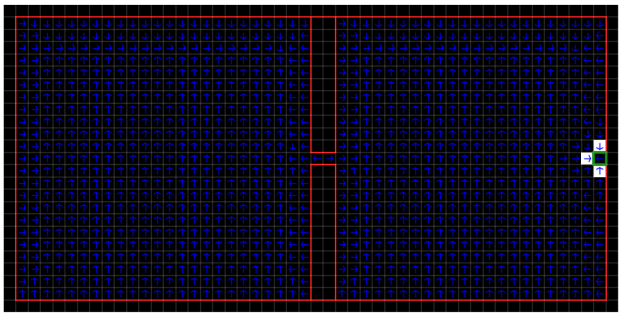
For y=0.01, we converged in 3 iterations that is max norm < 0.1



for t=1



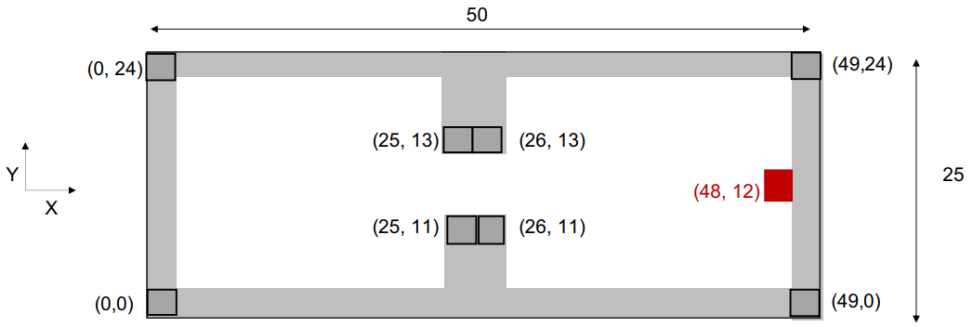
for t=2



for t=3
Please zoom in to view

2 Q-Learning

Initialization



- Each grid cell contains $\Longrightarrow$ reward(-1,0,100), qvalue list (north,east,south,west)
- q values initialised randomly for normal cells
- q values initialised 0 for goal cell and wall cell
- rewards as given

2.1 Q-Learner

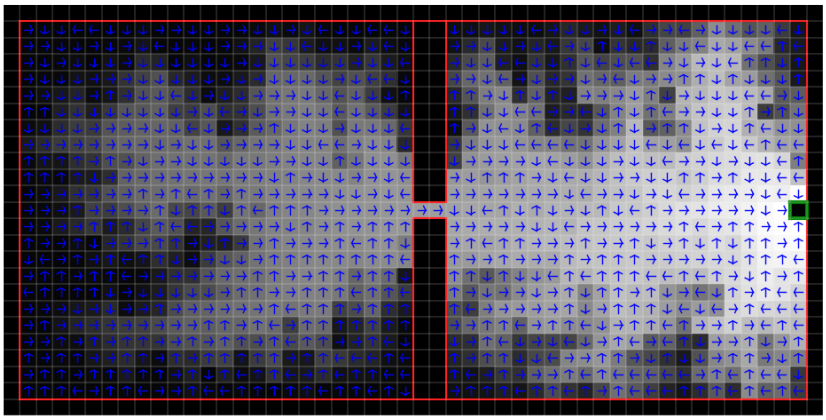
We will use q learning algorithm as follows

procedure q_Learner

1. Initialise grid world
2. for k in range(num_episodes):
3. pos = random initial position but not goal position
4. for i in range(episode length):
5. A = chooseAction using epsilon greedy
6. nextpos, reward = apply action (A, current pos)
7. Q(pos,A) $\longleftarrow Q(S,A) + \alpha * (reward + (\max_a$ Q(nextpos,a)) - Q(S,A))
8. if nextpos == goal then break

2.2 VISUALISING Q LEARNER

For y = 0.99 , alpha=0.25, episode length = 1000, num episode = 4000, epsilon=0.05
We get

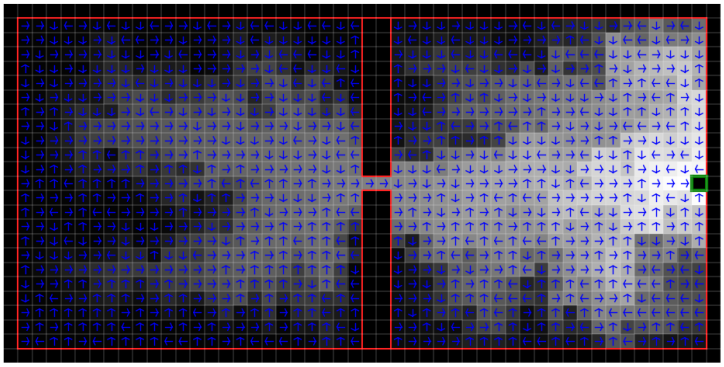


Observation :

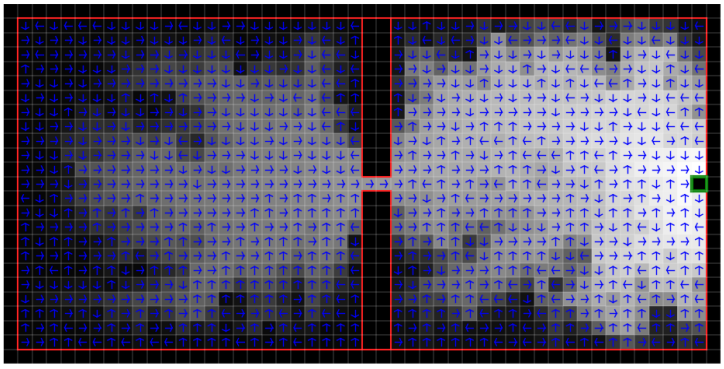
- As we can observe that the white cloud get brighter near the goal and is darker in first half.
- From animations i observed that once cloud reaches the opening, it starts spreading faster in first half.
- From the policy we can observe that once we are in the cloud, all policy actions if followed make the agent reach goal, though not in optimal fashion but it does reach.
- There is clear path to cross the opening
- Q learning is off policy, thus the danger of colliding with the wall does not propogate to adjacent cell due to max nature. Therefore the policy does not show any collision avoidance, though we can still say that there are almost no intention to collide with wall as well.

2.3 VARYING ϵ

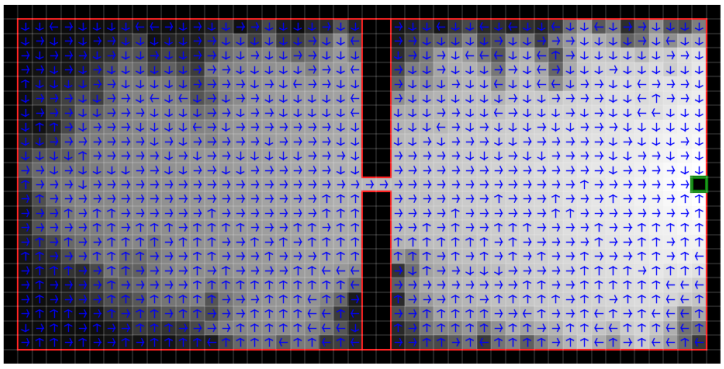
For epsilon = 0.005 we get



For epsilon = 0.05 we get



For epsilon = 0.5 we get

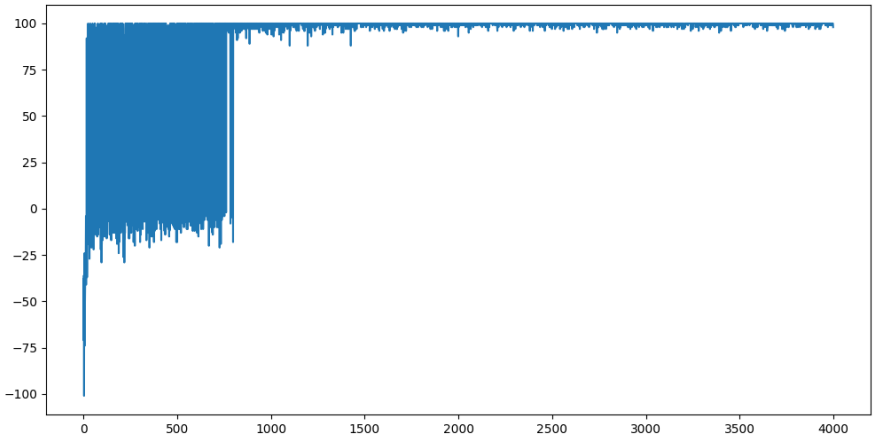


Observation :

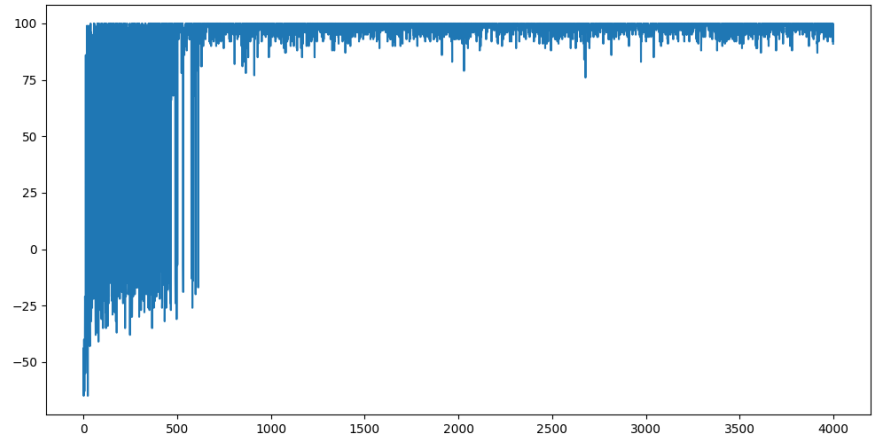
- We know epsilon is exploration factor, exploration is necessary in the beginning so that we can avoid the initial policy and try to search for higher rewards. Lower epsilon mean it will take lot more steps to reach goal state, more epsilon does our work easy and helps in distributing the message of higher reward to greater area.
- For epsilon = 0.005 we produce less epsiodes that reach goal state and thus the distant reward propogates less.
- For epsilon = 0.05 there is more propogation of goal reward as more episodes are able to reach goal state in the begining which leads to even more episodes to reach goal state as policy improves.
- For epsilon =0.5 we can see that policy almost looks like we solved the mdp. The actions are more uniform and leading towards the goal quickly.
- For epsilon=0.5 in the first half we can see that policy is trying to get agent to cross the clearing.
- Even in the first half, more the epsilon more refined policy we are getting.

2.4 Reward Accumulated Vs Episodes

For epsilon = 0.05 we get Reward vs episode number



For epsilon = 0.5 we get Reward vs episode number



Observation :

- We observe that saturation occurs faster in epsilon=0.5, as there are more episodes in the beginning that reach the goal, which in turn leads to more propogation of reward and thus in turn leading to more episodes reaching the goal.
- But after saturation in both on an average more reward is accumulated by epsilon 0.05 than 0.5, this happens because when we achieve a good policy, we don't need exploration, as exploration leads to more collision with wall, thus there are more fluctuations in 2nd graph
- In any range of epsiode numbers there is more fluctuation in second graph.

- In beginning there are lot of episodes with negative rewwards because of exploration and random policy

---

END
