COMP0037 2020/21 Coursework 1

COMP0037 Teaching Team

February 2021

Overview

- Coursework 01 Release Data: Friday, 5th February 2021
- **Assignment Due Date: Monday, 22nd February 12:00 (Midday UK time) 2021**
- Weighting: 40% of the module total
- Final Submission Format: Each group submits *three* things:
    1. A zip file (which contains the source code implemented to tackle this coursework). The name of the zip file must be of the form `COMP0037_CW1_GROUP_g.zip`, where `g` is the letter code of your group.
    2. A report in PDF format. It will be named `COMP0037_CW1_GROUP_g.pdf`.
    3. A video which describes your answers. You may use a variety of video formats, but the video must be playable by VLC. It will be named `COMP0037_CW1_GROUP_g.[ext]`.

We are aware that you can find implementations of almost all of these algorithms on the web. However, we strongly recommend you attempt to implement the code yourselves, because that is the most effective way to learn.

We will give credit for code you write yourselves. If you use code from a third party source, you *must* credit where it comes from. If we identify uncredited code, we will treat it as plagiarism.

The total marks for each question are written in the form [X marks]. In some cases, the marks will be written as [$m + n(c)$ marks], where $n(c)$ are the marks from writing your own code and m are the marks for all the other part of the question. The balance varies across the individual sub-parts of questions. However, about approximately 67% of the total coursework marks are from your analysis and 33% from your coding of the algorithms.

The final mark will be computed by summing all the marks awarded on individual questions together and dividing by the mark total.

Neither the video nor the code will be independently marked but must be provided. Both will be checked.

Note that the bulk of the marks are *not* be on the coding, but on your analysis and interpretation of the results.

Use Case

All the questions in this part of the module are grounded on an example of a robot which automatically cleans an airport arrivals area. Airports are generally very busy places and require constant cleaning.

Figure 1 shows an overview of the environment the robot operates in. Passengers arrive at the top of the map, collect their luggage from the baggage reclaim areas (R1–R4) and pass through customs. Beyond there are various food vendors and a waiting area. There are four toilet blocks (T1–T4). T1 and T4 are before customs, and T2 and T3 are after customs. Four charging stations are located throughout the terminal (C1–C4). To be able to move between the customs and open area, the robot has a secret robot door that only it can access. Rubbish will be deposited differently in different areas.

Questions

1. The cleaning robot is powered by battery which becomes depleted during regular operation. In the airport, there are four designated charging locations C1, C2, C3 and C4. Each charging location has a different charging rate resulting in variable charging time. The electrical engineers working on the problem have been able to model the charging rate as a Gaussian distribution. The mean varies at each location, but the variance is 1. The robot

CONTINUED

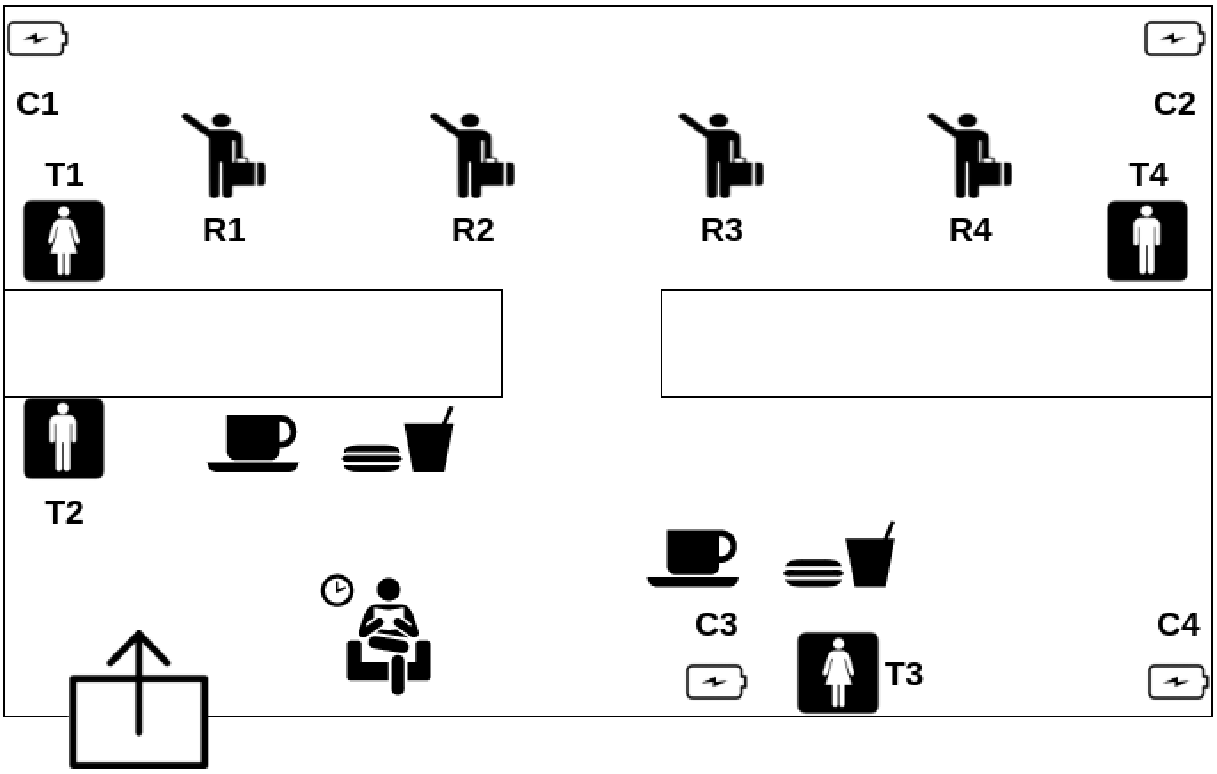

Figure 1: Overview of the arrivals area scenario. This is very loosely based on Heathrow Terminal 4. (Please note this figure does not mark the customs area or the robot access tunnel. These details are shown in Figure 2.)

has been tasked with learning the best charging location but can only do so with observed charging rates (does not know the modelling information). Assume the robot starts equal distance to all locations, only the charging rate should be considered. Charging rates directly determine charging times. Higher rates result in shorter times and lower rates result in higher times.

Charging Location	Mean Charging Rates (Amps)
1	4
2	8
3	6
4	12

Table 1: Charging information for locations.

a. Briefly describe how this problem can be expressed as a bandit and specify the number of arms. Develop a multi-arm bandit simulation to simulate the charging rates by the different stations. Generate violin plots to visualise the locations using the data generated from your simulator.

[10+5(c) marks]

b. Define regret, and briefly explain why it is used to assess the performance of a solution for these of problems. Although we can use regret for this problem, why might it not be an available tool for other problems?

[10 marks]

c. Briefly describe and implement the ϵ-greedy strategy for this bandit problem with $\epsilon = 0.1$. Present results for $2,000$ Monte Carlo runs averaged over 1000 timesteps.

[15+10(c) marks]

d. A number of ways to improve the ϵ-greedy approach have been proposed. Briefly describe one approach, implement it, and demonstrate its performance.

[15+5(c) marks]

e. Briefly describe Upper-Confidence-Bound (UCB) Action Selection. What is the basic differentiating principle when compared to other strategies? Implement and present the results on the scenario when the degree of exploration is 1.

[20+5(c) marks]

f. Investigate the performance of the UCB for several different choices of the degree of exploration. What trend do you observe, and what do you think is the value which gives the best performance. How do you think the degree of exploration relates to the variance of the charging rates?

[10 marks]

[Total 105 marks]

2. Now that the robot has identified the charging station which will be used, the engineers now begin to explore how the robot can reconcile the problems of keeping itself charged while undertaking its mission to keep the arrivals area clean. The robot's battery can be in one of three states: *high*, *medium* or *low*. The robot has three activities it can undertake. These are *search* (look for rubbish), *wait* (wait for rubbish to accumulate), or *recharge* (replenish its battery).

   The problem is modelled as an episodic Markov Decision Problem (MDP).

   At the start of each episode, the robot is at the charging station and its battery is fully charged. The robot carries out a series of activities, one per time step. The episode terminates when the robot runs out of power and its battery becomes flat.

   The battery state transition depends upon the activity the robot undertakes:

   - **Search:** If the battery level is high, the probability that the battery will remain high is $1 - \alpha$, the probability that it becomes medium is $\frac{2\alpha}{3}$ and the probability that it becomes low is $\frac{\alpha}{3}$. If the battery level is medium, there is a probability $1 - \beta$ that it remains at medium, and a probability β that it becomes low. Finally, if the robot battery level is low, there is a probability $1 - \gamma$ that it remains in a low state, and a probability γ that the battery becomes flat. If the battery becomes flat, the robot has to be manually taken to a charging station and the episode ends.
   - **Wait:** The battery state remains the same with probability 1.
   - **Recharge:** If the robot performs a charge, the battery will change its state from low to medium with probability δ and stay at the same level with probability $1 - \delta$ (does not charge) , and from medium to high with probability δ and stay at the same level with probability $1 - \delta$.

   At each time step, the robot receives a reward signal. The reward signal depends upon the robot's current activity. When searching for rubbish, the reward is r_{search} per time step. While waiting for rubbish to accumulate the reward is r_{wait}. The rewards for recharging is $r_{recharge}$. If the robot runs out of charge, a human has to push it to a charging station. The reward is $r_{discharged}$.

   To help you, we provide a set of Python files in the accompanying file `Q2.zip`. These files are based off of the code provided in the workbook for laboratory 2.

a. Define what a Finite Markov Decision Process (FMDP) is, and describe the main components of it.

[5 marks]

b. Define what s, r and a are for this problem, and construct the table of values for the state transition probability $p(s'|s, a)$ and the reward function $r(s, a, s')$. Your answer should incorporate the fact that the task is episodic.

[15 marks]

The parameters needed to fully define the scenario are given in Table 2.

Parameter	Value	Parameter	Value
α	0.4	β	0.1
γ	0.1	δ	0.1
r_{search}	10	r_{wait}	5
$r_{discharge}$	-10	$r_{recharge}$	0

Table 2: The parameter values.

c. Implement the FMDP for this problem by extending the Python code which has been provided (which is based on the scenario you saw in Lab 2). Ensure your code is parameterised so that arbitrary values for the parameters can be used. Present and explain the source code of your implementation. Propose a method to check if your implementation is correct, and present experimental results to verify its correctness. Your experiments should use the values provided in Table 2.

[25+15(c) marks]

d. Explain what *policy evaluation* and how it is calculated. Explain how can it be used to help the designer propose policies to control how the robot behaves.

[10 marks]

The engineer proposes to use random policies. Each policy has the form $\pi(a|s)$. Two policies (π_1 and π_2) have been identified, and for the non-terminal state they are shown in Table 3. In addition to the random policy, a greedy policy π_g is learned using the ϵ-greedy algorithm.

	Action		
State	Wait	Search	Recharge
Low	1/3	1/3	1/3
Medium	1/3	1/3	1/3
High	1/3	1/3	1/3

π_1: Uniformly Random Policy

	Action		
State	Wait	Search	Recharge
Low	0.8	0.1	0.1
Medium	0.2	0.8	0
High	0.2	0.8	0

π_2: Tuned Random Policy

Table 3: The two policies.

e. Implement the policy evaluation algorithm. Present and explain your code. Apply it to your implementation of the FMDP. Show that the value functions are

$$v_{\pi_2}(s) \approx \begin{bmatrix} 386.2 & 386.2 & 273.8 & 0 \end{bmatrix}.$$

Similarly show the policy evaluation functions for π_1 and π_g. Rank the different policies and identify which, if any, is the best. Explain your reasoning.

[20+25(c) marks]

f. A single day consists of fifty time steps, after which the episode resets. The state transition probabilities and reward functions are the same for all timesteps. Propose a way to expand the state space and the FMDP to incorporate this time dependency. You do not have to implement your solution.

[10 marks]

[Total 125 marks]

3. The cleaning robot has been tasked with finding locations to clean (food, toilets) while avoiding high traffic areas (reclaim, waiting area). Figure 2 depicts a discretised map of the airport, where the cleaning locations and high traffic area locations are given. Entering a cleaning location or high traffic area ends the episode, and the robot will be randomly placed in a valid empty position on the grid at the start of the next episode. To move on the map, the robot can rotate clockwise and counterclockwise, which changes the heading but remains at the same position. It can also move forward (in the direction of heading). Barriers and customs area are no-go zones. Attempting to enter these zones will result in no change in position. The robot has access to a trapdoors at location $x = 4$, $y = 8$ and $x = 2$, $y = 8$. To use it, the robot must be at the trapdoor position, heading east. With a forward action, the next position will be the other trapdoor heading west.

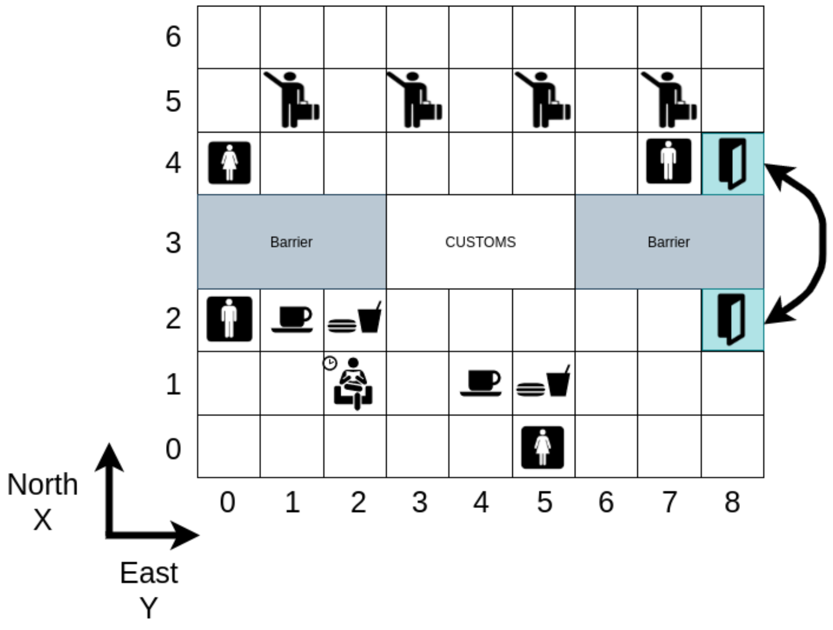

Figure 2: Discretised map of Figure 1. This figure is drawn to scale. Therefore, you may consult it to identify coordinates of different areas in the map.

The engineers have also developed a reward function for the robot to use. If the robot reaches a cleaning location it will receive a $+10$ reward whereas entering a high traffic area will result in a -10 reward. Any movement (turning or moving forward) costs the robot -1 reward.

The engineers initially assume that there are no substantial disturbances. Therefore, there are no nature actions, and the behaviour of the robot is completely deterministic.

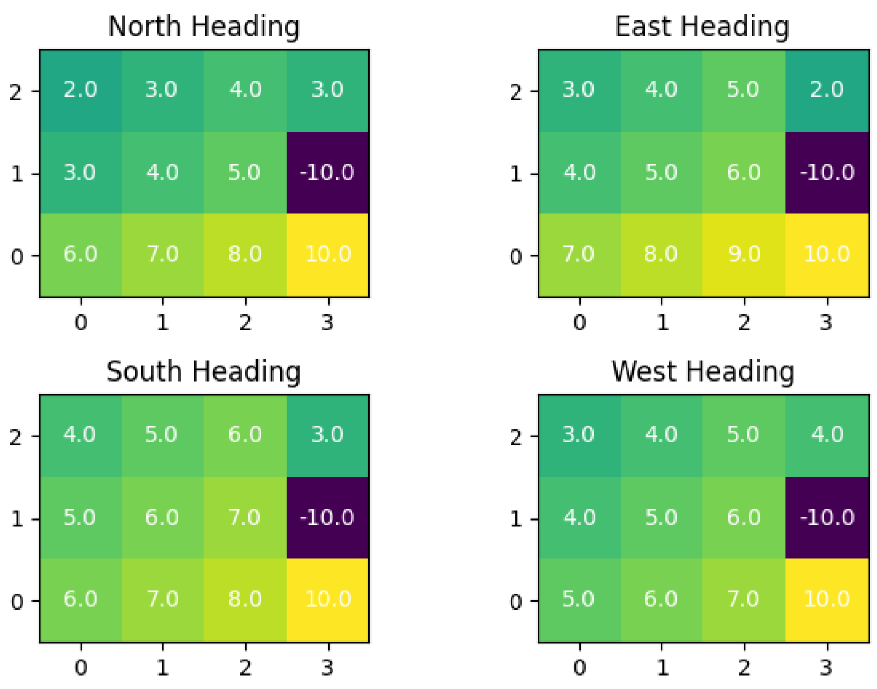

Figure 3: The optimal state value function for the cleaning robot in a simple grid world environment which is 3×4 cells. There are two terminal states — a +10 win state, and a -10 lose state. Note the four different graphs are used to capture the robot state (x, y, θ).

To develop and debug solutions, the engineers have initially developed an empty gridworld. This is a 4×3 grid of empty cells which contains a single cleaning location, a single high traffic location, and travel costs.

a. The engineers propose to use *policy iteration* to develop a policy to control the behaviour of the robot. Describe what policy iteration is for the *deterministic* case, and explain the main steps used in the iteration process. Your answer should include an explanation of the test used to determine when the optimal conditions have been met.

[10 marks]

b. Implement the policy iteration algorithm on the empty gridworld, using the provided code as a base. Present and explain your implementation. Your computed value function should be identical to that shown in Figure 3.

[25+35(c) marks]

CONTINUED

c. Present the final policy computed by your implementation and argue whether it is reasonable or not. One idea for doing this would be to illustrate the trajectory the robot takes from different locations. This includes visualizing the policy and the expected rewards.

[20 marks]

d. Implement value iteration on the empty gridworld and present the final state value function and policy. Compare issue such as iterations results with policy iteration and identify similarities and differences. Do you think value iteration is better or worse in this case?

[25+35(c) marks]

e. Present the final policy computed by the policy iteration and value iteration methods. As with Part c., argue whether you think think the policy is reasonable and illustrate by considering, for example, some of the trajectories the robot undertakes.

[20 marks]

f. Suppose the robot is unsure of its next state due to a malfunctioning local planner. This can be modelled as a nature action. Luckily, the cleaning robot has been fitted with a uncertainty modelling mechanism. Specifically, the robot can model the uncertainty with the transition probability function of the form

$$\mathcal{P}_{ss'} = \mathcal{P}(s'|s,a) = \mathbb{P}[S_{t+1} = s'|S_t = s, A_t = a] \tag{1}$$

where s' is the next state, s is the current state, t is the current step and a is the action. With this new stochastic environment, describe how your approach in policy iteration and evaluation changes.

[10 marks]

[Total 180 marks]

Video Submission

Please note that the video is not separately marked or graded. However, you *must* submit a video if your coursework mark is to count. The reason for providing a video is to help reduce the prevalence of plagiarism. We are not looking to mark / penalise presentation skills.

In your video, you will discuss your solution for each part you attempted for the coursework above. Each video will feature all the members of the team. Each member of the team will be expected to speak for at least a minute, and the overall video should run between 3 and 10 minutes. (Please note that there is no assumption that longer is better. Rather, producing a very compact short video can take a very long amount of time, which is not the intent of this task.)

The video should include the following:

- An introduction to the team. Each team member must introduce themselves on camera.
- For each part of the coursework attempted, explain your solution. You do not need to produced a polished presentation. It is sufficient to highlight parts of the report or code using a mouse cursor and to discuss the presented text. Simply reading out the text of the solution is not sufficient. Instead, you should discuss your solution and identify things such as where the challenges lie.
- For each part of the coursework not attempted, provide a brief explanation of why this was the case.

Recordings carried out using Zoom, Teams, etc. are more than sufficient. One idea might be to arrange a group call in which each person in turn shares their screen and explains one of the questions answered. If you can show videos of algorithms running in real-time this is even more effective, but it is not mandatory.

There are a variety of ways to encode videos including the standard formats used in Zoom or Teams. As noted, the requirement is that the video must be playable by VLC 3.0.5 or later (`https://www.videolan.org/vlc/releases/3.0.5.html`). (VLC is supported on Linux, Mac and Windows. It is typically plays a much wider range of media than, say, the Windows Media Player or the QuickTime Player.) Note that moodle has a maximum file upload size of 160MB.

CONTINUED

Getting Help

You are encouraged to use the assignment discussion forum to help when you get stuck. Please check the forum regularly and try and answer each other's questions. Notifications should now be set up, and we will be checking the forum as often as we can and will be answering any questions that have remained unanswered. Please note that if you have questions about coursework, please post them to the forum directly rather than emailing me. The reason is that all students will be able to see the reply.

Submission and Feedback

The deadline for submission is **12:00 (midday) on Monday 22nd February, 2021**. Late penalties will start to accrue if any element of the submission (report, code or video) has a receipt time of 12:01 or later.