CS 182, PROBLEM SET 4

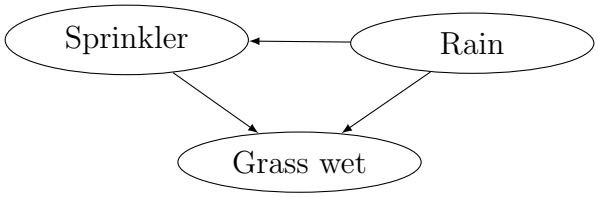
Due: November 15, 2021 11:59pm

This problem set covers Lectures 12, 13, 14, 15. The topics include Bayes' Nets, Hidden Markov Models (HMMs), Markov Decision Processes (MDPs), and Reinforcement Learning (RL).

1. (20 points) *Comprehension.*

(1) *Bayes Nets.* Consider the following Bayes net:

	Sprinkler	
Rain	T	F
F	0.4	0.6
T	0.1	0.9



Rain	
T	F
0.2	0.8

		Grass wet	
Sprinkler	Rain	T	F
F	F	0.1	0.9
F	T	0.2	0.8
T	F	0.3	0.7
T	T	0.4	0.6

In each table, the value of the conditioned variables are presented on the leftmost columns.

(a) (4 points) What is $\mathbb{P}(\text{Grass wet} = \text{True} \mid \text{Rain} = \text{False})$?

(b) (4 points) What is $\mathbb{P}(\text{Grass wet} = \text{True})$?

(c) (4 points) Suppose we did likelihood weighting on this network, where we observe evidence $Sprinkler = True$. What is the weight of a sample

$$\boldsymbol{x} = (Rain = True, Sprinkler = True, GrassWet = False)$$

obtained from the WEIGHTED-SAMPLE algorithm?

Date: November 1, 2021.

(2) (4 Points) *MDP's* Which of the following statements are true for an MDP? Select all that apply and briefly explain why.

(a) (4 points) If one is using value iteration and the values have converged, the optimal policy based on the current values must have converged as well.

(b) (4 points) Policies found by value iteration are superior to policies found by policy iteration, assuming that both algorithms have converged.

2. (20 points) *Hidden Markov Models.* Alice and Bob are living in a house. Bob never leaves the house or checks the weather outside, so he never knows for sure if it is rainy or sunny (assume for simplicity these are the only two weather phenomena). However, Alice does leave the house, and her mood is determined stochastically by the status of the weather that day. Bob is able to make inferences about the weather based on Alice's mood when she gets home, so he uses a hidden markov model in which the weather describes the underlying hidden states and Alice's mood are his observed states. He knows that in his city of Markovtown,

- If it is raining today, then it will rain with probability 0.8 on the next day.
- If it is currently sunny, then it will be sunny with probability 0.6 on the next day.

He also knows that Alice's mood depends on the weather in the following process:

- If it is raining, then Alice will be in a bad mood with probability 0.7.
- If it is sunny, then Alice will be in a good mood with probability 0.6.

One day, Bob checks the weather channel, so he knows for sure that it rained. He then recorded Alice's mood the next three days (not including the day he checked the weather channel): good, good, bad.

(1) (10 points) What is the probability that it will be rainy on day 4?

(2) (10 points) Under Bob's model, what is the most likely sequence of weather states over the three days?

3. (20 points) *Markov Decision Processes.* Annie is a 5-year old girl who loves eating candy and is ambivalent regarding vegetables. She can either choose to eat candy (Hershey's, Skittles, Peanut Butter Cups) or eat vegetables during every meal. Eating candy gives her +10 in happiness points, while eating vegetables only gives her +4 happiness points. But if she eats too much candy while sick, her teeth will all fall out (she won't be able to eat any more). Annie will be in one of three states: healthy, sick, and toothless. Eating candy tends to make Annie sick, while eating vegetables tends to keep Annie healthy. If she eats too much candy, she'll be toothless and won't eat anything else. The transitions are shown in the table below.

Health condition	Candy or Vegetables?	Next condition	Probability
healthy	vegetables	healthy	1
healthy	candy	healthy	1/4
healthy	candy	sick	3/4
sick	vegetables	healthy	1/4
sick	vegetables	sick	3/4
sick	candy	sick	7/8
sick	candy	toothless	1/8

(1) (4 points) Model this problem as a Markov Decision Process: formally specify each state, action, and transition $T(s, a, s')$ and reward $R(a)$ functions.

(2) (7 points) Write down the utility function $U^{\pi}(s)$ for this problem in all possible states under the following policies: π_1 in which Annie always eats candy and π_2 in which Annie always eats vegetables. The discount factor can be expressed as γ.

(3) (9 points) Start with a policy in which Annie always eats candy no matter what the her health condition is. Simulate the first two iterations of the policy iteration algorithm. Show how the policy evolves as you run the algorithm. What is the policy after the third iteration? Set $\gamma = 0.9$.

4. (40 points) *Planning and Reinforcement Learning.* In this problem, you will be implementing various planning and reinforcement learning algorithms on OpenAI's Frozen Lake Environment. You will need the packages `gym==0.21.0`, `IPython==7.29.0`, and `matplotlib==3.4.3`.

In this environment, an agent controls the movement of a character in a 4x4 grid world. Some tiles of the grid are walkable (S, for start, F, for frozen, G, for goal), and others lead to the agent falling into the water (H, for hole). The agent is rewarded +1 for reaching the goal state and 0 otherwise.

We will work with a few variations of the Frozen Lake environment. In the first version, the parameter `is_slippery` is set to False, so every action leads to a deterministic tile. When `is_slippery` is set to True, the movement direction of the agent is uncertain. In particular, if an agent chooses to go in one direction, there is a 1/3 probability the agent goes in the intended direction and a 1/3 probability that the agent goes in each of the directions that are perpendicular to the intended direction. If an agent is on the edge of the map and attempts to move off the map, it simply stays in place.

Please submit your code files along with this pdf in the same Gradescope submission. The Gradescope autograder scores correspond to the points from parts 2a and 3a. For the plots to add to your PDF submission, use `matplotlib`, but do not include this code in your submission to the autograder because the autograder does not support this package.

(1) (4 points) Model this problem as a Markov Decision Process: formally specify the states (including terminal states), actions, and transition and reward functions.

(2) (a) (10 points) Implement value iteration in `pset4.py` by filling out the method `value_iteration` within the class `DynamicProgramming`. You may find `updated_action_values` to be a useful helper function when writing this.

(b) (10 points) Write the mean and variance of the rewards over 1000 episodes of the final policy using the parameter `gamma` = 0.9. For an agent using the policy found by value iteration, plot a histogram of the number of steps it takes an agent to reach the goal. Let's say the agent takes 0 steps to reach the goal if the agent falls into a hole. Does this agent always reach the goal? Why or why not? Use the map to inform your explanation.

(3) (a) (10 points) Implement active, model free reinforcement learning in the form of Q-learning in `pset4.py` by filling out the functions `choose_action` and `q_learning` within the class `QLearning`. Use $\alpha(k_{sa}) = \min(0.1, 10k_{sa}^{-0.8})$[1].

[1]The technical conditions in order to theoretically guarantee convergence is that $\sum_{k_{sa}=1}^{\infty} \alpha(k_{sa}) = \infty$ and $\sum_{k_{sa}=1}^{\infty} \alpha(k_{sa})^2 < \infty$, and while you are welcome to change this so long as you converge to the correct value, this rate was chosen by staff as one that seems to work well in practice for this environment.

(b) (6 points) Plot the mean returns over 100 episodes of the Q-learning agent that acts solely based on max-Q values after every 1000 episodes (this should be done by using the `compute_episode_rewards` function). Use the parameters `gamma` $= 0.9,$ `epsilon` $= 0.001$. How does your agent compare to the agent following the policy derived from part (a)?