National University of Singapore
School of Computing

Semester 1, AY2023-24

CS4246/CS5446 AI Planning and Decision Making

Due: 17 Nov 2023 (Soft); 24 Nov 2023@23:59 (Hard)

Assignment 3

Instructions

- This assignment requires 2 submissions.
  - Submit the PDF file containing your solutions to this assignment on Canvas.
  - You can use this Overleaf project to write your solutions.
  - Submit the ZIP file containing your code to this separate assignment on Canvas.
- You have only one attempt on Canvas.
- You are required to specify your group details in your attempt (group number can be found on Canvas) by updating the `team.tex` file with the relevant details.
- **Total marks: 10; Weightage 10% of final marks**
- On collaboration:
  - The goal of the assignment is to understand and apply the concepts in the class.
  - You may discuss the assignment with other groups via the discussion forum.
  - It is OK for the solution ideas to arise out of such discussions. However, it is considered plagiarism if the solution submitted is highly similar to other submissions or to other sources.
- Citing help and reference
  - At the end of the assignment, clearly cite the sources you referred to when arriving at the answer (Books, External notes, Generative AI tools, etc.,).

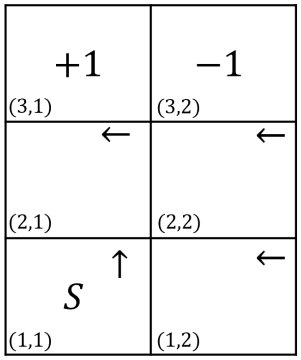
Team members

Group number:
Member 1 details:
- Name:
- NUSNet id:
- Student number:

Member 2 details:
- Name:
- NUSNet id:
- Student number:

1 Reinforcement Learning

[3 marks] Consider the environment shown in Figure 1 (left). The agent can execute actions to move in the four cardinal directions. The policy the agent follows is indicated using the arrows in the states. Note that the reward for the states other than $(3, 1)$ and $(3, 2)$ is zero. The agent runs the policy to obtain the trails as shown in Figure 1 (right).



Trial 1: (1,1) → (2,1) → (3,1)
Trial 2: (1,1) → (2,1) → (2,2) → (2,1) → (3, 1)
Trial 3: (1,1) → (2,1) → (2,2) → (1,2) → (2,2) → (3,2)
Trial 4: (1,1) → (1,2) → (2,2) → (3,2)

Figure 1: Environment and Trials

1. (1 mark) Compute the transition function for the transitions you can observe from the trials.
2. (2 marks) Describe the problem with ADP based on your observation from computing the transition function from the trials in the above.

2 Game Theory

[2 marks] Both Row player ($P1$) and Column player ($P2$) have three options A, B, C to choose. The payoff matrix for the combination of choices they make is as shown below

		P2		
		A	B	C
P1	A	-5, -5	0, 5	5, 10
	B	5, 0	-10, -10	-2.5, -7.5
	C	10, 5	7.5, -2.5	-15, -15

1. (2 mark) Compute the pure strategy Nash equilibria and write the strategy profile(s) for the game.

3 Programming assignment

[5 marks]

The programming assignment requires you to:

1. (3 marks) Implement an RL algorithm of your choice to solve the elevator problem. Please follow the implementation instructions in 3.2 for more details.
2. (2 marks) Briefly state and explain the algorithm you chose, hyperparams used (if any) and your observation/learning as you implemented the solution. Keep your description succinct. Do not write more than 1 page for this part! You are not required to give the algorithm pseudocode or detailed description.

3.1 Description of the algorithm

(2 marks)

Write your answer here

3.2 Implementation instructions

In this programming assignment, we will continue to use a discretized version of the elevator domain from the International Planning Competition, 2023.

Problem setup

The Elevator domain models evening rush hours when people from different floors in a building want to go down to the ground floor using elevators. Potential passengers arrive at a floor according to the Poisson process, with specific rates that can be different across different floors. An elevator can move upwards to pick up passengers; once it opens its door to let people in, it can only move down towards the ground floor, though the elevator can stop at intermediate floors to pick up more passengers.

Per each person in an elevator, there is a small penalty in rewards. Additionally, there is a penalty for each person waiting for an elevator. When an elevator delivers a passenger to their destination, a large positive reward is given per person. So, a good policy should be able to minimize the penalties while maximizing the positive rewards by delivering many people so that they can go back home!

The problem setup for this assignment is exactly the same as in Assignment 2, i.e., the states, actions, and reward function remain the same. Please refer to the Assignment 2 handout for details of the environment.

Objective

(3 marks) Your aim is to deliver as many people from other floors to the ground floor using a policy computed using *any* discrete RL algorithm of your choice. You are required to define the model you will use for training and update the `agent.py` (see Handout and task section for more details). We have created a template Google Collab file for you to train your agent.

GRADING:

- We will evaluate your implementation (the evaluator will use the model you upload) using three private test cases, and take the average of the score obtained.
- To obtain full 3 marks for this part, you need to attain an average score of 600 and above.

Handout and task

- Use the `assignment3_handout.zip` file on Canvas to attempt the programming assignment.
- The `README.md` file has additional details about the environment.
- Your task is to complete the `YourModel` class that will be the model of a deep RL algorithm in the `agent.py` file. (It is marked with `TODO: Define your model here:`)

Submission

- Please write both the team member names and student numbers (Axxxxxxx) at the top of the agent file (as comments) to help us identify your submissions.
- **Zip the `agent.py` and the `model.pt` files and submit this zip file.** Ensure the structure of your submission is as follows. If not, you will encounter runtime error on AiRENA.

```
agent.zip
    |---- agent.py
    |---- model.pt
```

- Submissions for the programming part will happen on the AiRENA portal for grading.
- You are **also required** to submit the **zip file containing the `agent.py`, the `model.pt`, and the `.ipynb`** (or equivalent file used to train the model) on Canvas for verification and record-keeping/backup purposes.
- Please refer to the user guide (`AiRENA Guide for CSxx46.pdf` file on Canvas) for submission instructions.
  - Please ensure you follow the instructions provided to create your account.
  - If the account details don't match your information on Canvas, such submissions will not be graded.
- You need to use the following token to join the course (if you haven't done so yet!): **b68f4d45-a95f-449d-93f7-2e1f0edf154e**

Support

- You can use the Google collab from Assignment 2 to understand the elevator domain.
- We have prepared a Google collab template for you to train the agent.
- If you have questions about the assignment, please ask on the discussion forums.

References

[Optional] Use this file to declare your reference sources.