Parallel Exercises for Project C

The coding part of Project C consists of completing questions Q1 ... Q7 (**not** Q8) of UC Berkeley's Pac-Man **Project 3 (Reinforcement Learning)**. In parallel, you are required to write a lab-report addressing the answers to the exercises below. General information about expectations and grading can be found on the projects page of the course website.

The exercises correspond to the questions Q1 ... Q7 and can only be answered upon implementation of those questions. Some exercises cover multiple questions, so make sure to read all of them before proceeding with the first one!

Lab-report

Write a report with answers to the exercises below. Be sure to state your answers clearly and to explain and/or motivate all your answers, choices and ideas. Your answers should be a convincing proof of understanding and include, where relevant, concrete experimental results. The projects page on the course website gives an explanation of what we expect concerning the questions that require experiments; please review this information carefully prior to starting the experiments!
You can obtain a maximum of 95 points for the lab-report.

Do not forget to include your names and student-numbers in the report!

The purpose of the first question is to get further acquainted with the Pac-Man framework, and more specifically with the MDP and q-learning code; answer the questions in enough detail to convince us that your understanding of the framework suffices for successful completion of the project.

Exercise 1

a. The files `valueIterationAgents.py` and `qlearningAgents.py` are both set up to implement a type of learning agent. Describe in your own words what the difference between these two types of agent is.

b. In the file `mdp.py` a number of methods is defined. For two of them a remark is made concerning their applicability in reinforcement learning. Describe in your own words why applicability of these methods is restricted or impossible in reinforcement learning.

c. The file `gridworld.py` defines the Gridworld MDP. MDP implementations can differ in how they handle terminal states and when exactly rewards are given. Discuss the implementation of the Gridworld MDP with respect to these issues. Are the Gridworld examples in the course-slides based on a similar implementation? Motivate your answers.

d. Give the definition of a terminal state as can be found in the literature (provide your source), and compare that definition with the implementations of such states in `gridworld.py` and the course slides.

Points: a, b: 2; c, d: 3

Exercise 2

Describe, in your own words, the difference between an *online* planner and an *offline planner*.

Points: 2

Exercise 3

a. The inclusion of one or more terminal states in an MDP is often done to model a certain type of task. How do we call this type of task or the MDP that implements it?

b. The value of a state in an MDP is typically defined as the expected sum of future (possibly discounted) rewards. Should the presence of terminal states in an MDP have an effect on the discounting factor used for computing the value of non-terminal states? Explain your answer.

Points: a: 1; b: 2

The Gridworld MDP has a number of parameters that may influence the computed policy of an agent. In the following two questions we will focus on three of these parameters:

- the discount rate γ for future rewards: option '-d'
- the noise n in the planned movement: option '-n'
- the immediate reward r at each time step: option '-r'

Exercise 4

In Q2 you have to create a value iteration agent that crosses the bridge.

a. Analyse and report for (at least) 5 different discount parameters whether or not the agent crosses the bridge, or shows evidence of at least attempting to cross the bridge. Repeat your analysis for (at least) 5 different noise parameters. Be sure to specify the parameter values used.

b. Which parameter did you change in `analysis.py` and to what value? Clearly motivate why you chose to change this parameter, and why to this specific value. Does it have the desired effect in that the agent crosses the bridge? If so, can you argue why? If not, can you think of an explanation?

Points: a: 4; b: 2

Exercise 5

For Q3:

a. For each of the 5 policy types (3a,. . . , 3e), give the 3-tuple (γ, n, r) that you set in `analysis.py`.

b. For each of the 5 policy types, provide theoretically backed arguments why this parameter setting has the desired effect, or why your answer is 'NOT POSSIBLE'.

Points: a: 0 (-1 if unanswered); b: **10**

A Q-learner uses a number of learning parameters. In the following questions we will focus on:

- the learning rate α (option '-l' in gridworld)
- the exploration rate ϵ (option '-e' in gridworld)

For the crawler robot the parameters can be changed in the GUI.

Recall that the projects page on the course website gives an explanation of what we expect concerning the questions that require experiments. If you haven't read this yet, be sure to do so prior to starting the experiments!

Exercise 6

This exercise assumes that you are able to run the Q-learning crawler robot (see Q5) and experiment with its parameters. **For each experiment you should clearly describe which parameter(s) you vary, what values you considered and what the corresponding outcomes were!**

a. Experiment with 5 different values of learning rate α; use the results to hypothesize about how changes in only the learning rate affect the agent's policies and actions. Provide further evidence for your hypothesis using (theoretical) arguments or additional experimental results.

b. Experiment with 5 different values of exploration rate ϵ; use the results to hypothesize about how changes in only the exploration rate affect the agent's policies and actions. Provide further evidence for your hypothesis using (theoretical) arguments or additional experimental results.

c. Experiment with different combinations of α and ϵ; use the results to hypothesize about how changes in both these parameters affect the agent's policies and actions. Provide further evidence for your hypothesis using (theoretical) arguments or additional experimental results.

d. Does the effect of changing α depend on the value of the discount rate γ? If so, how and why? If not, why not? Give clear (theoretical) arguments and/or describe which experiments you have done to arrive at your conclusion.

e. Does the effect of changing ϵ depend on the value of the discount rate γ? If so, how and why? If not, why not? Give clear (theoretical) arguments and/or describe which experiments you have done to arrive at your conclusion.

Points: a, b: 4; c,d,e: 6

Exercise 7

For each experiment you should clearly describe which parameter(s) you vary, what values you considered and what the corresponding outcomes were! For Q6:

a. Does training with the following command result in an optimal policy?

```
python gridworld.py -a q -k 50 -n 0 -g BridgeGrid -e 1
```

Clearly motivate your answer.

b. Experiment with 5 different values of learning rate α; use the results to hypothesize about how changes in only the learning rate affect the agent's policy. Provide further evidence for your hypothesis using (theoretical) arguments or additional experimental results.

c. Experiment with 5 different values of exploration rate ϵ; use the results to hypothesize about how changes in only the exploration rate affect the agent's policy. Provide further evidence for your hypothesis using (theoretical) arguments or additional experimental results.

d. Experiment with different combinations of α and ϵ; use the results to hypothesize about how changes in both these parameters affect the agent's policy. Provide further evidence for your hypothesis using (theoretical) arguments or additional experimental results. Use your observations to motivate your answer for `question6()` in `analysis.py`.

Points: a: 2; b, c: 4; d: 8

Exercise 8

In Q7 you work with PacmanQAgent. The default Q-learning parameter settings of pacman agents can be changed using the option '-a' (agentArgs); see `python pacman.py -h`.

a. Experiment with the learning parameters (ϵ, γ and α) of the PacmanQAgent on `smallGrid`. Clearly describe your experiment and use your observations to motivate why the default settings for Pacman ($\epsilon = 0.05$, $\gamma = 0.8$, and $\alpha = 0.2$) seem the most effective.

b. Give a detailed sketch of the MDP that the Pacman agent is learning: describe state(s), transitions and rewards (in words or picture(s)). You description should convince us that you understand the MDP beyond the description given in Q7.

c. Train PacmanQAgent on mediumGrid for at least 2000 games and test it. Report:

   - the number of training games you used
   - your settings of the learningparameters (ϵ, γ and α)
   - the average reward over all training games
   - the number of test games you used
   - the win rate and average score

   What was your expectation of Pacman's performance? Are the results better or worse than you expected? At hindsight, can you explain that?

Points: a: **10**; b: 6; c: 4

Exercise 9

List the *autograder* scores for all the project questions:

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Total
x/6	x/1	x/5	x/5	x/3	x/1	x/1	0[1]	$\sum_{Qi}$ x / 22 = ...

Note that we will run the autograder on your code to verify these scores!

Points: 0 (-1 if unanswered)

[1] You are not required to implement Q8; hence the zero points