Reinforcement Learning Homework

1. Suppose you are training an AI agent to drive a car down a perfectly straight road. The road has three lanes, and you want the AI to always stay in the middle lane.

   Suppose we set up our Markov Decision Process with three states, corresponding to the three lanes of the road, called Left, Center, and Right. So the agent will always be in the state corresponding to its lane.

   The agent may take one of three actions: Go-Straight, Shift-Left, or Shift-Right. These actions are designed to (in a perfect world), keep the car in the same lane, shift one lane to the left, or shift one lane to the right. Shift-Left is not possible from the left lane, and Shift-Right is not possible from the right lane.

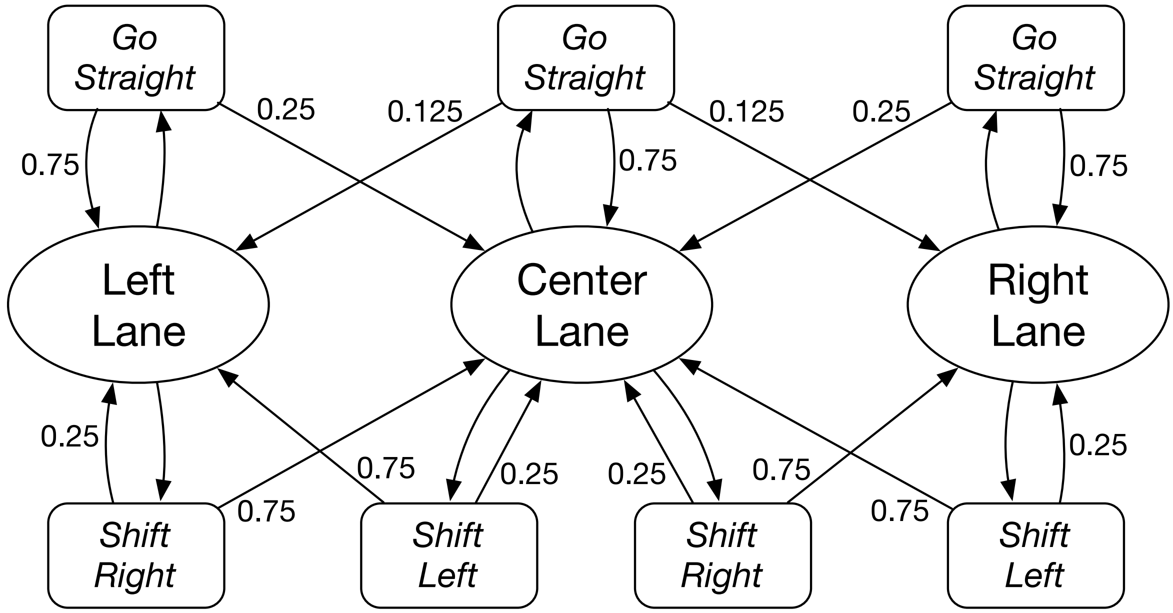
   However, this agent is driving in Memphis, which has streets full of potholes, so each action only ever has a 0.75 probability of succeeding. The other 25% of the time, the agent will end up in a different lane than the one intended, according to this chart:

Agent in Lane	Action	Result and probability	
Left	Go-Straight	Left	0.75
""	""	Center	0.25
""	Shift-Right	Center	0.75
""	""	Left	0.25
Right	Go-Straight	Right	0.75
""	""	Center	0.25
""	Shift-Left	Center	0.75
""	""	Right	0.25
Center	Go-Straight	Center	0.75
""	""	Left	0.125
""	""	Right	0.125
""	Shift-Right	Right	0.75
""	""	Center	0.25
""	Shift-Left	Left	0.75
""	""	Center	0.25

   The agent receives a reward depending on the state it enters (after taking an action). The reward is +2 for being in the center lane, and +1 for being in either the left or right lane.

   a. Draw the MDP describing this situation. To make the diagram less cluttered, do not draw the rewards in the diagram, just the states (Left, Center, Right), and show arrows with the actions and probabilities.

ANSWER:



b. Suppose after running value iteration on this MDP, with a gamma value of 0.9, the values that we converge to are all identical (this is actually true). V[Left] = V[Center] = V[Right] = 17.5.

Use these three V-values and the recursive relationship between the V and Q equations (see the RL notes handout) to fill in the following table of Q-values: (the first is already done for you)

Q[Center, Go-Straight] = **17.5**
Q[Center, Shift-Left] = **17**
Q[Center, Shift-Right] = **17**
Q[Left, Go-Straight] = **17**
Q[Left, Shift-Right] = **17.5**
Q[Right, Go-Straight] = **17**
Q[Right, Shift-Left] = **17.5**

Hint: This is the equation you want to use:

$$Q^*(s,a) = \sum_{s'} P(s' \mid s,a)\big[R(s,a,s') + \gamma V^*(s')\big]$$

c. Use your table of Q-values to derive the optimal policy for the agent (tell me the optimal action for each state). To be fair, this should be pretty obvious.

ANSWER:

π*[Center] = Go-Straight
π*[Left] = Shift-Right
π*[Right] = Shift-Left

d. (1 bonus point extra credit): **Why** are all three V-values the same (17.5), given that we get 2 points for being in the center lane but only 1 point for being in the left or right lanes?

ANSWER: The value of being in any of the three lanes is the same because from *any* lane, we can take an action (the optimal policy, to be specific), that always has a 75% chance of landing us in the center lane (with a reward of +2), and a 25% chance of landing in a different lane (with a reward of +1). In other words, it's no easier to remain in the center lane once you are there than to try to get there from either the left or the right lanes, and therefore the value of being in any lane (which is equivalent to the total future rewards) is the same.

These V-values would not all be the same if, for instance, once you were in the center lane, you had an 85% chance of staying there with the Go-Straight action. Then V[Center] would be higher than V[Left] and V[Right].

2. In the game of Nim, two players alternate taking sticks away from piles. On their turn, each player may remove any number of sticks, but only from **one** pile at a time. The player who is forced to take the last stick *loses*. *(Note: A player loses when they take the last stick of the entire game, not the last stick from a pile.)*

   We will call the two players "A" and "B." We will represent states by the letter of the player who is about to move, followed by the number of sticks left in each pile. So if we start with three piles of 3, 4, and 5 sticks, then the first state of the game would be "A345." We will represent actions by two-digit numbers, the first digit being the number of pile the player chooses to remove sticks from (0-based, so 0=the first pile, 1=the second pile, etc), and the second digit being the number of sticks the player chooses to remove from that pile.

   Let's imagine a game that is close to being over: say we are in state A012. At this point, pile 0 is empty, pile has 1 stick left, and pile 2 has 2 sticks left. Player A has three choices for actions: 11 (choose pile 1, take 1 stick), 21 (choose pile 2, take 1 stick), and 22 (choose pile 2, take 2 sticks). Each of these three actions would result in the next state being B002, B011, and B010. Note that there are no probabilities in this game; it is deterministic: each action only has one possible resulting next-state. (Makes the MDP much simpler).

   The whole game only has two rewards. If "A" wins, they get +1000, and if "B" wins, they get -1000. (So "A" is the maximizing player, and "B" is the minimizing player.)

   We will use gamma = 0.9, and alpha = 1. The reason we use alpha=1 is because in a deterministic environment (one with no randomness), Q-learning learns quickest with alpha=1 and still guarantees convergence to the correct Q* values.

In this project, you will walk through a few rounds of (2-player) Q-learning with the game of Nim. Specifically, we will start with piles of 0, 1, and 2 sticks at the beginning of each game, and player A always moves first.

Here are the games I want you to use. (Remember, in the real world, these games would all be played according to some policy that balances exploration and exploitation, but I'm going to just tell you what moves to make in this case.)

Game 1: A plays action 22, then B plays action 11 (ends the game).
Game 2: A plays action 11, then B plays action 22 (ends the game).
Game 3: A plays action 11, then B plays action 21, then A plays action 21 (ends the game).
Game 4: A plays action 22, then B plays action 11 (ends the game).

At the end of this homework are two tables similar to those we used in class to practice Q-learning with blackjack. Fill them in for games 1-4 (you can ignore games 5, 6, and 7 unless you want more practice).

Player A update: $Q[s, a] \leftarrow Q[s, a] + \alpha \left[r + \gamma \min_{a'} Q[s', a'] - Q[s, a]\right]$

Player B update: $Q[s, a] \leftarrow Q[s, a] + \alpha \left[r + \gamma \max_{a'} Q[s', a'] - Q[s, a]\right]$

Player A optimal policy: $\pi(s) = \operatorname{argmax}_{a} Q[s, a]$

Player B optimal policy: $\pi(s) = \operatorname{argmin}_{a} Q[s, a]$

Q[s, a]	Init	Game 1	Game 2	Game 3	Game 4	Game 5	Game 6	Game 7	Limit
Q[A012, 11]	0		0	0					
Q[A012, 21]	0								
Q[A012, 22]	0	0			900				
Q[B002, 21]	0			0					
Q[B002, 22]	0		1000						
Q[B011, 11]	0								
Q[B011, 21]	0								
Q[B010, 11]	0	1000			1000				
Q[A001, 21]	0			-1000					
Q[A010, 11]	0								

	s	a	r	s'	max/min Q[s', a']		s	a	r	s'	max/min Q[s', a']
Game 1	A012	22	0	B010	0	5	A012	21			
	B010	11	1000	A000	0			11			
Game 2	A012	11	0	B002	0			21			
	B002	22	1000	A000	0	6	A012	21			
Game 3	A012	11	0	B002	0			21			
	B002	21	0	A001	0			11			
	A001	21	-1000	B000	0	7	A012	11			
Game 4	A012	22	0	B010	1000			21			
	B010	11	1000	A000	0			21			