National University of Singapore
School of Computing

Semester 1, AY2021-22

CS4246/CS5446 AI Planning and Decision Making

Issued: 10 Sep 2021 Due: 27 Sep 2021@23:59

Assignment 2

General Guidelines

Please complete the Assignment in the **same 2-person TEAM** as for past assignment(s). **ONE person from each team should submit ALL the answers (for both programming and homework problems)**. Grading will be done team-wise.

On collaboration

- You are encouraged to discuss solution ideas. However, each team *must write up the solutions independently*. It is considered as plagiarism if the solution write-up is highly similar to other teams's write-ups or to other sources.

Homework Problems Submission

- Submission will take place through **LumiNUS Quiz**.
- No late submissions for the homework assignment solutions are allowed.

Programming Problems Submission

- **ONE person from each team should submit the solutions**. Grading will be done team-wise.
- **IMPORTANT!** Please write the following on the top of your solution files for EACH problem in the assignment:
  - *Name*, *metric number* and *Tutorial Group* of **all team members** as they appear on LumiNUS
  - Collaborators (write **None** if no collaborators)
  - Sources, if any, or if you obtained the solutions through research, e.g. through the web.
- **Upload the code of your programming assignment to the aiVLE evaluation server**, by following the instructions here. Please keep a copy of your submission zip file for safety.
- Please name your files, using ONE of the members' information according to this format: `A1_<Member1-TutorialGroup>-<Member1-name>-Task1` and `A1_<Member1-TutorialGroup>-<Member1-name>-Task2`.
  E.g., `A1_TG6-xiaomei.tan-Task1.zip` and `A1_TG6-xiaomei.tan-Task2.zip`
- A late score penalty of 20% will be incurred for late programming assignment solutions submissions.

1 Homework Assignment (LumiNUS Quiz)

[Preparation]: *For this problem, while the numbers may be worked out manually, you are encouraged to use one of the academic or free trial versions of computer programs for decision modeling, e.g., Bayesfusion GeNIE modeler or SMILE engine, Netica, Syncopian DPL 9, TreeAge, etc., to manipulate the influence diagrams and the decision trees.*

1.1 Homework Problem 1: Decision Theory

[10 marks] The CEO of Orange Computer Inc., Tim Bean is facing a dilemma: The pandemic has caused serious disruptions in the main supply chain of its hit digital watch, iTick, while the demand is still so strong that the current production capacity is unable to keep up. A decision has to be made between two alternatives:

i. Keep to the current production unit

ii. Set up a new production unit

Without any increase in the production, production will definitely fall below demand in 3 months. However, since setting up a new production unit would require a complete rearrangement of work with the current production line, there is a 0.25 probability that production will fall below demand immediately. If the new production unit is successful, production is likely to be kept up for 1 year, at that time the product will be replaced with a new version.

a) *[2 marks]* Draw an influence diagram for this simple decision problem. Show the structure and the numerical parameters.

b) *[2 marks]* Draw a decision tree for this simple decision problem. Show the structure and the numerical parameters.

c) *[2 marks]* Let U(x) denote the company's utility function, taking into account its risk attitude and the potential revenue, where x is the number of months until production falls below demand. Assume that U(12)=1.0 and U(0)=0.0, how low can the company's utility for falling behind demand in 3 months be and still have the new production unit option be preferred?

For the rest of the problem, assume that U(3)=0.70. Now the main supplier for iTick's components, The Sparkling Chips, offers to form a joint venture with Orange to speed up production of the iTick. CEO Bean has to consider the new option (in addition to the 2 options above): If the new joint venture succeeds, there is a 0.15 chance that the production will fall below demand in 3 months' time, or production will be kept up for at least 1 year. If the joint venture fails, however, Orange can still consider the option of adopting the original plan (without working with the Sparkling Chips) for a new production unit, which now has 0.4 chance of falling below demand immediately, or otherwise able to keep up production for 1 year. There is a probability of 0.3 that the joint venture will succeed.

d) *[4 marks]* What is the best option for CEO Bean? Why?

1.2 Homework Problem 2: Markov Decision Process

[10 marks]

a) *[4 marks]* Can any finite search problem be translated exactly into a Markov decision problem such that an optimal solution of the latter is also an optimal solution of the former?

If so, explain precisely how to translate the problem and how to translate the solution back; if not, explain precisely why not (i.e., give a counterexample).

b) *[6 marks]* Sometimes MDPs are formulated with:

i a reward function $R(s)$ that depends only on the current state s, or

ii a reward function $R(s, a)$ that also depends on the action a taken in the state, or

iii a reward function $R(s, a, s')$ that also depends on the action a and the outcome state s'.

Write the Bellman equations for these formulations.

2 Programming Assignment (aiVLE submission)

In the first assignment, we learnt to solve deterministic planning problems with PDDL solvers. In this Assignment, we will learn to solve planning problems in non-deterministic and stochastic environments with two separate algorithms, Value Iteration and Monte Carlo Tree Search.

Installation Instructions
We assume that you have already set up the programming environment for the assignments.

Programming Assignment Submission Instructions
Please follow the instructions listed here. Task specific submission instructions are as follows:

a. For Task 1: You have been provided with a vi_agent_template.zip file in the correct submission format. Complete the functions marked with "FILL ME" in `__init__.py`. Your submission zip file should have the exact same structure as the zip file you received.

b. For Task 2: Modify the `__init__.py` in the mcts_agent_template.zip by completing the functions with "FILL ME" and make a separate submission.

Note :

1. Please remember to set the variable SUBMISSION in `__init__.py` to False when testing locally, and to True before making a submission.

2. Please do not print anything to the console, as it might interfere with the grading script.

Getting started
We will be using the same gym_grid_environment (https://github.com/cs4246/gym-grid-driving) to simulate the solutions. All dependencies have been installed in the docker image "cs4246/base" which you have already downloaded for Assignment 1.

By now, you should be familiar with the gym_grid_environment. Go through the IPython Notebook file on Google Colab here, if you have already not done so in the previous programming assignment.

You are now ready to begin solving the two tasks which are listed below!

2.1 Programming problem 1: Value Iteration

[10 marks] In the crossing the road task in Assignment 1, we assumed all the cars move with a constant speed of -1. Most of the times, this assumption does not hold. There is inherent noise in the car speeds due to reasons like bumpy roads, uneven driving, etc. This non-deterministic behaviour is not captured in the PDDL format.

For problems that are not very large, planning algorithms like Value Iteration can be used to handle non-deterministic behaviours. In this task, we model the problem as a Markov Decision Process and use Value Iteration algorithm to find the optimal policy.

We have provided the script which models the problem as an MDP. Every state in this MDP is represented as a tuple of features: $\{\text{Agent}_x, \text{Agent}_y, \text{Car}_x^1, \text{Car}_y^1, \cdots, \text{Car}_x^N, \text{Car}_y^N, \text{isTerminal}\}$, where:

- $\text{Agent}_x, \text{Agent}_y$ denote the x and y coordinates agent.
- $\text{Car}_x^N, \text{Car}_y^N$ denote the x and y coordinates of Car^N.
- isTerminal is a boolean variable denoting whether the state is terminal.

Your task is to fill up the function that implements the Value Iteration algorithm which is marked with "FILL ME".

You can test your code with the test configurations given in the script by running `python __init__.py N`, where N is an int value between 0 to 5, on the docker (using the `docker run ...` command).

HINT : You can use Matrix multiplication to optimise the code, but refrain from using the function `np.matmul()/np.multiply()/np.dot()` as these functions interfere with the evaluation script. Instead you can use function `matmul()` defined in the same script.

2.2 Programming problem 2: Monte Carlo Tree Search

[20 marks] Unfortunately, we moved to a bigger parking lot and Value Iteration is not feasible as it does not scale well with large state spaces. However, we can use Monte Carlo Tree Search (MCTS) to handle such cases.

We have provided a separate script which solves the problem with MCTS, Your task is to complete the script by filling up 2 code snippets inside the functions marked with "FILL ME".

The functions required to be filled up are:

a) backpropagate() : This function should implement the backpropation step of MCTS.

b) chooseBestActionNode() : Populate the list bestNodes with all children having maximum value. The value of the i^{th} child node can be calculated with the formula:

$$\frac{v_i}{n_i} + c\sqrt{\frac{\log N}{n_i}}$$

where,

- v_i : sum of returns from the i^{th} child node
- n_i : Number of visits of the i^{th} child node
- N : Number of visits of the current node
- c : The exploration constant. We set it to be 1.

For more details on MCTS, it might be helpful to look here. You are also encouraged to look through the rest of the code to see how the entire MCTS algorithm is implemented.

You can test your code with the test configurations given in the script by running `python __init__.py N`, where N is an int value between 0 to 5, on the docker (using the `docker run ...` command).