15-281 Artificial Intelligence HW7

INSTRUCTIONS

- **Due: Thursday, October 30, 2025 at 10:00 PM EDT.** Remember that you may use up to 2 slip days for the Written Homework making the last day to submit **Saturday, November 1, 2025 at 10:00 PM EDT**.
- **Format:** Write your answers in the `yoursolution.tex` file and compile a pdf (preferred) or you can type directly on the blank pdf. Make sure that your answers are within the dedicated regions for each question/part. If you do not follow this format, we may deduct points. Handwritten solutions are not acceptable and may lead to lost points.
- **How to submit:** Submit a pdf with your answers on Gradescope. Log in and click on our class 15-281, click on the appropriate assignment, and upload your pdf containing your answers.
- **Generative AI:** As per course policy, you may use generative AI tools to better learn course content, but they may not be used to generate any part of your assignment submission. Their purpose is to assist learning, not replace it.
- **Policy:** See the course website for homework policies and academic integrity.

Name	
Andrew ID	
Hours to complete?	

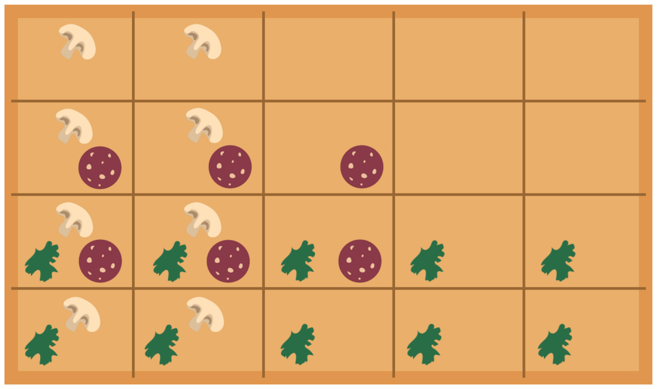
Q1. [11 pts] Probability: Product Rule and Chain Rule

Omega Pizzeria

As we step through the probability concepts of product rule and chain rule, you may find it helpful to check your understanding by referring back to our Omega Pizzeria example.

Let's define the following binary random variables and outcomes:

- X_1: Spinach random variable
  - $-x_1$: no spinach outcome
  - $+x_1$: spinach outcome
- X_2: Mushroom random variable
  - $-x_2$: no mushrooms outcome
  - $+x_2$: mushrooms outcome
- X_3: Pepperoni random variable
  - $-x_3$: no pepperoni outcome
  - $+x_3$: pepperoni outcome



Product Rule

Sometimes you have conditional and marginal distributions, but you actually want to compute the full joint distribution. We know the basic product rule:

$$P(X_1, X_2) = P(X_1 \mid X_2)P(X_2) \tag{1}$$

and

$$P(X_1, X_2) = P(X_2 \mid X_1)P(X_1) \tag{2}$$

We can generalize the product rule a bit more when there are more than two variables. The following are two valid ways to break up the joint distribution of three variables using a more general application of product rule:

$$P(X_1, X_2, X_3) = P(X_1, X_2 \mid X_3)P(X_3) \tag{3}$$

and

$$P(X_1, X_2, X_3) = P(X_1 \mid X_2, X_3)P(X_2, X_3) \tag{4}$$

You can check the math with the pizzeria to make sure that all three of these sentences are equal:

- The probability of getting a slice with spinach, mushrooms, and pepperoni.
- (The probability of getting a slice with spinach and mushrooms, given that we asked for a slice with pepperoni) times (the probability of getting a slice with pepperoni)
- (The probability of getting a slice with spinach, given that we asked for a slice with mushrooms and pepperoni) times (the probability of getting a slice with mushrooms and pepperoni)

Chain Rule

We can further break down equation 4 by applying the product rule again, $P(X_2, X_3) = P(X_2 \mid X_3)P(X_3)$:

$$\begin{aligned} P(X_1, X_2, X_3) &= P(X_1 \mid X_2, X_3)P(X_2, X_3) && (4) \\ &= P(X_1 \mid X_2, X_3)P(X_2 \mid X_3)P(X_3) && (5) \end{aligned}$$

This brings us to the general chain rule for N random variables, $X_1, X_2, \ldots, X_N$:

$$P(X_1, X_2, \ldots, X_N) = \prod_{n=1}^{N} P(X_n \mid X_1, \ldots, X_{n-1}) \tag{6}$$

Mutton, Lettuce, and Tomato

"True love is the greatest thing in the world...except a nice MLT: mutton, lettuce, and tomato sandwich, where the mutton is nice and lean..." *–Miracle Max*

You are given the following probability tables for binary random variables M, L, T:

T	$P(T)$
$+t$	0.3
$-t$	0.7

T	L	$P(L \mid T)$
$+t$	$+l$	0.6
$+t$	$-l$	0.4
$-t$	$+l$	0.35
$-t$	$-l$	0.65

L	T	M	$P(M \mid L, T)$
$+l$	$+t$	$+m$	0.1
$+l$	$+t$	$-m$	0.9
$+l$	$-t$	$+m$	0.2
$+l$	$-t$	$-m$	0.8
$-l$	$+t$	$+m$	0.05
$-l$	$+t$	$-m$	0.95
$-l$	$-t$	$+m$	0.5
$-l$	$-t$	$-m$	0.5

(a) [8 pts] Calculate $P(L, T, M)$ from the tables given. Write out calculation for partial credit. Answers are at most 4 decimal places.

L	T	M	$P(L, T, M)$
$+l$	$+t$	$+m$	
$+l$	$+t$	$-m$	
$+l$	$-t$	$+m$	
$+l$	$-t$	$-m$	
$-l$	$+t$	$+m$	
$-l$	$+t$	$-m$	
$-l$	$-t$	$+m$	
$-l$	$-t$	$-m$	

(b) [3 pts] Which of the following are valid decompositions of the joint probability distribution of M, L, and T, given no assumptions about the relationship between these random variables? Select all that apply.

- ☐ i) $P(M)P(L)P(T)$
- ☐ ii) $P(M)P(M, L)P(M, L, T)$
- ☐ iii) $P(M, L \mid T)P(T)$
- ☐ iv) $P(M \mid L, T)P(L)$
- ☐ v) $P(M \mid L, T)P(T)$
- ☐ vi) None of the above

Q2. [14 pts] Probability: Chain Rule, Joint Distributions, and Marginalization

Probability Notation

Let X be a random variable with domain of possible values $\{x_1, \ldots, x_n\}$. In this course we use several abbreviations:

- $P(x_i)$ is an abbreviation for $P(X = x_i)$
- $\sum_x P(x)$ is an abbreviation for $\sum_{x \in \{x_1,\ldots,x_n\}} P(X = x)$

We also use $P(X)$ to refer to the table of probabilities of all elements in X's domain.

X	$P(X)$
x_1	$P(x_1)$
$\vdots$	$\vdots$
x_n	$P(x_n)$

Marginalization

Recall *marginalization*, which means summing out unwanted variables from a joint distribution. Consider three binary random variables A, B, and C with domains $\{+a, -a\}$, $\{+b, -b\}$, and $\{+c, -c\}$, respectively. Remember that $P(A)$ refers to the table of probabilities of all the elements of A's domain.

(a) [3 pts] Express $P(A)$ in terms of the joint distribution $P(A, B, C)$. **Your answer should contain the abbreviated summation notation.**

$P(A) =$

(b) [3 pts] Express $P(A)$ in terms of $P(C)$, $P(B \mid C)$ and $P(A \mid B, C)$. **Your answer should contain the abbreviated summation notation.**

$P(A) =$

(c) [4 pts] Express $P(A \mid C)$ in terms of the joint distribution $P(A, B, C)$. **Your answer should contain the abbreviated summation notation.**

$P(A \mid C) =$

(d) [4 pts] Express $P(A \mid C)$ in terms of $P(C)$, $P(B \mid C)$ and $P(A \mid B, C)$. **Your answer should contain the abbreviated summation notation.**

$P(A \mid C) =$

Q3. [14 pts] More Probability

Pacman is trying to find a ghost in the Pacman grid. The ghost is not visible to Pacman, but Pacman can take sensor readings at different grid locations to help learn more about the ghost's location.

- The sensor lights up one of five different colors that give a noisy indication of how far away the ghost is.
- The grid has 5 x 6 = 30 different locations.

We are given probability tables for:

- $P(G)$: The probability of a ghost being at a location
- $P(C_{1,1} \mid G)$: The probability of the color of the sensor in grid location (1,1) given the location of the ghost.

We want to help Pacman compute:

- $P(G \mid C_{1,1})$: The probability of the ghost being at a location given the color of the sensor in (1,1).

(a) [8 pts] How many entries are in each of the tables? If you sum all the values in the table, what do they sum to?

(i) [2 pts] $P(G)$

Num: | **Sum:**

(ii) [2 pts] $P(C_{1,1} \mid G)$

Num: | **Sum:**

(iii) [2 pts] $P(G \mid C_{1,1})$

Num: | **Sum:**

(iv) [2 pts] $P(G \mid C_{1,1} = orange)$

Num: | **Sum:**

(b) [3 pts] Given that the sensor in location (1,1) is *orange*, write an equation for the probability that the ghost is in location (1,1), referencing the probability tables that we have been given.

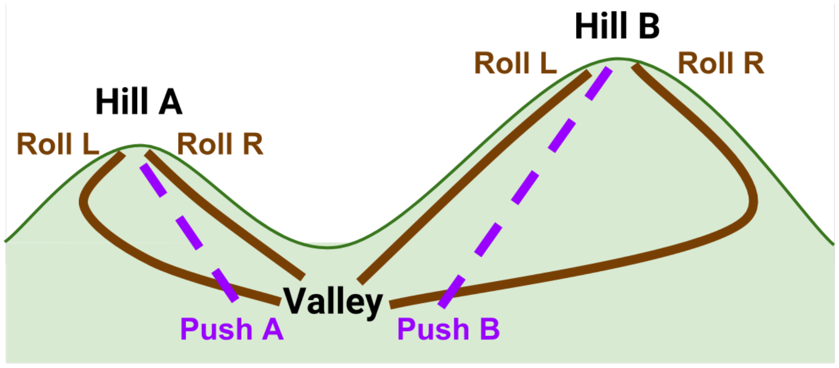
$P(G = ghost_{1,1} \mid C_{1,1} = orange) =$

(c) [3 pts] Given that the sensor in location (1,1) is *orange*, write an equation that computes the most likely value of the variable G, referencing the probability tables that we have been given.

$\hat{G} =$

Q4. [19 pts] More Buggy Bots

Recall the buggy bots problem from Online 6. As a quick refresher, the buggy bots move between three states: $Valley$, $HillA$, and $HillB$. From the valley, bots can choose to push up to Hill A or push up to Hill B. From the top of either hill, the available actions are to roll left or roll right.



In this problem, we explore how domain representation affects performance and computational time when learning policies for MDPs. The true transition matrix and reward functions are repeated below for convenience.

s	a	s'	$T(s,a,s')$	$R(s,a)$
$HillA$	$RollL$	$Valley$	1.0	-2.0
$HillA$	$RollR$	$Valley$	1.0	0.7
$HillB$	$RollL$	$Valley$	0.8	-2.3
$HillB$	$RollL$	$HillA$	0.2	-2.3
$HillB$	$RollR$	$Valley$	0.9	1.5
$HillB$	$RollR$	$HillA$	0.1	1.5
$Valley$	$PushA$	$HillA$	1.0	-0.2
$Valley$	$PushB$	$HillB$	1.0	-0.9

Again, Robot One has a sensor that tells it whether it is in state $HillA$, $HillB$, or $Valley$ (allowing it to fully represent the true state). Robot Two has a simpler representation, having only a hill/valley sensor, which allows it to distinguish between $Valley$ and $HillA$ or $HillB$, but it can't distinguish between $HillA$ and $HillB$. You may assume that if Robot Two is on a $Hill$, then it will be on $HillA$ or $HillB$ with equal probability.

(a) [7 pts] Using a discount factor of $\gamma = 0.9$, Robot One learns the following Q-values:

s	a	$Q(s,a)$
$HillA$	$RollL$	0.302
$HillA$	$RollR$	3.00
$HillB$	$RollL$	0.082
$HillB$	$RollR$	3.84
$Valley$	$PushA$	2.51
$Valley$	$PushB$	2.56

(i) [2 pts] What is the optimal policy for Robot One?

$HillA$:	◯ $RollL$	◯ $RollR$
$HillB$:	◯ $RollL$	◯ $RollR$
$Valley$:	◯ $PushA$	◯ $PushB$

(ii) [2 pts] Using $\gamma = 0.9$, what is the value of this policy? Specify the values with three significant digits. *Note*: You may want to code this up to quickly compute these.

$V^{\pi_1}(HillA)$:	$V^{\pi_1}(HillB)$:	$V^{\pi_1}(Valley)$:

(iii) [2 pts] Using $\gamma = 0.9$, what is a better policy with one time look ahead?

$\pi_{i+1}(HillA)$**:**	$\pi_{i+1}(HillB)$**:**	$\pi_{i+1}(Valley)$**:**

(iv) [1 pt] How many steps of policy iteration did it take for the policy to converge?

Iterations:

(b) [7 pts] Robot Two uses a uniform exploration strategy while learning, so the observed rewards and transition probabilities in the $Hill$ state are the average of the rewards and transitions in $HillA$ and $HillB$, as each are visited roughly equally. Under this exploration strategy, with a discount factor of $\gamma = 0.9$, Robot Two learns the following Q-values:

s	a	$Q(s,a)$
$Hill$	$RollL$	1.79
$Hill$	$RollR$	5.01
$Valley$	$PushA$	4.31
$Valley$	$PushB$	3.61

(i) [1 pt] What is the optimal policy for Robot Two?

$Hill$: ◯ $RollL$ ◯ $RollR$

$Valley$: ◯ $PushA$ ◯ $PushB$

(ii) [2 pts] How is this policy represented in the original domain, i.e how would Robot One represent this policy?

Answer:

(iii) [2 pts] What is the value of this policy in the original domain, i.e. what would the value of this policy be if it were evaluated by Robot One. Specify the values with three significant digits. *Note*: You may want to code this up to quickly compute these.

$V^{\pi_2}(HillA)$**:**	$V^{\pi_2}(HillB)$**:**	$V^{\pi_2}(Valley)$**:**

(iv) [2 pts] What is a better policy with one time look ahead?

$\pi_{i+1}(Hill)$**:**	$\pi_{i+1}(Valley)$**:**

(c) [2 pts] Based on the results from Robot One and Robot Two, answer the following questions:

(i) [1 pt] Which of the policies performs better on the original domain?

◯ Robot One ◯ Robot Two

(ii) [1 pt] Instead of training a new policy for robot two, can the optimal policy we gave you in the original domain be transferred to the new representation? Explain your reasoning in one sentence.

Answer:

(d) [1 pt] Consider a generic MDP in which all actions are deterministic, and all states can be reached from any given state.
Suppose we have two representations of this MDP: R1, which has m states, and R2, which has n states. Assume that one iteration of value iteration using R1 takes t seconds. How long, in terms of t, should we expect one iteration of value iteration to take for R2?

Robot Two time:

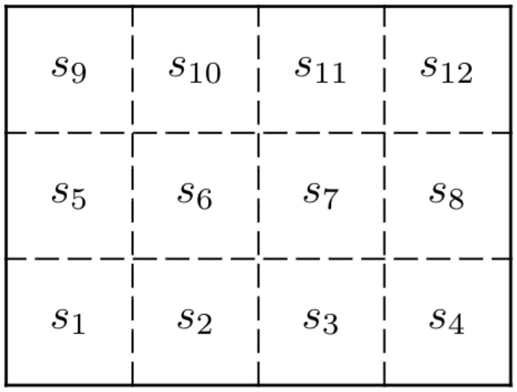
(e) [2 pts] Please give **one Advantage and one Disadvantage** of choosing more expressive representations for problems with large state spaces? Please limit your response to 2-3 sentences.

Answer:

Q5. [16 pts] Dark Room Q-learning

A robot mysteriously finds itself in a dark room with 12 states as shown below. The robot can take four actions at each state (North, East, South, and West). We do not know where the interior walls are, except for the walls surrounding the room (represented by solid lines).

An action against a wall leaves the robot ***in the same state***. Otherwise, the outcome of an action deterministically moves the robot in the direction corresponding to the action.



The robot learns the converged Q-values below. Note that only the maximum values for each state are shown, as the other values (represented with '-') do not affect the final policy.

Table 1: Converged Q-values Learned

	N	E	S	W
s_1	62	62	-	-
s_2	-	73	-	-
s_3	-	86	-	-
s_4	101	-	-	-
s_5	73	-	-	-
s_6	-	-	62	62
s_7	101	101	-	-
s_8	119	-	-	-
s_9	-	86	-	-
s_{10}	-	101	-	-
s_{11}	-	119	-	-
s_{12}	140	140	-	-

(a) [4 pts] Using the learned Q-values, what are the first six actions the robot takes if it starts in state s_6? If there is more than one best action available, choose one randomly. For now, ignore any potential interior walls.
Hint: Try drawing arrows between states in the diagram above representing the optimal actions for each state.

(b) [6 pts] We are told there exists a single state s^* that gives a nonzero reward r^*. All other states give no reward. In other words:

- There is a unique state s^* such that $R(s^*, a, s') = r^* > 0 \; \forall a, s'$.
- For all other states $s \neq s^*$, $R(s, a, s') = 0 \; \forall a, s'$.

(i) [2 pts] How can we use the learned Q-values or the optimal policy to determine the single state, s^*, where we receive a reward?

(ii) [2 pts] What is this state, s^*? Using the learned Q-values, select an optimal action a^* taken from s^*.

$s^* =$	$a^* =$

(iii) [1 pt] Write an equation for $Q(s^*, a^*)$, in terms of $r*$, γ, α, and any other Q-value you deem necessary. Try starting with the update equation for Q-values, but simplify the $max_{a'}$. DO NOT leave a max operation in your equation.

$Q(s^*, a^*) =$

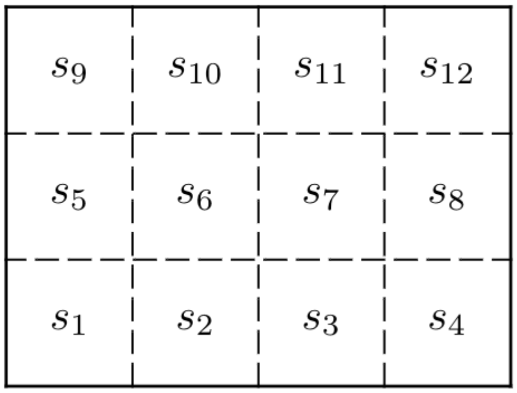
(iv) [1 pt] Using this equation, the given Q-values in the table, and $\gamma = 0.85$, solve for r^*.

(c) [6 pts]

Now the robot wants to locate the interior walls within the room.

(i) [4 pts] Where are these walls? Draw them in by filling the corresponding dashed lines below. For your submission to this problem, you may do one of the following:

- Draw/annotate on top of the existing images in the pdf.
- Edit the `figures/darkroom_states.png` image file to add markings.



(ii) [2 pts] In 1-2 sentences, explain how you know where the walls are.

Q6. [26 pts] Approximate Q-learning

A robot is trying to get to its office hours, occurring on floors 3, 4, or 5 in GHC. It is running a bit late and there are a lot of students waiting for it. There are three ways it can travel between floors in Gates: the stairs, the elevator, and the helix.

The **state** of the robot is the floor that it is currently on (either 3, 4, or 5).

The **actions** that the robot can take are *stairs*, *elevator*, or *helix*.

In this problem, we are using a linear, feature based approximation of the Q-values:

$$Q_w(s,a) = \sum_{i=0}^{3} f_i(s,a) w_i$$

We define the feature functions as follows:

Features	**Initial Weights**
$f_0(s,a) = 1$ (this is a bias feature that is always 1)	$w_0 = 1$
$f_1 = f_{\text{speed}}(s,a) = (\lvert s-4\rvert + 1)t$, where $t = \begin{cases} 20, a = \text{elevator} \\ 15, a = \text{stairs} \\ 10, a = \text{helix} \end{cases}$	$w_1 = 0.2$
$f_2 = f_{\text{accessibility}}(s,a) = \begin{cases} 2, a = \text{stairs} \\ 5, a = \text{helix} \\ 8, a = \text{elevator} \end{cases}$	$w_2 = 3$
$f_3 = f_{\text{emptiness}}(s,a) = \begin{cases} 40, a = \text{elevator}, s = 3 \\ 80, a = \text{elevator}, s = 4 \\ 40, a = \text{elevator}, s = 5 \\ 0, \text{otherwise} \end{cases}$	$w_3 = 0.5$

Furthermore, the weights will be updated as follows:

$$w_i \leftarrow w_i + \alpha[r + \gamma \max_{a'} Q_w(s',a') - Q_w(s,a)] \frac{\delta}{\delta w_i} Q_w(s,a)$$

(a) [6 pts] Calculate the following initial Q values given the initial weights above.

(i) [2 pts] Q_w**(4, elevator):**

(ii) [2 pts] Q_w**(4, stairs):**

(iii) [2 pts] Q_w**(4, helix):**

(b) [2 pts] For this question, suppose that the initial Q-values for state 3 happen to be equal to the corresponding initial Q-values for state 5.

(i) [1 pt] In this problem, as you update the weights, will these values remain equal? I.e., will $Q_w(3, a) = Q_w(5, a)$ given any action a and vector w?

○ Yes ○ No

(ii) [1 pt] Why or why not?

Answer:

(c) [6 pts] Given the Q-values for state 4 calculated in part (a), what are the probabilities that each of the following actions could be chosen when using ϵ-greedy exploration from state 4 (assume random movements are chosen uniformly from all actions)? Recall that ϵ is the probability of taking a random action, while $1 - \epsilon$ is the probability we follow our policy. Let $\epsilon = \frac{1}{3}$ for this problem.

(i) [2 pts] Elevator

$P(elevator) =$

(ii) [2 pts] Stairs

$P(stairs) =$

(iii) [2 pts] Helix

$P(helix) =$

(d) [8 pts] Given a sample with start state 3, action = helix, successor state = 4, and reward = -3, update each of the weights using learning rate $\alpha = 0.15$ and discount factor $\gamma = 0.4$.

(i) [2 pts] w_0

$w_0 =$

(ii) [2 pts] w_1

$w_1 =$

(iii) [2 pts] w_2

$w_2 =$

(iv) [2 pts] w_3

$w_3 =$

(e) [4 pts]

(i) [2 pts] What is an advantage of using approximate Q-learning instead of the standard Q-learning?

Advantage:

(ii) [2 pts] What is a disadvantage of using approximate Q-learning instead of the standard Q-learning?

Disadvantage: