COMP3702 Artificial Intelligence (Semester 2, 2022)

Assignment 2: HEXBOT MDP – **Report Template**

Name: Eskil Pedersen

Student ID: 47613722

Student email: e.pedersen@uqconnect.edu.au

Note: Please edit the name, student ID number and student email to reflect your identity and **do not modify the design or the layout in the assignment template**, including changing the paging.

**Question 1** (Complete your full answer to Question 1 on the remainder of page 1)

Question 1. MDP problem definition (15 marks)

a) Define the State space, Action space, Transition function, and Reward function components of the HexBox MDP as well as where these are represented in your code. (10 marks)

State space: Consists of all possible legal configurations of the robot positions and orientation as well as each widget’s centre position and orientation. The state space is defined under

self.states = self.bfs()

def *bfs*(self):

states = []

states.append(self.environment.get\_init\_state())

frontier = [self.environment.get\_init\_state()]

*while* len(frontier) > 0:

current\_state = frontier.pop()

*for* action *in* ROBOT\_ACTIONS:

cost, new\_state = self.environment.apply\_dynamics(

current\_state, action)

*if* new\_state not in states and new\_state not in frontier:

frontier.append(new\_state)

*if* current\_state not in states:

states.append(current\_state)

*return* states

Action space: Consists of the four possible actions [forwards, backwards, spin left and spin right].

Transition function: Takes in an action, a state, and a next state, and gives the probability of ending up in the next state when starting from state and preforming the given action. , where S is state, A is the given action, S’ is the resulting state, and P is the probability of that happening. When calling self.stoch\_action(a) I get different probabilities for the different combinations of actions that can occur when performing action a. I can then use the function apply\_dynamics(state, action) to get the resulting next state when performing the original action a.

Reward function: Maps a state to a real value, that indicate how desirable it is for the agent to occupy that state. It is implemented in lines 140 to 149 in my code.

b) Describe the purpose of a discount factor in MDPs. (2.5 marks)

When solving an MDP, we want to find the highest sum of rewards. To make sure this sum doesn’t go to infinity, or minus infinity, a solution to this problem is to discount the future rewards, such that the mathematical series converge, and the infinite series is changed to finite series.

c) State what the following dimensions of complexity are of your agent in the HexBot MDP. (See https://artint.info/html/ArtInt\_12.html for definitions) (2.5 marks)

Planning Horizon: Indefinite.

Sensing Uncertainty: Fully observable.

Effect Uncertainty: Stochastic.

Computational Limits: Perfect rationally.

Learning: Knowledge is given.

**Question 2** (Complete your full answer to Question 2 on page 2)

Question 2. Comparison of algorithms and optimisations

(15 marks)

This question requires a comparison of your implementation of value iteration (VI) and policy iteration (PI). If you did not implement PI, you may receive partial marks for this question by providing insightful relevant comments on your implementation of VI. For example, if you tried standard VI and asynchronous VI, you may compare these two approaches for partial marks.

a) Describe your implementations of value iteration and policy iteration in one sentence each. Include details such as whether you used asynchronous updates, and how you handled policy evaluation in PI. (2 marks)

The value iteration is using synchronous updates, and therefore does not use asynchronous updates.

The policy evaluation in PI is using linear algebra to update the rewards for each state.

b)

Pick three representative testcases and compare the performance of Value Iteration (VI) and Policy Iteration (PI) according to the following measures. Please include the numerical values of your experiments in your comparisons.

• Time to converge to the solution. (3 marks)

• Number of iterations to converge to the solution. (3 marks)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Testcase | VI Time | VI Iterations | PI Time | PI Iterations |
| 1 | 6.6 s | 54 | 6.9 s | 10 |
| 2 | 7.3 s | 56 | 8.5 s | 17 |
| 3 | 7.2 s | 56 | 7.9 s | 17 |
| 6 |  |  | 197.2 s | 24 |

c) Comment on the difference between the numbers you found for VI and PI. List any reasons why the differences either make sense, or do not make sense. (7 marks)

As we can see, do PI have fewer iterations than VI in all testcases, however, PI is slightly slower than VI in all of the testcases.

It makes sense that PI have fewer iterations than VI. T

However, it does not really make sense that VI is faster than PI. This can be explained by one of, or a combination of the reasons below:

* The initialise of PI takes longer time than the initialise of VI. There are nothing in the initialise of VI, all of the required variables are declared in \_init\_. But in pi\_initialise we have to compute both the t-model and r-model, which loops thorugh all combinations of legal states and robot-actions. From my test, it seems like pi\_initialise takes around 3 seconds.
* The policy update in PI is as slow as a value iteration pass.

**Question 3** (Complete your full answer to Question 3 on page 3)

Question 3. “Risky Business”: Optimal policy variation based on probabilities & costs (10 marks)

One consideration in the solution of a Markov Decision Process (i.e. the optimal policy) is the trade off between a risky higher reward vs a lower risk lower reward, which depends on the probabilities of non-deterministic dynamics of the environment and the rewards associated with certain states and actions.

Consider testcase ex6.txt, which includes a risky (but lower cost) path through the top half of the grid, and a less risky (but higher cost) path through the bottom half of the grid. Explore how the policy of the agent changes with hazard penalty and transition probabilities.

If you did not implement PI, you may change the solver type to VI in order to answer this question.

a) How do you expect the optimal path to change as the hazard penalty and transition probabilities change? Use facts about the algorithms to justify why you expect these changes to have such effects.(5 marks)

b) Picking three values for hazard penalty, and three sets of values for the transition probabilities, explore how the optimal policy changes over the 9 combinations of these factors. Do the experimental results align with what you thought should happen? If not, why? (5 marks)