**Final Project for Reinforcement learning**

**Student Name: Gajanan Wadekar**

**Project Name: POMDP Learning to Navigate with Uncertain Information**

POMDP are also called as Discrete Marcov Decision Process. So, here the MDP agent/learner does not have the observability. The agent do not have any idea about its location state or its orientation. The difference between POMDP and MDP is, the agent gives the probabilistic observation of what states it might be in instead of giving certain information about the current state.

I have designed a state space model for a grid of 15x25. With one goal state at [15, 25] and some obstacle. I changed number of obstacles and their location just to observe how agent behaves. The grid has some constraint such as when we take some action it has some specified probability based on the action to be taken. Say the action is to move forward or backward then the agent can execute the action and move to next state with probability of 0.8 and it will stay at the same state 20% of time. Likewise, it can turn left/right with 0.9 probability and it will stay at the same state with 0.1 probability.

The reward received by an agent on reaching the goal state is +100 whereas if the it gets into obstacle it will get -100. Also, in POMDP the agent will receive reward of -100 if the action taken takes agent out of grid and receives the reward of 0 otherwise.

Here in the problem of POMDP agent only has a knowledge of when it is hitting the wall or obstacle or goal. The only assumption made is we know the location of obstacles and goal.

The POMDP models the relationship between an agent and the environment.

POMDP -> (S, A, T, R, O, B, bo)

Where, S – set of states

A – Action set

O - Observation set

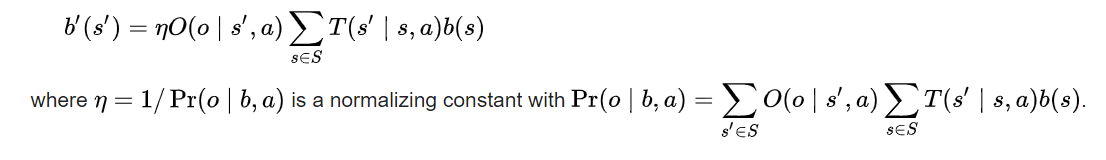
T – Transition probabilities

B – Observation probabilities

b0 - Initial or prior state probabilities.

In designing underlying MDP I have wrote several functions which will give me required data on giving some parameters. Like, computing the transition probability, getting reward at specifics state, checking if the state is valid or not, taking to next state on taking action a, computing observation probabilities, getting all the states that leads to specific states on taking specific action, updating the belief state as well.

As we know in POMDP, the agent is unaware about its current state, therefore, the agent must take a guess of probable state it could be in. That guess is called agents belief of being in certain state. Belief is just a probability distribution over state space. As the agent takes action and make a move to next state the agent will compute the beliefs for the next state it could be in. The belief is updated using below formula.



Here, the start state for the agent is [0,0] and orientation is 0 i.e. UP ( or North).

Therefore, B(s=start state) = 1.

Then to update the beliefs at next state, we need observation probabilities, transition probabilities and agent’s belief of current state.

Here in POMDP problem, I am assuming only one observation and that is if the agent is **not hitting a wall or not landing into obstacle**. To compute the observation probability we have to check if the agent is hitting a wall or landing into obstacle on taking action a at state s. I am assigning the probability of 1 if the agent is not hitting a wall or not landing into obstacle and if the agent is hitting a wall or landing into obstacle then the observation probability is 0.

To get the transition probability, I have to find out if the agent is taking particular action and actually moving to that state or staying at the same place. Also, if moved to next state I have to find out if the move was forward/backward or turning left/right.

As per the grids constraint the probability of moving forward/backward on taking action a is 0.8 and staying at same state is 0.2. the probability of taking turn left/right is 0.9 and staying at same state is 0.1.

To update the belief at next state, the numerator is,



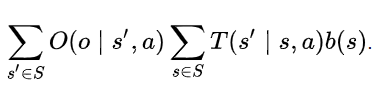
Where, O(o | s’, a) is the observation probability at state s’ (next state)

State space is set of states from where we can reach next state (s’) by taking action a.

b(s) is the belief of agent at state s.

We have to do the summation over multiplication of transition probability from state space and belief.

Then, the denominator is,

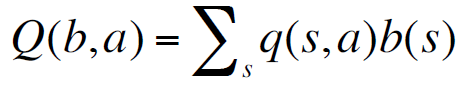


Now here while computing the denominator we have to get the observation probability for all the states in state space from where we can reach to next state.

After updating the beliefs, I need to update the Q value function. I am using linear Q learning to do it.



Qt(b,a) is computed as,



Q(s,a) is the q value function at state s and action a. b(s) is the agents belief at of being at state s.

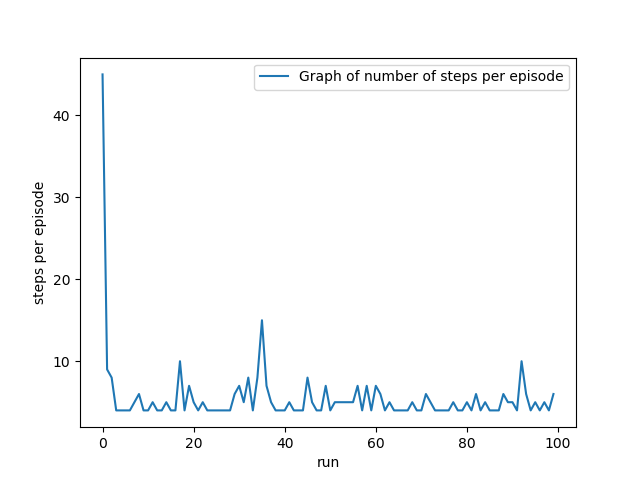
MAXc Qt (b’,c) is the maximum values of a value function at the current state with all the action choices. Its computed using above formula.

Then, in next value iteration, I am choosing epsilon greedy method to choose the action to take. I will be taking the action based on q value function if the random number is more than epsilon value.

**Results**

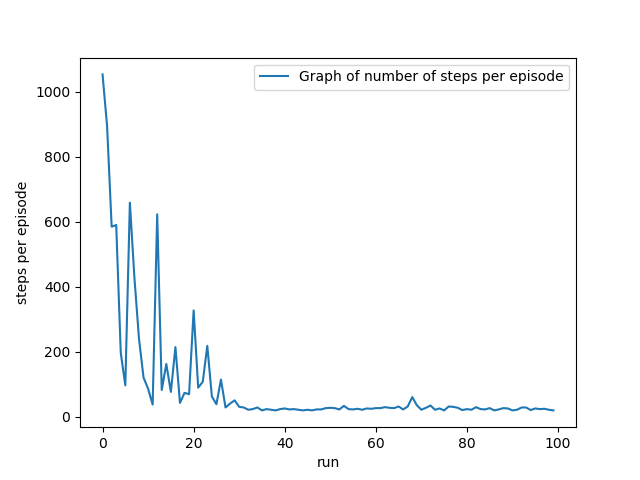
For result I am printing the number of steps an agent id taking to reach the goal. I am also plotting it against number of episode/run.

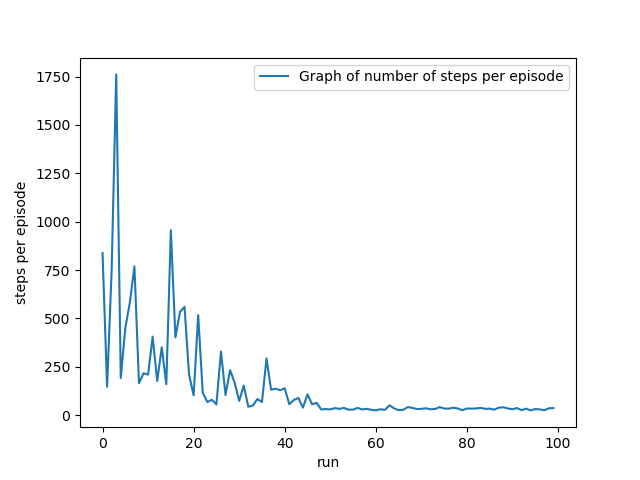
I had started the implementation with grid of size 3x3. With one obstacle and one goal state. The result is as below,

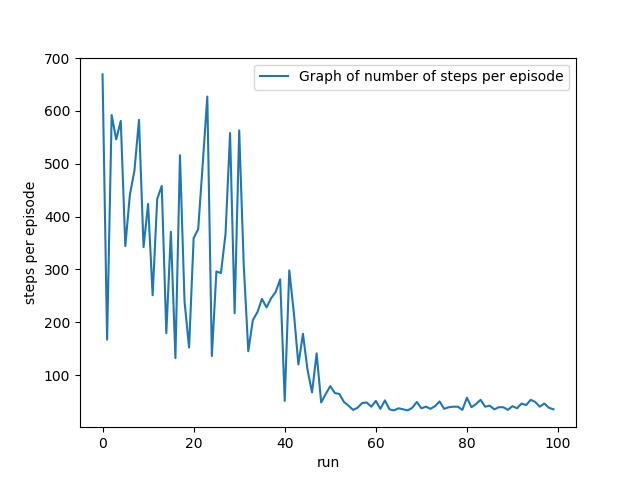


Then I experimented with different grid size and different number of obstacle.

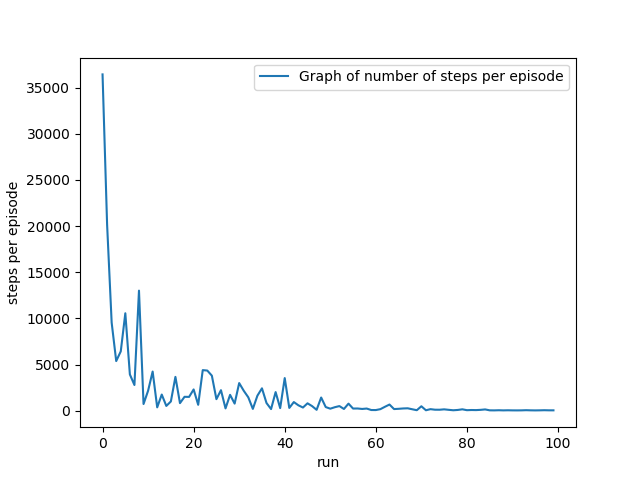
1. Grid Size: 10x10, 5 obstacles.



1. Grid size: 15x10, 8 obstacles
2. Grid Size: 15x10, 12 obstacles.



1. Grid size: 15x25, 12 obstacles



If there are more obstacles the agent will take a greater number of steps to reach the goal which will delay the convergence. If there are more than one goal the agent will always(most of the time) go to the one it finds first.

References:

1. Dr. Huber’s Presentations slides.
2. <https://en.wikipedia.org/wiki/Partially_observable_Markov_decision_process>
3. <https://www.cs.cmu.edu/~ggordon/780-fall07/lectures/POMDP_lecture.pdf>
4. <https://www.cs.mcgill.ca/~jpineau/talks/jpineau-dagstuhl13.pdf>
5. <http://cs.brown.edu/research/ai/pomdp/tutorial/pomdp-solving.html>