CS747 Assignment-2

Rohan Gupta, Roll Number: 180010048

October 2020

Contents

1 Introduction

Task 1 explains the three algorithms implemented in the assignment, namely, Value iteration, Howard's policy iteration, and Linear programming. Task 2 explains all the design assumptions and the values of tunable parameters taken while encoding a maze into an MDP and vice versa. Finally, observations list down all the reflections of the assignment.

2 Task 1

The task was to write three algorithms as stated below to find the optimum policy of any MDP. All the algorithms presented calculate the optimal policy, V^{π^*}, with a resolution of $exp(-12)$. A point to be noted is that no specific changes has been made for an episodic tasks. This is because the code assumes that there will be no transitions listed for terminal states. Such states are always initialised with 0 value function, 0 transition function and 0 reward. Hence, for terminal states, the value function will remain 'fixed' at 0, making it solvable.[1] The algorithm specific assumtions are described now.

[1] If task is episodic, guaranteed to have a unique solution, even if $\gamma = 1$, after we fix $V^{\pi}(s^T) = 0$ [slides(23)]

2.1 Value Iteration

The algorithm uses the Bellman optimality operator to find the optimal value and policy.As mentioned, the value function is initialised at 0 for all states. Along with it, the policy is initialised at 0 as well.

It takes under 30 seconds to find the optimal solution for the 100x100 grid given.

2.2 Howard's Policy iteration

The value function and policy are initialised at 0 as in Value iteration. Howard's policy iteration switches all improvable states greedily- it switches to the action which has the maximum Q value among the improvable actions.

2.3 Linear Programming

After solving the the linear program, the policy is obtained from it using the bellman optimality operator. The policy is then evaluated to find V^{π^*}, again for a higher resolution of $exp(-12)$. Note that as the transition function and the reward function are 0 for terminal states, the only equation fed to the linear programmer(for such states) is $V(s^T) \geq 0$. However, as lp minimises V(s), $V(s^T)$ always comes out to be 0. Hence, no extra constraint for episodic tasks are added here either.

3 Task 2

3.1 Encoder and decoder

The encoder encodes a maze in such a way that the reward is 0 for every valid transition, and 1 for reaching a terminal state. The discount used is $1 - \frac{0.01}{size(maze)}$ where size(maze) is the product of its width and height. The directions correspond to the actions as follows:

direction	action
W	0
E	1
N	2
S	3

Table 1: Directions corresponding to mdp actions

The mdptype is assumed to be 'continuous' due to possible loops. However, It does not really matter because the code does not depend on it.

4 Observations

The algorithm that performed the best in both task 1 and task 2 was 'Value Iteration'. As the constraints increase linear program becomes very slow as pulp needs considerable amount of time to add them. Howard's iterations policy seems slightly slower than than value iteration in both the tasks.