CS 188 Spring 2020 Introduction to Artificial Intelligence Midterm

- You have approximately 110 minutes.
- The exam is closed book, closed calculator, and closed notes except your one-page crib sheet.
- Mark your answers ON THE EXAM ITSELF. Provide a *brief* explanation if applicable.
- For multiple choice questions,
  - ☐ means mark **all options** that apply
  - ◯ means mark a **single choice**
  - When selecting an answer, please fill in the bubble or square **completely** (● and ■)

First name	
Last name	
SID	
Student to your right	
Student to your left	

Your Discussion TA(s) (fill all that apply):

☐ Ajan	☐ Albert	☐ Amitav	☐ Angela	☐ Anusha	☐ Arin
☐ Benson	☐ Carl	☐ Cathy	☐ Charles	☐ Harry (Huazhe)	☐ Jade
☐ Jasmine	☐ Jeffrey	☐ Jierui	☐ Lindsay	☐ Mesut	☐ Pravin
☐ Rachel	☐ Ryan	☐ Saagar	☐ Yanlai		

For staff use only:

Q1.	Are You A Robot?	/0
Q2.	State Spaces	/12
Q3.	Sliding Puzzle	/10
Q4.	MDP	/14
Q5.	Minesweeper	/13
Q6.	Pitcher of Milk	/16
Q7.	Pacman and the Casino	/15
Q8.	A Convoluted Nontrivial Network	/20
	Total	/100

THIS PAGE IS INTENTIONALLY LEFT BLANK

SID: _____

Q1. [0 pts] Are You A Robot?

Welcome to the non-proctored midterm of CS 188 Spring 2020! To ensure that you are a CS 188 student, and you are typing into the correct answer sheet, we will ask you to perform a quick verification. **Failure to complete all parts of this question can result in voiding the exam score.**

(a) Please confirm that the Gradescope Answer Sheet contains the same image as the PDF Exam

- ◯ Yes, my Gradescope Answer Sheet contains the same image as the PDF Exam
- ◯ No, and I have emailed cs188@berkeley.edu about this situation

(b) Which of the squares above contain **bicyclists**? Select all that apply. *You should spend less than 30 seconds on this.*

☐ A ☐ B ☐ C ☐ D ☐ E ☐ F ☐ G ☐ H ☐ I

(c) Please confirm that you are indeed the CS 188 Student whose name will be on this submission, and you will complete this exam individually, abiding by the Berkeley honor code.

◯ I am indeed the CS 188 Student whose name will be on the submission of this exam. I will complete this exam individually, and I will abide by the Berkeley honor code.

Q2. [12 pts] State Spaces

Pacman finds himself in an N by M grid again. He starts in the top left corner of the grid and his goal is to reach the bottom right corner. However there are G ghosts in fixed locations around the grid that Pacman needs to kill before he reaches the goal. Pacman accomplishes this using his new car the Pacmobile.
Every turn Pacman chooses a radius value, r, which can be any integer from **0 to R inclusive.** This shocks and kills all ghosts within r grids of him (by Manhattan distance). However, the Pacmobile has a limited battery that contains E (a positive integer) units of charge. Whenever it produces an electric field of radius r, the Pacmobile loses r units of charge. The Pacmobile has 5 moving actions: left, right, up, down, and stop. All of those choices will cost 1 unit of charge as well. If the Pacmobile runs out of charge Pacman can no longer do anything and the game ends in a loss.

To further clarify, on each turn Pacman chooses a movement action and radius value r. He then shocks all squares in a radius r around him, performs the movement, and loses r + 1 units of charge. Recall he starts with E units of charge and he has to kill G ghosts in fixed locations before reaching the bottom right square to win the game.

(a) Mark true or false for the following subparts (Assume at least one solution exists for the search problem)

(i) [1 pt] Running Breadth First Search on this search problem will return the solution that takes the smallest amount of turns.

◯ (A) True ◯ (B) False

(ii) [1 pt] Running Breadth First Search on this search problem will return the solution that uses the smallest amount of charge.

◯ (A) True ◯ (B) False

(iii) [1 pt] Running Breadth First Search on this search problem will return the solution where Pacman travels the least amount of grid distance.

◯ (A) True ◯ (B) False

(b) Write your answers to the following two subparts in terms of N,M,G,R,E and any scalars you need.

(i) [1 pt] What is the size of the state space, X, using a minimal state representation?

$X =$ [ ]

(ii) [1 pt] What is the maximum possible branching factor from any state?

[ ]

(c) Now we are going to investigate how the size of the state space changes with certain rule changes. Write your answers to the following two subparts **in terms of X** (the size of the original state space) and the terms N,M,G,R,E and any scalars. Each subpart is independent of the other.

(i) [2 pts] The ghosts are now more resilient so they have H health points. This means each ghost needs to spend H turns in the electric field before they die. What is the size of the state space with this adjustment?

$X \times$ [ ]

(ii) [2 pts] The Pacmobile can no longer choose any radius value at every turn because it now takes time to increase and decrease the radius. If pacman is using a radius r on one turn, on the next turn he must choose radius values from $r - 1$, r, or $r + 1$.

$X \times$ [ ]

(d) Mark whether the following heuristics are admissible or not. Following the original problem description.

(i) [1 pt] $h_1(n)$ = The amount of charge remaining.

◯ (A) Admissible ◯ (B) Not Admissible

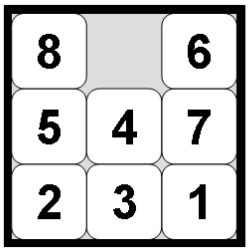
(ii) [1 pt] $h_2(n)$ = The number of ghosts still alive.

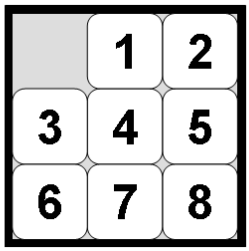
◯ (A) Admissible ◯ (B) Not Admissible

(iii) [1 pt] $h_3(n)$ = The Manhattan distance between the Pacmobile and the bottom right corner.

◯ (A) Admissible ◯ (B) Not Admissible

Q3. [10 pts] Sliding Puzzle





Consider the sliding puzzle game as a search problem. The sliding puzzle game consists of a three by three board with eight numbered tiles and a blank space. At each turn, only a tile adjacent to the blank space (left, right, above, below) can be moved into the blank space. The objective is to reconfigure any generic starting state to the goal state (displayed on the right).

(a) [2 pts]

(i) [1 pt] What is the size of the state space?

◯ (A) 9 ◯ (B) 9*9 ◯ (C) 9!

(ii) [1 pt] Does every board have a unique sequence of moves to reach the goal state?

◯ (A) Yes ◯ (B) No

(b) [8 pts] For the following heuristics, mark whether or not they are only admissible (admissible but not consistent), only consistent (consistent but not admissible), both, or neither.

(i) [1 pt] $h_1(n)$ = Total number of misplaced tiles.

◯ (A) Only admissible ◯ (B) Only consistent ◯ (C) Both ◯ (D) Neither

(ii) [1 pt]

$$h_2(n) = \begin{cases} 0 & \text{if } h_1(n) < 3 \\ h_1(n) & \text{else} \end{cases}$$

◯ (A) Only admissible ◯ (B) Only consistent ◯ (C) Both ◯ (D) Neither

(iii) [1 pt] $h_3(n)$ = Sum of Manhattan distances for each individual tile to their goal location.

◯ (A) Only admissible ◯ (B) Only consistent ◯ (C) Both ◯ (D) Neither

(iv) [1 pt] $h_4(n) = \max(h_1(n), h_3(n))$.

◯ (A) Only admissible ◯ (B) Only consistent ◯ (C) Both ◯ (D) Neither

(v) [1 pt] $h_5(n) = \min(h_1(n), h_3(n))$.

◯ (A) Only admissible ◯ (B) Only consistent ◯ (C) Both ◯ (D) Neither

(vi) [1 pt] $h_6(n) = h_1(n) + h_3(n)$.

◯ (A) Only admissible ◯ (B) Only consistent ◯ (C) Both ◯ (D) Neither

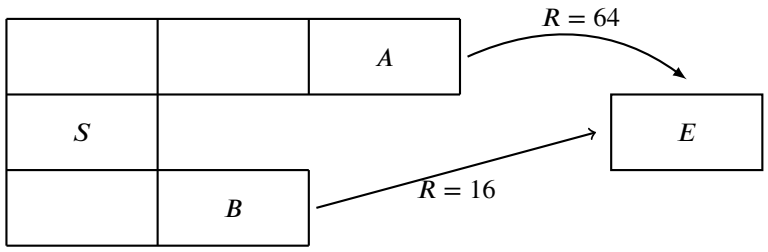
(vii) [3 pts] Comparing the heuristics above that you marked as admissible (or both), which one is best (dominates the others)? If multiple heuristics are tied mark them all.

☐ (A) h_1 ☐ (B) h_2 ☐ (C) h_3 ☐ (D) h_4 ☐ (E) h_5 ☐ (F) h_6

SID: _____

Q4. [14 pts] MDP

Pacman finds himself inside the grid world MDP depicted below. Each rectangle represents a possible state. At each state, Pacman can take actions up, down, left or right. If an action moves him into a wall, he will stay in the same state. At states A and B, Pacman can take the exit action to receive the indicated reward and enter the terminal state, E. Note $R(s, a, s') = 0$ otherwise. Once in the terminal state the game is over and no actions can be taken. Let the discount factor $\gamma = \frac{1}{2}$ for this problem, unless otherwise specified.



(a) **(i)** [1 pt] What is the optimal action at state S?

◯ (A) Up ◯ (B) Down

(ii) [1 pt] How many iterations k will it take before $V_k(S) = V^*(S)$?

$k =$ ☐

(iii) [2 pts] Select all values that $V_k(S)$ will take on during the entire process of value iteration.

☐ (A) 0 ☐ (B) 2 ☐ (C) 4 ☐ (D) 8

☐ (E) 16 ☐ (F) 32 ☐ (G) 64 ◯ (H) None of these

(b) Now Ghost wants to mess with Pacman. She wants to change some of the rules of this grid world so that Pacman does not exit from state A. **All subquestions are independent** of each other so consider each change on its own.

(i) [1 pt] First, Ghost wants to change the discount factor. Write a bound on the discount factor that guarantees Pacman exits from B instead of A. Choose an inequality symbol and fill in the box with a number to compare to. Any valid value of $\gamma \in (0, 1]$ that satisfies your inequality should cause Pacman to exit from B instead of A.

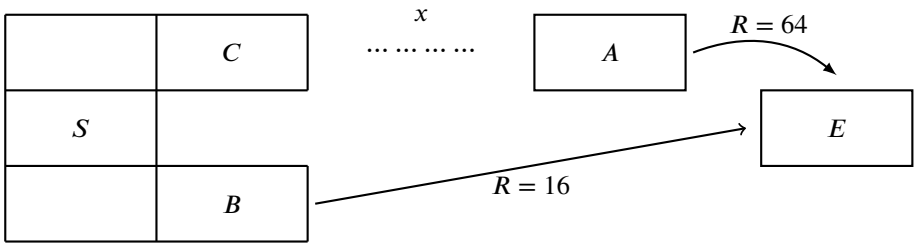
γ ◯ (A) > ◯ (B) < ☐

(ii) [2 pts] Next, Ghost thinks she can change the reward function to accomplish this. Write a bound on the reward from A, $R(A, exit, E)$, that guarantees Pacman exits from B instead of A? Choose an inequality symbol and fill in the box with a number to compare to. Any value of $R(A, exit, E)$ that satisfies your inequality should cause Pacman to exit from B instead of A.

$R(A, exit, E)$ ◯ (A) > ◯ (B) < ☐

(iii) [3 pts] Ghost came up with a bunch of reward functions, $R'(s, a, s')$. Select the new reward functions that cause Pacman not to exit from state A. Note $R(s, a, s')$ is the original reward function from the problem description so the reward from every state is going to be affected.

☐ (A) $R'(s, a, s') = 1 + R(s, a, s')$
☐ (B) $R'(s, a, s') = 100 + R(s, a, s')$
☐ (C) $R'(s, a, s') = -1 + R(s, a, s')$
☐ (D) $R'(s, a, s') = -100 + R(s, a, s')$
☐ (E) $R'(s, a, s') = -R(s, a, s')$
☐ (F) $R'(s, a, s') = 2R(s, a, s')$
◯ (G) None of these



(iv) [2 pts] Ghost realizes she can stop Pacman from exiting from A by adding a certain amount of grids, x, in between C and A as depicted above. Give a lower bound for x that **guarantees** Pacman does not exit from A.

$x \geq$ ____

(c) [2 pts] Another way that Ghost can mess with Pacman is by choosing parameters such that Pacman's value iteration never converges. Select which reward function and discount factor pairs cause value iteration to never converge.

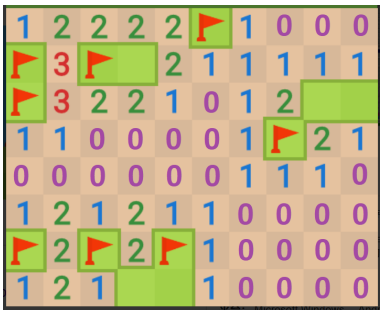
- ☐ (A) $R'(s, a, s') = 100 + R(s, a, s')$, $\gamma = 0.9$
- ☐ (B) $R'(s, a, s') = -100 + R(s, a, s')$, $\gamma = 0.9$
- ☐ (C) $R'(s, a, s') = -1 + R(s, a, s')$, $\gamma = 1.0$
- ☐ (D) $R'(s, a, s') = 1 + R(s, a, s')$, $\gamma = 1.0$
- ◯ (E) None of these

SID: _____

Q5. [13 pts] Minesweeper

In this problem, we will try to solve a classical puzzle game, *Minesweeper*.
We denote grid squares using (x, y) notation, where **x is the row number** and **y is the column number**. $(1, 1)$ would be the top-left grid square.
We denote grid square (a, b) as **adjacent** to grid square (x, y) if $(a, b) \neq (x, y)$, $|a - x| \leq 1$, and $|b - y| \leq 1$. So $(1, 1)$ has 3 adjacent grid squares, and grid squares that are not on the corners or edges have 8 adjacent grid squares.
You are presented an N by M table where each grid square can be contain a mine or it is empty. The grid squares with a digit are explored, and other grid squares are unexplored. Explored grid squares cannot be a mine (or you would have exploded), and the digit shows the total number of mines in its adjacent grid squares.

The player can **mark** a mine by placing a flag on an unexplored grid square. The final goal is to correctly mark all the mines. Below is an example game with some mines already marked.



(a) [6 pts] We can formulate the game as a CSP as follows:

- Each *unexplored grid square* is a variable of domain 2 — either a mine or empty.
- Each *digit* is a constraint indicating the total number of mines among its adjacent grid squares.

Consider the CSPs corresponding to the following two games, where grid squares without a digit are unexplored. Among all the constraints (digits), how many are unary, binary and ternary?

(i) [3 pts]

1	2	2
1		
1	2	2

Unary: _____
Binary: _____
Ternary: _____

(ii) [3 pts]

	1	1
1		
1	1	1

Unary: _____
Binary: _____
Ternary: _____

(b) We run the following **filtering process** to eliminate invalid solutions for a game.

1. For each constraint, if there exists a variable that has exactly one value satisfying the constraint, we assign the variable that value.
2. If any constraint is violated, return "Constraint violated" and terminate.
3. Repeats step 1 and 2 until either all the variables are assigned a value, in which case we have found a solution, or we can no longer uniquely determine the value of any variable, in which case "Game unsolved" is returned.

Select (A) if the filtering process finds a solution (i.e., assign a value to each variable) , and select (B) otherwise. To help you understand the process, the answer to game (1) as well as the explanation are given.

1	2	2
1		
1	2	2

● (A)
○ (B)

Explanation: Consider $\boxed{1}$ at grid square $(2, 1)$. For variable $(2, 2)$, there is only 1 value ("mine") that satisfies the constraint, so we assign "mine" to variable $(2, 2)$. No constraints are violated, so we repeat step 1, considering $\boxed{2}$ at grid square $(1, 2)$. For variable $(2, 3)$, there is only 1 value ("mine") that satisfies the constraint, so we assign "mine" to variable $(2, 3)$. No constraints are violated. We've assigned a value to each variable, hence (A) is selected.

(i) [2 pts)

	2	1
1		
	1	1

○ (A)
○ (B)

(ii) [2 pts]

	1	1
1		
1	1	1

○ (A)
○ (B)

(iii) [2 pts] Because we cannot always use filtering to find a solution, we must use backtracking search. Below is an example of a game that cannot be solved by only running filtering.

	1	1
1		
	1	1

We run backtracking search by considering one variable at a time and applying filtering after each assignment. Suppose we are now considering (1, 1). Note (1,1) is the top left grid.

1. If we assign value "mine" to grid square (1, 1) and then run filtering, what can we conclude?
   ○ (A) We will find a solution ○ (B) Constraint violated ○ (C) Game unsolved
2. If we assign value "empty" to grid square (1, 1) and then run filtering, what can we conclude?
   ○ (A) We will find a solution ○ (B) Constraint violated ○ (C) Game unsolved

(iv) [1 pt] Is there a unique solution if we run backtracking search on game (4) above? If so, select A and fill in the grid squares of mines while leaving others empty. Else, select "no solution" or "multiple solutions."

○ (A) Unique solution
○ (B) No solutions
○ (C) Multiple solutions

D ☐	1	1
1	E ☐	F ☐
G ☐	1	1

Q6. [16 pts] Pitcher of Milk

Consider the task of pushing a cup toward a stationary pitcher of milk (i.e., the pitcher does not move), where both the pitcher and the milk are sitting on the table. We have a robot with an arm made of 7 joints, where each of those 7 joints can be in one of 10 possible angles. We can send commands to tell these joints where to go, and our desired goal in this problem is for the robot to learn how to push the cup toward the pitcher.

(a) [4 pts] **Write the problem as an MDP**

Let's first cast this robotic task as an MDP before we figure out how to solve it. Assign each of the following parameters $(1-5)$ to its corresponding MDP component below. If none of these parameters define a particular component, choose "None of the above."

1. (c_x, c_y): position of the cup on the table
2. (p_x, p_y): position of the pitcher on the table
3. $(j_1, j_2, j_3, j_4, j_5, j_6, j_7)$: angle of each joint in the robot's arm (where each j_i takes only one of the 10 possible values)
4. $(\tau_1, \tau_2, \tau_3, \tau_4, \tau_5, \tau_6, \tau_7)$: torques (i.e., commands that tell a joint how to move) for each joint in the robot's arm
5. $(-d_x, -d_y)$: negative distance between cup and pitcher

1) Action a:
☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ◯ None of the above

2) State s : (Hint: we want the minimal state representation, so only include elements which are necessary)
☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ◯ None of the above

3) Transition function $T(s, a, s')$:
☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ◯ None of the above

4) Reward function $R(s, a, s')$:
☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ◯ None of the above

(b) [6 pts] **Learn Optimal Value Functions**

In class, we discussed approaches for using the Bellman equation to solve MDP's and find out the optimal policy and optimal values for all states.

Similarly, let's say we want to now figure out the optimal values $V(s)$ for the states in this problem. Since we don't know exactly what will happen as a result of various robot actions until we actually execute them in the real world, we must observe episodes (as shown below) and use those to learn the optimal values.

$$E_0 = \{s_0^{(0)}, a_0^{(0)}, r_0^{(0)}, s_1^{(0)}, a_1^{(0)}, r_1^{(0)}, \ldots, s_T^{(0)}, a_T^{(0)}, r_T^{(0)}\}$$
$$E_1 = \{s_0^{(1)}, a_0^{(1)}, r_0^{(1)}, s_1^{(1)}, a_1^{(1)}, r_1^{(1)}, \ldots, s_T^{(1)}, a_T^{(1)}, r_T^{(1)}\}$$
$$\vdots$$
$$E_N = \{s_0^{(N)}, a_0^{(N)}, r_0^{(N)}, s_1^{(N)}, a_1^{(N)}, r_1^{(N)}, \ldots, s_T^{(N)}, a_T^{(N)}, r_T^{(N)}\}$$

Using this collected data, direct evaluation and TD learning are two approaches for learning $V(s)$. The 2 main differences between these approaches are **when** to perform updates, as well as **what** the update is.

(i) [1 pt] Which approach requires you to wait until the end of the episode before you can perform any updates to the value function?
◯ (A) Direct Evaluation
◯ (B) TD Learning

(ii) [2 pts] Select the box that equals the result of direct evaluation.

Here, each state s_i has been visited N_i times. And $N(s_i)$ a set of all places where s_i is seen: that is, $N(s_i) = \{(n, t_n)$: state s_i is seen at time t_n in episode n $\}$.

○ (A) $V(s_i) = \sum_{(n,t_n)\in N(s_i)} \sum_{t'=t_n}^{T} r_{t'}^{(n)}$

○ (B) $V(s_i) = \frac{1}{N_i} \sum_{(n,t_n)\in N(s_i)} \sum_{t'=t_n}^{T} r_{t'}^{(n)}$

○ (C) $V(s_i) = \max_{(n,t_n)\in N(s_i)} \sum_{t'=t_n}^{T} r_{t'}^{(n)}$

○ (D) $V(s_i) = \frac{1}{N_i} \max_{(n,t_n)\in N(s_i)} \sum_{t'=t_n}^{T} r_{t'}^{(n)}$

(iii) [3 pts] Consider a toy example where a collected episode can be written as follows:
$A \xrightarrow{r_1} B \xrightarrow{r_2} C \xrightarrow{r_3} D$

Note that none of the states (A, B, C, D) are visited more than once, so we do not need to explicitly maintain counts in this problem. Fill out the procedures below for what direct evaluation and TD learning would calculate/learn after each collected time step of data. Assume each $r_i = 2$, the discount rate $\gamma = 1$, and the learning rate $\alpha = 1$. Make sure to fill in all the blanks in the answer sheet with what the corresponding capital letter is supposed to be. I.E. for A you should fill in, $V_{DE}(A)$, (the value of A under Direct Evaluation at $t = 1$).

Initialize at $t = 0$

$V_{DE}(A)$	$V_{DE}(B)$	$V_{DE}(C)$	$V_{DE}(D)$
0	0	0	0

$V_{TD}(A)$	$V_{TD}(B)$	$V_{TD}(C)$	$V_{TD}(D)$
0	0	0	0

Collect data: $A \xrightarrow{r_1=2} B$

Update at $t = 1$

Which values below should be updated?

☐ $V_{DE}(A)$ ☐ $V_{DE}(B)$ ☐ $V_{DE}(C)$ ☐ $V_{DE}(D)$
☐ $V_{TD}(A)$ ☐ $V_{TD}(B)$ ☐ $V_{TD}(C)$ ☐ $V_{TD}(D)$

$V_{DE}(A)$	$V_{DE}(B)$	$V_{DE}(C)$	$V_{DE}(D)$
A	B	C	D

$V_{TD}(A)$	$V_{TD}(B)$	$V_{TD}(C)$	$V_{TD}(D)$
E	F	G	H

Collect data: $A \xrightarrow{r_1=2} B \xrightarrow{r_2=2} C$

Update at $t = 2$

Which values below should be updated?

☐ $V_{DE}(A)$ ☐ $V_{DE}(B)$ ☐ $V_{DE}(C)$ ☐ $V_{DE}(D)$
☐ $V_{TD}(A)$ ☐ $V_{TD}(B)$ ☐ $V_{TD}(C)$ ☐ $V_{TD}(D)$

$V_{DE}(A)$	$V_{DE}(B)$	$V_{DE}(C)$	$V_{DE}(D)$
A2	B2	C2	D2

$V_{TD}(A)$	$V_{TD}(B)$	$V_{TD}(C)$	$V_{TD}(D)$
E2	F2	G2	H2

Collect data: $A \xrightarrow{r_1=2} B \xrightarrow{r_2=2} C \xrightarrow{r_3=2} D$

Update at $t = 3$

Which values below should be updated?

☐ $V_{DE}(A)$ ☐ $V_{DE}(B)$ ☐ $V_{DE}(C)$ ☐ $V_{DE}(D)$
☐ $V_{TD}(A)$ ☐ $V_{TD}(B)$ ☐ $V_{TD}(C)$ ☐ $V_{TD}(D)$

$V_{DE}(A)$	$V_{DE}(B)$	$V_{DE}(C)$	$V_{DE}(D)$
A3	B3	C3	D3

$V_{TD}(A)$	$V_{TD}(B)$	$V_{TD}(C)$	$V_{TD}(D)$
E3	F3	G3	H3

(c) [6 pts] **Learn Optimal Policy**

After we use the algorithms described above to estimate $V(s)$ for our problem of pushing the mug, we still have the issue of figuring out the optimal policy $\pi(s)$ (i.e., what should the robot actually do from the state that it's in, in order to achieve high reward and solve the task).

(i) [1 pt] What is a policy in an MDP?

◯ (A) A mapping, π, of states to actions.
◯ (B) A mapping, π, of states to next states.
◯ (C) A mapping, π, of states to maximum expected value of state.

(ii) [1 pt] Which of the following is performing correct policy extraction, from $V(s)$ to $\pi(s)$? Recall that the discount factor, γ, is applied to the value of the future state but not to the reward.

◯ (A) $\pi(s) = \sum_a \sum_{s'} T(s,a,s')\gamma[R(s,a,s') + V(s')]$

◯ (B) $\pi(s) = \max_a \sum_{s'} T(s,a,s')\gamma[R(s,a,s') + V(s')]$

◯ (C) $\pi(s) = \arg\max_a \sum_{s'} T(s,a,s')\gamma[R(s,a,s') + V(s')]$

◯ (D) $\pi(s) = \max_{s'} \sum_a T(s,a,s')\gamma[R(s,a,s') + V(s')]$

◯ (E) $\pi(s) = \arg\max_{s'} \sum_a T(s,a,s')\gamma[R(s,a,s') + V(s')]$

◯ (F) $\pi(s) = \sum_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V(s')]$

◯ (G) $\pi(s) = \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V(s')]$

◯ (H) $\pi(s) = \arg\max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V(s')]$

◯ (I) $\pi(s) = \max_{s'} \sum_a T(s,a,s')[R(s,a,s') + \gamma V(s')]$

◯ (J) $\pi(s) = \arg\max_{s'} \sum_a T(s,a,s')[R(s,a,s') + \gamma V(s')]$

(iii) [2 pts] Instead of knowing the optimal value function $V(s)$, what if we instead know the optimal q-value function $Q(s,a)$? What is the policy extraction procedure from $Q(s,a)$ to $\pi(s)$?

◯ (A) $\pi(s) = \sum_a \sum_{s'} T(s,a,s')\gamma Q(s,a)$

◯ (B) $\pi(s) = \max_a \sum_{s'} T(s,a,s')\gamma Q(s,a)$

◯ (C) $\pi(s) = \sum_a Q(s,a)$

◯ (D) $\pi(s) = \max_a Q(s,a)$

◯ (E) $\pi(s) = \arg\max_a Q(s,a)$

◯ (F) $\pi(s) = \max_{s'} \sum_a T(s,a,s')Q(s,a)$

◯ (G) $\pi(s) = \arg\max_{s'} \sum_a T(s,a,s')Q(s,a)$

(iv) [2 pts] Your friend Cora suggests that for your MDP, it would be much easier to learn $Q(s,a)$ and extract a policy from that, rather than learning $V(s)$ and extracting a policy from that. Do you agree? Why or why not?

◯ (A) Yes, because it's easier to learn Q values than it is to learn V values
◯ (B) Yes, because calculating argmax is easier than calculating a max
◯ (C) Yes, because this avoids the need to learn the full $T(s,a,s')$ for the task
◯ (D) Yes, because this avoids the need for a discount rate
◯ (E) No, because the input space for $V(s)$ is much smaller than $Q(s,a)$
◯ (F) No, because the use of a discount rate is critical for good learning
◯ (G) No, because calculating 1 sum is better than 2 sums for the policy extraction
◯ (H) No, because they're both equally good

SID: _____

Q7. [15 pts] Pacman and the Casino

Tired of hiding from ghosts, Pacman traveled to Las Vegas and went to a casino.

(a) Pacman went to play the lotteries. We assume that Pacman is a rational agent, and denote $U(x)$ as Pacman's utility with respect to x, the amount of dollars he receives.

(i) [1 pt] Let $U(x) = x^2$. Consider a lottery L that has probability of 0.5 to give \$1 and probability 0.5 to give \$3. What is the expected utility of playing the lottery?

[ ] .

(ii) [2 pts] Which of the following are a risk-seeking preference:

☐ (A) $U(x) = 4x$ ☐ (B) $U(x) = \sqrt{x}$ ☐ (C) $U(x) = x\sqrt{x}$
☐ (D) $U(x) = -x^2$ ☐ (E) $U(x) = -\ln x$ ☐ (F) $U(x) = -\sqrt{x}\ln x$
○ (G) None of the above

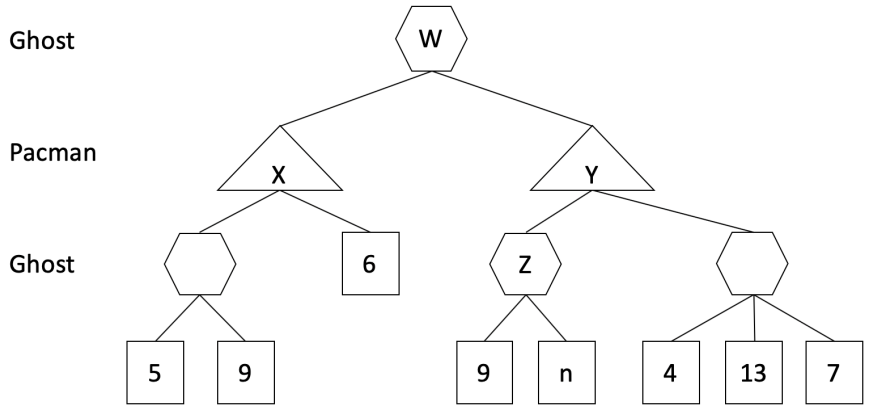
(iii) [3 pts] Pacman is at a lottery that gives 1, 2, or 3 dollars with some probabilities, and the expected number of dollars that the lottery gives is m. Suppose the probability that the lottery gives 1 dollar is p. Now, Pacman has two options: play the lottery, or take m dollars from the casino and leave.

Instructions: You can use p, m, the function U, and scalars to fill in the blanks for this question.

Suppose Pacman will choose to play the lottery if he is indifferent about the choice. Pacman will choose to take m dollars from the casino and leave if and only if:

$U(1) \cdot ($ [A] $) + U(2) \cdot ($ [B] $) + U(3) \cdot ($ [C] $) <$ [D]

(b) Pacman left the lottery and is sitting in front of a machine to play a game, in which there are two players taking turns. Upon the absence of the other player, the machine uses an algorithm that chooses actions uniformly at random. Of course, Pacman is maximizing the score in his moves.
The ghost finds Pacman, and she sneaks into the seat for the other player to play against Pacman, minimizing the score. However, Pacman does not know that the ghost is there.
The value of each node is the score that Pacman actually receives. You can use expressions using n and scalars in the blanks, but your answers must simplified to receive credit.



(i) [1 pt] $X =$ [ ]

(ii) [3 pts] If $-\infty <$ n $<$ [A], W = [B]. Else, W = [C].

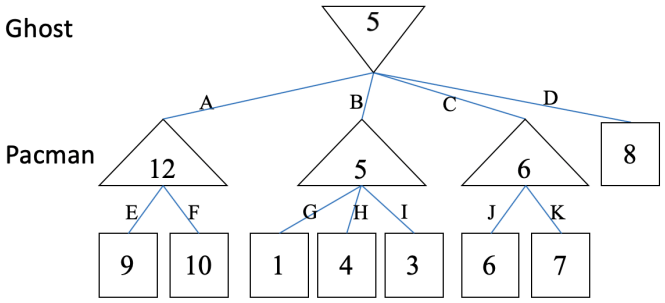
(c) Pacman looked over the machine and found the ghost. They decided to restart the game, Pacman being the maximizer and the ghost being the minimizer, taking turns in the game.

The machine now displays the optimal score of internal nodes for each player. So when Pacman is running alpha-beta pruning, he can check the optimal value of an internal node before looking at its children. However, the machine is not perfect and has errors when displaying internal values. For each internal node i with optimal value x_i, the machine will display $x_i + k_i$. The machine is accurate when displaying the end results.

Each k_i is chosen randomly from a set of values. Pacman knows this set, but does not know any of the individual k_i values.

For all subquestions below, **select the pruned branches** if we use alpha-beta pruning and take into account the internal node information. **We prune on equality.** If a parent branch is pruned, all its **descendant branches must be selected**.

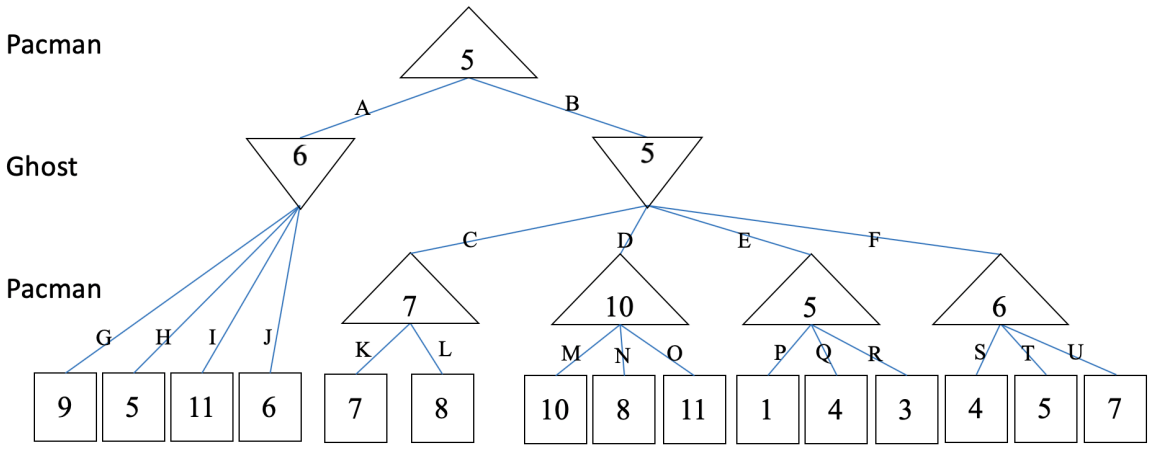
(i) [2 pts] **Set of values for k:** $\forall i,\ k_i \in \{0, \pm 1, \pm 2, \pm 3\}$



☐ A ☐ B ☐ C
☐ D ☐ E ☐ F
☐ G ☐ H ☐ I
☐ J ☐ K

○ (L) None should be pruned

(ii) [3 pts] **Set of values for k:** $\forall i,\ k_i \in \{0, \pm 1\}$



☐ A ☐ B ☐ C
☐ D ☐ E ☐ F
☐ G ☐ H ☐ I
☐ J ☐ K ☐ L
☐ M ☐ N ☐ O
☐ P ☐ Q ☐ R
☐ S ☐ T ☐ U

○ (V) None should be pruned

SID: _____

Q8. [20 pts] A Convoluted Nontrivial Network

In lecture and discussion, you have seen how to use Bayes nets to represent conditional independence relationships between variables. Here we will introduce the concept of a more general graph structure that encodes conditional independence relations: A Markov Random Field (MRF).

Like a Bayes Net, an MRF has edges that represent correlation. However, in an MRF, **edges are undirected and may form cycles.**

An MRF has the following independence assumptions:

- **Local Markov Property**: A variable is conditionally independent of all non-neighbor variables given all of its neighbors.
- **Global Markov Property**: If two subsets of variables $\{X\}, \{Y\}$ are conditionally independent given a third subset of variables $\{Z\}$ that completely separates $\{X\}$ from $\{Y\}$, then $X_i \perp\!\!\!\perp Y_j | \{Z\}, \forall i, j$.

(a) Indicate if each set of independence relations can be encoded in a Bayes Net, MRF, both, or neither:

(i) [1 pt] The past is independent of the future given the present.

◯ (A) Bayes Net Only ◯ (B) Both ◯ (C) Neither ◯ (D) MRF Only

(ii) [1 pt] Common cause triple independence assumptions.

◯ (A) Bayes Net Only ◯ (B) MRF Only ◯ (C) Both ◯ (D) Neither

(iii) [1 pt] Common effect triple independence assumptions.

◯ (A) Bayes Net Only ◯ (B) MRF Only ◯ (C) Both ◯ (D) Neither

(iv) [1 pt] Causal Chain triple independence assumptions.

◯ (A) Bayes Net Only ◯ (B) MRF Only ◯ (C) Both ◯ (D) Neither

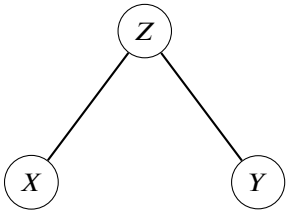
(v) [1 pt] There are $n \geq 3$ students gossiping, where each student hears exactly one secret and whispers exactly one secret. Student 1 starts the whispering, and at each timestep i, the i-th student whispers to the $i+1$-th student (so the $i+1$-th student listens to the i-th student). At the last timestep n, student 1 listens from student n. There are n random variables, each of which represents what each student believes was whispered to them by the previous student.

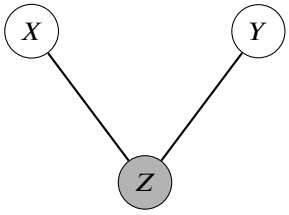
◯ (A) Bayes Net Only ◯ (B) MRF Only ◯ (C) Both ◯ (D) Neither

(vi) [2 pts] You have four variables and want to encode only the 2 following conditional independence relations (and no additional independence or conditional independence relations): $X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$ and $X_2 \perp\!\!\!\perp X_3 \mid X_1, X_4$

◯ (A) Bayes Net Only ◯ (B) MRF Only ◯ (C) Both ◯ (D) Neither

(b) Consider the following two MRFs.





(i) [1 pt] In the left graph, is $X \perp\!\!\!\perp Y$?

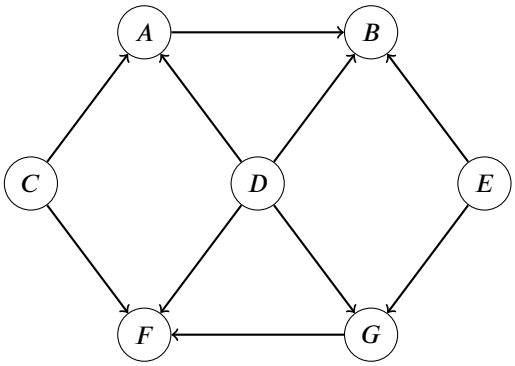
◯ (A) Yes ◯ (B) No

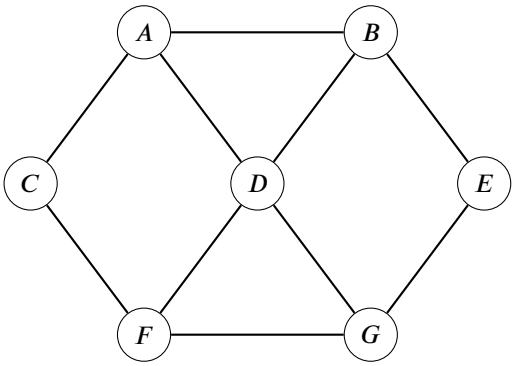
(ii) [1 pt] In the right graph, is $X \perp\!\!\!\perp Y | Z$?

◯ (A) Yes ◯ (B) No

You can think of the above as the active/inactive triples for an MRF and use the Global Markov Property listed above as an analog for the d-separation algorithm of Bayes Nets. Below, we present a Bayes Net (left) and an MRF (right).

(c)





(i) [1 pt] Observe the Bayes Net (left). Is $C \perp\!\!\!\perp E | B, G$?

◯ (A) Yes ◯ (B) No

(ii) [1 pt] Observe the Bayes Net (left). What is the least number of nodes that need to be given for $C \perp\!\!\!\perp D$?

◯ (A) 0 (already independent) ◯ (B) 1 ◯ (C) 2 ◯ (D) 3 ◯ (E) impossible

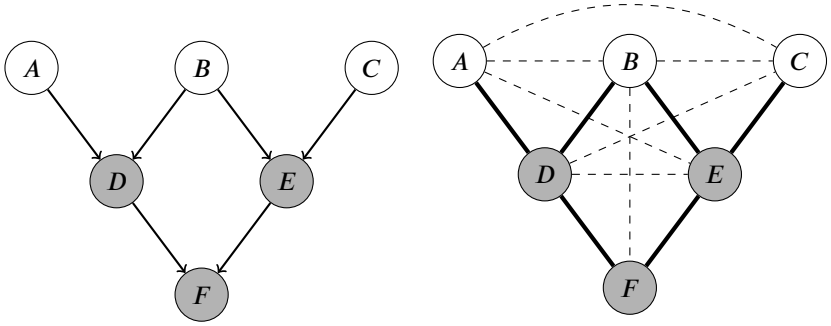
(iii) [1 pt] Observe the MRF (right). Is $C \perp\!\!\!\perp E$ guaranteed?

◯ (A) Yes ◯ (B) No

(iv) [1 pt] Observe the MRF (right). What is the least number of nodes that need to be given for $C \perp\!\!\!\perp D$?

◯ (A) 0 (already independent) ◯ (B) 1 ◯ (C) 2 ◯ (D) 3 ◯ (E) impossible

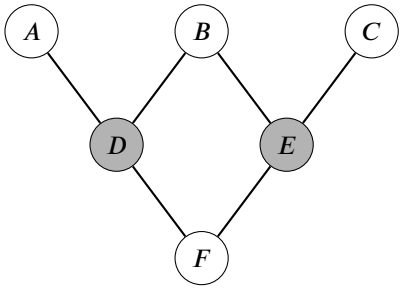
(d) [3 pts] Consider the following Bayes Net (left) and analogous MRF (right), where D, E, and F are given.



Which of the dotted line edges do we need to add so that the resulting MRF can encode the same conditional independence relations among the variables A, B, C as the Bayes Net above?

☐ (A) A $\Longleftrightarrow$ B
☐ (B) A $\Longleftrightarrow$ C
☐ (C) A $\Longleftrightarrow$ E
☐ (D) B $\Longleftrightarrow$ C
☐ (E) B $\Longleftrightarrow$ F
☐ (F) C $\Longleftrightarrow$ D
☐ (G) D $\Longleftrightarrow$ E
◯ (H) No edges need to be added.
◯ (I) None of these enables the MRF to encode the same conditional independence relations as the Bayes Net.

(e) Let's work with the following MRF. Note that F is not given, but D and E still are.



(i) [2 pts] Using only the guaranteed conditional independence assumptions encoded by the MRF above, which of the following expressions is equivalent to $P(A, B, C, F|D = d, E = e)$?

- ☐ (A) $P(A|d)P(B|d,e)P(C|e)P(F|d,e)$
- ☐ (B) $P(A|d,e)P(B|d,e)P(C|d,e)|P(F|d,e)$
- ☐ (C) $P(A)P(B)P(C)P(F|d,e)$
- ☐ (D) $P(A|d,e)P(B|d,e)P(C|A,d,e)P(F|A,d,e)$
- ◯ (E) None of the above

(ii) [2 pts] Using only the guaranteed conditional independence assumptions encoded by the MRF above, which of the following are equivalent to $P(A|B, C, d, e)$? Keep in mind $D = d$ and $E = e$ are still given.

- ☐ (A) $\frac{\sum_{f'} P(A,B,C,d,e,f')}{\sum_{a',f'} P(a',B,C,d,e,f')}$
- ☐ (B) $P(A|d)$
- ☐ (C) $\sum_{b',c'} P(A|b', c', d, e)$
- ☐ (D) $\frac{P(A,B,C,d,e)}{P(e)P(d|e)P(C|e)P(B|d,e)}$
- ◯ (E) None of the above