COSC1125/1127 Artificial Intelligence
Assignment 3 Reinforcement Learning
Due: 11:59pm, Monday 3 June 2019

This is an individual or pair assignment. It has **100 points** in total and is worth **15%** of the overall course grade. You may not collude with any other individual, or plagiarise their work. Students are expected to present the results of their own thinking and writing. Never copy another student's work (even if the other student "explains it to you first.") and never give your written work to others. Never copy from the web or any other resource. Remember you are meant to generate the solution to the questions by yourself. Suspected collusion or plagiarism will be dealt with according to RMIT University policy. If you choose to work as a pair, you need to clearly spell out your individual contribution to the assignment. We assume each of you will contribute equally to this assignment.

This assignment has two parts. The first part includes two sets of questions, that we expect you provide written answers. The second part is a programming task requiring you to implement the value iteration and policy iteration functions for an RL scenario [1]

1 Written questions

1.1 Optimal Policy for Simple MDP (20 marks)

Consider the simple n-state MDP shown in Figure 1. Starting from state s_1, the agent can move to the right (a_0) or left (a_1) from any state s_i. Actions are deterministic and always succeed (e.g. going left from state s_2 goes to state s_1, and going left from state s_1 transitions to itself). Rewards are given upon taking an action from the state. Taking any action from the goal state G earns a reward of $r = +1$ and the agent stays in state G. Otherwise, each move has zero reward ($r = 0$). Assume a discount factor $\gamma < 1$.

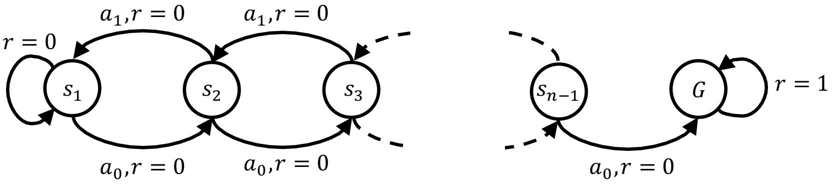

Figure 1: n-state MDP

[1] Acknowledgement: this assignment spec is adapted with permission from Prof. Emma Brunskill of Stanford University.

(a) The optimal action from any state s_i is taking a_0 (right) until the agent reaches the goal state G. Find the optimal value function for all states s_i and the goal state G. (5 marks)

(b) Does the optimal policy depend on the value of the discount factor γ? Explain your answer. (5 marks)

(c) Consider adding a constant c to all rewards (i.e. taking any action from states s_i has reward c and any action from the goal state G has reward $1 + c$). Find the new optimal value function for all states s_i and the goal state G. Does adding a constant reward c change the optimal policy? Explain your answer. (5 marks)

(d) After adding a constant c to all rewards now consider scaling all the rewards by a constant a (i.e. $r_{new} = a(c + r_{old})$). Find the new optimal value function for all states s_i and the goal state G. Does that change the optimal policy? Explain your answer, If yes, give an example of a and c that changes the optimal policy. (5 marks)

1.2 Value Iteration (10 marks)

In this problem we construct an example to bound the number of steps it will take to find the optimal policy using value iteration. Consider the infinite MDP with discount factor $\gamma < 1$ illustrated in Figure 2. It consists of 3 states, and rewards are given upon taking an action from the state. From state s_0, action a_1 has zero immediate reward and causes a deterministic transition to state s_1 where there is reward $+1$ for every time step afterwards (regardless of action). From state s_0, action a_2 causes a deterministic transition to state s_2 with immediate reward of $\gamma^2/(1-\gamma)$ but state s_2 has zero reward for every time step afterwards (regardless of action).

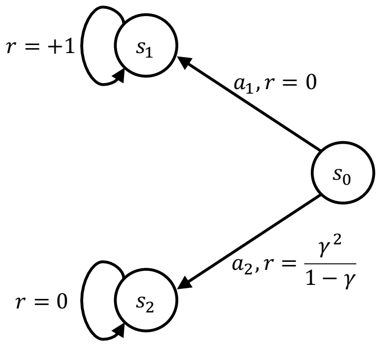

Figure 2: infinite 3-state MDP

(a) What is the total discounted return ($\sum_{t=0}^{\infty} \gamma^t r_t$) of taking action a_1 from state s_0 at time step $t = 0$? (5 marks)

(b) What is the total discounted return ($\sum_{t=0}^{\infty} \gamma^t r_t$) of taking action a_2 from state s_0 at time step $t = 0$? What is the optimal action? (5 marks)

2 Frozen Lake MDP (70 marks)

Now you will implement value iteration and policy iteration for the Frozen Lake environment from OpenAI Gym. We have provided custom versions of this environment in the start-up code.

(a) **(coding)** Read through `vi_and_pi.py` and implement `policy_evaluation`, `policy_improvement` and `policy_iteration`. The stopping tolerance (defined as $\max_s |V_{old}(s) - V_{new}(s)|$) is tol $=$ 10^{-3} . Use $\gamma = 0.9$. Return the optimal value function and the optimal policy. (30 marks)

(b) **(coding)** Implement `value_iteration` in `vi_and_pi.py`. The stopping tolerance is tol $=$ 10^{-3} . Use $\gamma = 0.9$. Return the optimal value function and the optimal policy. (25 marks)

(c) **(written)** Run both methods on the Deterministic-4x4-FrozenLake-v0 and

Stochastic-4x4-FrozenLake-v0 environments. In the second environment, the dynamics of the world are stochastic. How does stochasticity affect the number of iterations required, and the resulting policy? (15 marks)

2.1 Setup

Please be sure you have Python 3.6.x installed on your system. The following instructions should work on Mac or Linux.

[Optional] virtual environment: Once you have unzipped the starter code, you might want to create a virtual environment for the project. If you choose not to use a virtual environment, it is up to you to make sure that all dependencies for the code are installed on your machine. To set up a virtual environment, run the following:

```
cd assignment3
sudo pip install virtualenv      # This may already be installed
virtualenv .env                  # Create a virtual environment
source .env/bin/activate         # Activate the virtual environment
pip install -r requirements.txt  # Install dependencies
# Work on the assignment for a while ...
deactivate                       # Exit the virtual environment
```

Install requirements (without a virtual environment): To install the required packages locally without setting up a virtual environment, run the following:

```
cd assignment3
pip install -r requirements.txt  # Install dependencies
```

3 Submission instructions

You need to submit a zip file (`<yourStudentNumber>.zip`, or `<student1Number+student2Number>.zip` if you work as a pair) containing the following 3 files via the Canvas course page:

1. A text file in PDF called `q1.pdf` containing written answers to Q1.1 and Q1.2.

2. A Python source code called `vi_and_pi.py` containing your `policy_evaluation`, `policy_improvement` and `policy_iteration`, as required in 2(a); `value_iteration`, as required in 2(b).

3. A report in PDF called `report.pdf` containing your result analysis in answering 2(c). If you work as a pair, please write a paragraph detailing how each of you contributes individually to this project.

If you work as a pair, please state clearly at the start of each submitted file, both your student names and student numbers.

Late Submissions: 10% of the possible marks for this assignment will be deducted for each day late and assignments submitted 5 or more days late will not be marked.