Name: Yaru Niu GTID: 903525523

Problem Set 2
Deadline: 12:00pm EDT, 03 September 2019

Assistant Professor Matthew Gombolay
CS 8803 - Interactive Robot Learning

August 29, 2019

Instructions: Write your name in the top, left-hand corner. **You may work with others (collaborate) to complete this assignment.** Here, "collaborate" means that you talk about the assignment, teach/learn from each other, and even compare answers. However, you *must* write your own code, and you *must not* help debug each other's code. The execution of the coding must be done on your own. Finally, you must list the names of the people with whom you collaborated. Sign here acknowledging adherence to completing this assignment according to these instructions:

Signature: Yaru Niu

Collaborators: _____

Problem 1. What is the correspondence problem?

Sometimes a direct mapping does not exist between the teacher and learner due to differences in sensing ability, body structure or mechanics. The challenges which arise from these differences are referred to broadly as the correspondence problem.

Problem 2. What are the three peculiarities of how human's teach robots according to the slides from Lecture 2 as gleaned by experiments conducted by Thomaz & Breazeal?

1. Humans use reward channel not only for feedback, but also for future-directed guidance.
2. Humans have a positive bias to their feedback, possibly using the signal as a motivational channel.
3. Humans change their behavior as they develop a mental model of the robotic learner.

Name: Yaru Niu GTID: 903525523

Problem 3. Figure 7.2 of Chapter 7 from Russel & Norvig depicts "a typical Wumpus world."

1. The world is described as a 4x4 grid of cells. How many unique positions are there that the agent could be?

   The agent could be at 16 unique positions.

2. Figure 7.3 shows the possible descriptors or "features" for the cells. How many features are there?

   There are 8 features (A, B, G, OK, P, S, V, W).

3. Given the number of cells in the world and the set of features, how many possible "states" exist in an MDP operating on this world?

   There are $(2^8)^{16}$ possible "states".

Name: Yaru Niu GTID: 903525523

Problem 4. Implement the Wumpus world as depicted in Figure 7.2. The set of actions for your agent are up/down/left/right/stay. However, you must filter out actions that would place the agent outside of the world. For example, the agent in s_o is located at grid cell $(1, 1)$, which means the agent can neither go down or left. Let us assume that, when the agent applies action a (e.g., left), the intended result is achieved with probability $p = 0.9$, and the agent moves to an accidental cell with probability $1 - p = 0.1$. The location it ends up in is drawn uniformly randomly based upon the set of feasible actions.

For example, if the agent is in cell $(2, 2)$ and wants to go up, with probability p the agent will end up in $(2, 3)$; with probability $1 - p$, the agent ends up in one of the following cells: $(2, 2)$, $(3, 2)$, $(1, 2)$, $(2, 1)$, each with probability $(1 - p)/4$. However, in the original state, s_o, if the agent tries to go up, only 2 alternative outcomes are possible: $(1, 1)$ and $(2, 1)$, each with probability $(1 - p)/2$. Finally, if the agent either occupies the same cell as the Wumpus, a pit, or the gold, the game ends (i.e., no further moves are possible). States where this occurs are called "terminal states."

Clarification: Your world should ba an exact copy of the scenario depicted in Figure 7.2. You are *not* randomly initializing the location of the Wumpus, pits, or gold. Their locations are fixed as shown in Figure 7.2. The only randomness you have in the environment is over the outcomes of your actions as defined above. Typically, reinforcement learning problems involve randomness in the initial configuration of the world and often have elements that are active (e.g., the Wumpus is trying to find and eat your agent). However, we are going to start off with something simpler so that you can succeed in the one week time frame given for the homework!

Problem 5. Implement exact Q-learning to solve this problem. To show that your algorithm works, report two metrics/figures. Assume that the agent gets a reward of $+1$ for finding the gold, a reward of -1 for falling in the pit, and a reward of -1 for occupying the same cell as a Wumpus. Remember, when an agent arrives in a terminal state, you do not consider any future rewards – the game ends. As such, you'll need to slightly tweak the q-function update from the pseudo-code presented in class. Finally, you may quit iterating/training once $\Delta = 0.01$.

- Show an x-y plot where the x-axis is the iteration number of the learning algorithm and the y-axis is the Δ value at that iteration.
- Report the value of q-values of the optimal policy (i.e., the one you achieved at the final training iteration) for each of the three actions available to the agent in s_o. Report the value to three decimal places. Δ : 1 : 0.929 ; 2 : 0.829 ; 3 : 0.737 ; 4 : 0.070 ; 5 : 0.009

1. $Q(s_o, \text{"up"}) =$ 0.813
2. $Q(s_o, \text{"right"}) =$ 0.814
3. $Q(s_o, \text{"stay"}) =$ 0.777