CS747 FILA: Assignment 2 Report

Markov Decision Process

The code submitted is complete and solves the necessary problem statement subparts

Design Decisions -

1. I maintain a 3D dict of s1,a,s2 i.e. state initial, action and state next with the value stores as [p,r] where p is the probability of transition and r is the associated reward ([0,0] wherever required as placeholders), thus I assume that the r provided are non-stochastic
2. Planner.py has 3 methods as different classes making use of the MDP object
3. LP uses the pulp repo to solve the Constraint Equations and uses machine precision to break
4. Policy Iteration makes uses of pulp library again to solve the Bellman equations (I tried NumPy but there were memory issues in large cases)
5. Value Iteration is pretty simple, where the value function is initialized to a random vector of 0’s and 1’s
6. I ensure that the Value function for end states is set to 0 by skipping update steps for them or setting the LP variables to -1 to be ignored

Observations -

1. In terms of running time lp and hpi are comparable in the planner case with vi being slower.
2. However, there is a change in that order the case of Task2 probably because s1,a,s2 is deterministic and lp finds optimum fastest in the case
3. LP has issues with precision which need to be dealt, apart from the default else needs to looping
4. lp while has better runtime in the Maze case it is still slower and probably an inverted index approach might be useful to speed up the code

MDP formulation for Maze

1. I consider only the valid states and consider a state_num to grid_pos list in the encoder in addition to an inverted list in the decoder
2. The actions are mapped as [“N”,” S”,” W”,”E”] to 0,1,2,3 action numbers
3. The reward from any state to a non-terminating state is set -1 to ensure minimum path length with gamma set to 1
4. The reward on entering the end state is set to N*M where the grid size is N*M
5. There no outgoing transitions from the end state