CS 188 Spring 2020 — Introduction to Artificial Intelligence

Final

- You have approximately 170 minutes.
- The exam is closed book, closed calculator, and closed notes except your two-page crib sheet.
- Mark your answers on gradescope, and click "submit".
- For multiple choice questions,
  - ☐ means mark **all options** that apply
  - ○ means mark a **single choice**

First name	
Last name	
SID	

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.	Potpourri	/17
Q2.	Learn From Old Data	/6
Q3.	Value of Perfect Information	/11
Q4.	Languages	/20
Q5.	Pacman Loses Control	/13
Q6.	A Nonconvolutional Nontrivial Network	/9
Q7.	Searching for a Bayes Network	/13
Q8.	Particle Madness	/11
	Total	/100

THIS PAGE IS INTENTIONALLY LEFT BLANK

Q1. [17 pts] Potpourri

(a) Recall that in search algorithms, that nodes are expanded and each node corresponds to a state s and a path from the start state to s. Recall further that in graph search one can only expand a node corresponding to a state once. Also, below we will denote a heuristic for A^* by $h(\cdot)$. Assume all state transitions have a positive cost.

(i) [1 pt] Which search algorithm would typically have a bigger fringe in graph search for large search problems?

○ BFS ○ DFS

(ii) [1 pt] Which search algorithms are complete for tree search?

☐ BFS ☐ DFS ☐ A^* with any heuristic ☐ A^* with a consistent heuristic. ☐ Uniform Cost Search

(iii) [1 pt] Which search algorithms are optimal for graph search?

☐ BFS ☐ DFS ☐ A* with any heuristic ☐ A* with a consistent heuristic. ☐ Uniform Cost Search

(iv) [1 pt]

If for a constant c and all states s, $h(s) = c$, A^* is equivalent to UCS (uniform cost search).

○ True ○ False]

(v) [1 pt]

If for any pair of states, s and s', $h(s) - h(s') \leq c(s, s')$ where $c(s, s')$ is the minimum cost path from s to s', then A^* will expand a node at most once in a tree search.

○ True ○ False

(vi) [1 pt]

In A^* tree search with any heuristic, one may not get the optimal path.

○ True ○ False

(vii) [1 pt] In a minimax tree of depth 2 (one max layer, one min layer, and a leaf layer) with a branching factor of 3, what is the maximum number of nodes that can be pruned by alpha-beta pruning?

○ 0 ○ 2 ○ 3 ○ 4 ○ 6 ○ 9 ○ None of the above

(viii) [1 pt]

In full depth minimax search, with $\alpha - \beta$ pruning, the minimum number of leaves that can be explored is that one can do in a depth-D tree (the depth is the total depth considering both players) with branching factor B (both players have B options) is

○ $(B/2)^D$ ○ $D^{B/2}$ ○ B^D ○ D^B ○ $B^{D/2}$

(ix) [1 pt]

In expectimax search with two players, one max and the other chance, one can use pruning to reduce the search cost.

○ True ○ False

(b) Suppose we are solving a CSP that has 20 variables (X_i for $i = 1, 2, ..., 20$) and all constraints are binary. X_1 is involved in binary constraints with 6 other variables. X_2 is involved in binary constraints with 9 other variables.
While running the arc consistency algorithm, we reach a point when all variables have 4 values left in their domains, and we have one last arc in the queue: $X_1 \longrightarrow X_2$. (Recall a value x for a variable X is consistent with an arc $X \longrightarrow Y$ if there is still a value y for variable Y where (x, y) is valid for the binary constraint on X and Y.)

(i) [1 pt] Now, we are processing the arc $X_1 \longrightarrow X_2$. We are able to remove a value from the domain of a variable because it was inconsistent, and add the necessary arcs into the queue.
How many arcs are in the queue now?

◯ 3 ◯ 4 ◯ 5 ◯ 6 ◯ 9 ◯ 18
◯ None of these ◯ We cannot determine the exact number of arcs

(ii) [1 pt] Following the previous part, we processed any arcs that may have been added to the queue. No more values were removed from any variable. As a result, we plan to assign a value to one of the variables to continue with backtracking search. Pick the statement below that is most valid.

◯ We should assign a value to X_1 because it has the Least Constraining Value
◯ We should assign a value to X_1 because it has the Minimum Remaining Values
◯ We should assign a value to X_2 because it has the Least Constraining Value
◯ We should assign a value to X_2 because it has the Minimum Remaining Values
◯ We should assign a value to some X_i ($i \geq 3$) that has the Least Constraining Value
◯ We should assign a value to some X_i ($i \geq 3$) that has the Minimum Remaining Values

(c) We are given the traditional game of Pacman with food and 2 ghosts on an M by N grid. Pacman can move up, left, right, or down each turn and he wins by eating all the food. Whenever Pacman moves, the ghosts will move in the opposite direction.

(i) [1 pt] **State representation 1** is to keep track of Pacman's location, the location of both ghosts, and which food was eaten. Please select all of the terms below that should be multiplied together as one product to correctly quantify the number of states in this representation.

☐ 3^{MN} ☐ MN ☐ 2^{MN} ☐ $(MN)^2$

(ii) [1 pt] **State representation 2** is to keep track of whether each food was eaten, and what is in each location (whether it contains Pacman, a Ghost, or nothing). Please select all of the terms below that should be multiplied together as one product to correctly quantify the number of states in this representation.

☐ 3^{MN} ☐ MN ☐ 2^{MN} ☐ $(MN)^2$

(iii) [1 pt] Which state representation would have a higher branching factor?

◯ **State representation 1** will have a higher branching factor ◯ **State representation 2** will have a higher branching factor ◯ They will have the same branching factor

(iv) [1 pt] If we were to run DFS on a particular starting game using both state representations which representation will expand more states?

◯ **State representation 1** will expand more states ◯ **State representation 2** will expand more states
◯ They will expand the same amount of states

(d) [1 pt] Which of the following are true statements about perceptrons?

☐ A larger magnitude of activation corresponds to higher certainty.
☐ The main purpose of the bias term is to restrict the activation values to fall in our desired range.
☐ We can use a black-box binary perceptron to build a multi-class perceptron.
◯ None of the above

(e) [1 pt] You are training a logistic regression model and you find that your training loss is near 0 but test loss is very high. Which of the following is expected to help to reduce test loss? Select all that apply.

☐ (A) Increase the training data size.
☐ (B) Decrease the training data size.
☐ (C) Increase model complexity.
☐ (D) Decrease model complexity.
☐ (E) Train on a combination of your training data and your test data but you test only on your test data
☐ (F) Conclude that Machine Learning does not work

Q2. [6 pts] Learn From Old Data

You encounter a generic MDP and decide to play the game, performing random actions at every timestep. You record all of your transitions in the form (s, a, s', r), where s is the orginal state, a is the action taken, s' is the state you reached, and r is the reward you earned from that transition. After collecting a large amount of these data points you decide reinforcement learning some things about this MDP.

(a) [1 pt] You want to estimate a function $Q(s, a)$ which returns the total rewards you will earn in expectation if you start in state s, take action a, and act optimally from then on. You will do this by initializing $Q(s, a)$ to 0 for every state and action pair. Then you will examine your data points one at a time, updating your approximation after every data point. Assign each of the following terms into an entry of the equation below, which indicates how to use **a collected data point** (s, a, s', r) to update your $Q(s, a)$ function. Fill in the blanks with an integer from 1 to 5 corresponding to the correct expression. Note, α is the learning rate.

1. r
2. α
3. $Q_{\text{curr}}(s', a')$
4. $1 - \alpha$
5. $\max_{a'}$

$$Q_{\text{updated}}(s, a) \leftarrow \boxed{\text{A}}\, Q_{\text{curr}}(s, a) + \boxed{\text{B}}\, [\, \boxed{\text{C}} + \gamma\, \boxed{\text{D}}\, \boxed{\text{E}}\,]$$

(b) After you finish examining your data in the order you collected it and updating the $Q(s, a)$ function accordingly you realize you have converged to the true optimal and accurate $Q(s, a)$ function. However, you have an accident and lose all of your memory about that $Q(s, a)$ function. Even worse, you also randomly shuffled the order of your data points and have no way of knowing what the original order was.

(i) [1 pt] Can you use your procedure from part (a) on the shuffled data to recreate the optimal $Q(s, a)$ given that you only examine each data point once?

◯ Yes ◯ No

(ii) [1 pt] Can you use your procedure from part (a) on the shuffled data to recreate the optimal $Q(s, a)$ given that you are allowed to run the procedure as much as you want (You are allowed to examine the same data points multiple times)?

◯ Yes ◯ No

(c) Later on, you decide to try to approximate a different function, $V^*(s)$ which returns the total rewards you will earn in expectation if you start in state s and act optimally from then on. You want to do this by initializing $V^*(S)$ to 0 for every state. Then you will examine your data points one at a time, updating your approximation after every data point.

(i) [1 pt] Assign each of the following terms into an entry of the equation below, which indicates how to use **a collected data point** (s, a, s', r) to update your $V^*(s)$ function. Note, α is the learning rate.

1. r
2. α
3. $V_{\text{curr}}(s')$
4. $1 - \alpha$

$$V_{\text{updated}} \leftarrow \boxed{\text{A}}\, V_{\text{curr}}(s) + \boxed{\text{B}}\, [\, \boxed{\text{C}} + \gamma\, \boxed{\text{D}}\,]$$

(ii) [2 pts] Can you use your data points to train the true optimal and accurate $V^*(s)$ function?

◯ (A) Yes, because the r and s' in the definition is the result of taking action a from state s. This is true for r and s' from any data point (s, a, s', r) collected under any policy (i.e., even an old one)

◯ (B) Yes, because the source of the data doesn't matter in RL

◯ (C) No, because the old data is useless without the policy

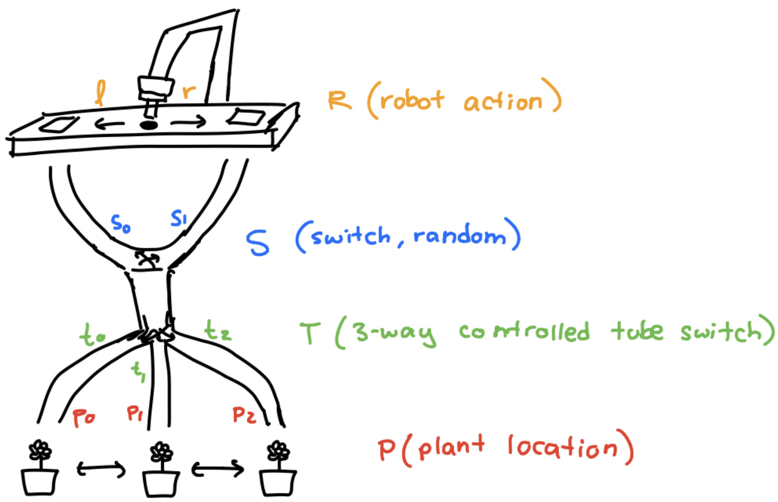
◯ (D) No, because the max cannot be calculated after-the-fact

◯ (E) No, because the r and s' in the definition is the result of taking the optimal action from s. This is only true for r and s' from a data point (s, a, s', r) collected under the most recent/optimal policy

Q3. [11 pts] Value of Perfect Information

Consider the setup shown in the figure below, involving a robotic plant-watering system with some mysterious random forces involved. Here, there are 4 main items at play.
(1) The robot (R) can choose to move either left (l) or right (r). Its chosen action pushes a water pellet into the corresponding opening.
(2) The random switch (S) is arbitrarily in one of two possible positions $\{s_0, s_1\}$. When in position (s_0), it accepts a water pellet only from the (l) tube. When in position (s_1), it accepts a water pellet only from the (r) tube.
(3) A controllable three-way switch (T) can be chosen to be placed in one of three possible positions $\{t_0, t_1, t_2\}$.
(4) A plant (P) is arbitrarily located in one of three possible locations $\{p_0, p_1, p_2\}$. When in position p_i, it can only be successfully watered if the corresponding tube t_i has been selected **and** if the water pellet was sent in a direction that was indeed accepted by the first switch (S).

Finally, in this problem, utility (U) is 1 when the plant successfully receives the water pellet, and 0 otherwise.

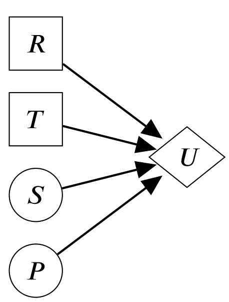


(a) Let's first set this problem up as a decision network.

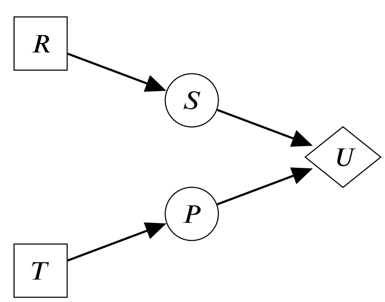
(i) [1 pt] Which of the following decision networks correctly describe the problem described above? Select all that apply. Recall the conventions from the lecture notes:

action nodes as rectangles ☐, chance nodes as ovals ◯, and utility nodes as diamonds ◇.

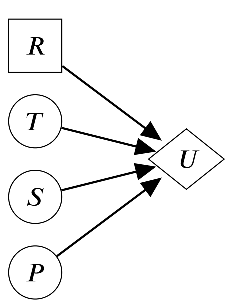
☐ (A)



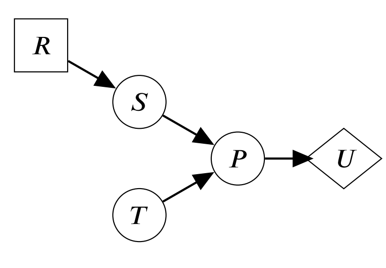
☐ (B)



☐ (C)



☐ (D)



(ii) [1 pt] Fill in the following probability tables, given that there is an equal chance of being at each of their possible locations.

S	$P(S)$

P	$P(P)$

(iii) [1 pt] Consider the table below.

[ ] rows would have $U = 1$, and [ ] rows would have $U = 0$.

R	S	T	P	U
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

(b) Before selecting your actions, suppose that someone could tell you the value of either S or P. Follow the steps below to calculate the **maximum expected utility (MEU)** when knowing S, or when knowing P. Then, decide which one you would prefer to be told.

(i) [1 pt] What is $MEU(S)$?

◯ 0 ◯ $\frac{1}{9}$ ◯ $\frac{1}{6}$ ◯ $\frac{1}{4}$ ◯ $\frac{1}{3}$ ◯ $\frac{1}{2}$
◯ $\frac{2}{3}$ ◯ $\frac{3}{4}$ ◯ $\frac{5}{6}$ ◯ 1 ◯ None of the above

(ii) [1 pt] What is $MEU(P)$?

◯ 0 ◯ $\frac{1}{9}$ ◯ $\frac{1}{6}$ ◯ $\frac{1}{4}$ ◯ $\frac{1}{3}$ ◯ $\frac{1}{2}$
◯ $\frac{2}{3}$ ◯ $\frac{3}{4}$ ◯ $\frac{5}{6}$ ◯ 1 ◯ None of the above

(iii) [1 pt] Would you prefer to be told S or P? ◯ S ◯ P

(c) **(i)** [1 pt] What is $MEU(S,P)$?

◯ 0 ◯ $\frac{1}{9}$ ◯ $\frac{1}{6}$ ◯ $\frac{1}{4}$ ◯ $\frac{1}{3}$ ◯ $\frac{1}{2}$
◯ $\frac{2}{3}$ ◯ $\frac{3}{4}$ ◯ $\frac{5}{6}$ ◯ 1 ◯ None of the above

(ii) [1 pt] In this problem, does $VPI(S,P) = VPI(S) + VPI(P)$? ◯ Yes ◯ No

(iii) [1 pt] In general, does $VPI(a,b) = VPI(a) + VPI(b)$? Select all of the statements below which are true.

◯ Yes, because of the additive property.
◯ Yes, because the order in which we observe the variables does not matter.
◯ Yes, but the reason is not listed.
◯ No, because the value of knowing each variable can be dependent on whether or not we know the other one.
◯ No, because the order in which we observe the variables matters.
◯ No, but the reason is not listed.

(d) For each of the following new variables introduced to this problem, what would the corresponding VPI of that variable be?

(i) [1 pt] A new variable X indicates the weather outside, which affects the overall health of the plant.

◯ VPI(X)<0 ◯ VPI(X)=0 ◯ VPI(X)>0

(ii) [1 pt] A new variable X indicates the weather outside, which affects the metal of switch S such that when it's hot outside, the switch is most likely to remain in position s_0 with probability 0.9 (and goes to s_1 with probability 0.1).

◯ VPI(X)<0 ◯ VPI(X)=0 ◯ VPI(X)>0

Q4. [20 pts] Languages

To discuss a strategy to play against Pacman, the ghosts send each other encrypted messages. Pacman knows that they are using one of English, Romanian, French, German and Swedish. He intercepted the ghosts' messages, but only characters "a", "o", "u", "ä", "ö", "ü" and "â" are decrypted correctly and everything else were lost.
Pacman would like to know which language the ghosts are using. He gathered information about the 5 languages, as below. Assume that characters that are not checked in the table will never appear in texts in that language.

Characters occurrence table:

	a	o	u	ä	ö	ü	â
English	✓	✓	✓				
Romanian	✓	✓	✓				✓
French	✓	✓	✓			✓	✓
German	✓	✓	✓	✓	✓	✓	
Swedish	✓	✓	✓	✓	✓		

Frequency table:

	a	o	u
English	0.4	0.4	0.2
Romanian	0.6	0.1	0.25
French	0.4	0.25	0.3
German	0.4	0.2	0.3
Swedish	0.5	0.2	0.1

(a) Probability Warm Up

(i) [1 pt] If Pacman does not see any ü, select the languages that are possible:

☐ English ☐ Romanian ☐ French ☐ German ☐ Swedish ○ None of the above

(ii) [2 pts]

English	Romanian	French	German	Swedish
%	%	%	%	%

Pacman assumes that the ghosts have chosen their language uniformly randomly, and he did research on the character frequencies and normalized them among the 7 characters in each language, as listed in the frequency table above. Unfortunately Pacman lost the old decryption data and can only recall from memory that he has seen ä. **For languages that are not possible, please write down 0 in the table above.**

The ghosts sent a new message just now, and Pacman has 4 characters successfully decrypted, which are 3 occurrences of a and 1 occurrence of o. Assuming the occurrences of characters are mutually independent, what is Pacman's best estimation of the probabilities of each language, in percentages? **Write down the nearest integers in the table above.**

Pacman thinks it's a good idea to train neural networks to classify the text based on the decrypted characters.

(b) **(i)** [1 pt] Given an arbitrary function from x to y, with enough training time and appropriate hyper-parameters, a neural net with 2 hidden layers that have sufficient number of parameters can gain a training accuracy arbitrarily close to 100% for an arbitrarily large training set. ◯ True ◯ False

(ii) [1 pt] Given any dataset and enough training time, with appropriate hyper-parameters, a sufficiently large neural net can gain a training accuracy arbitrarily close to 100%. ◯ True ◯ False

(iii) [1 pt] Given any dataset and enough training time, with appropriate hyper-parameters, a sufficiently large neural net can gain a test accuracy arbitrarily close to 100%. ◯ True ◯ False

(c) [7 pts]

Given the following concepts in neural nets, match them to the ...

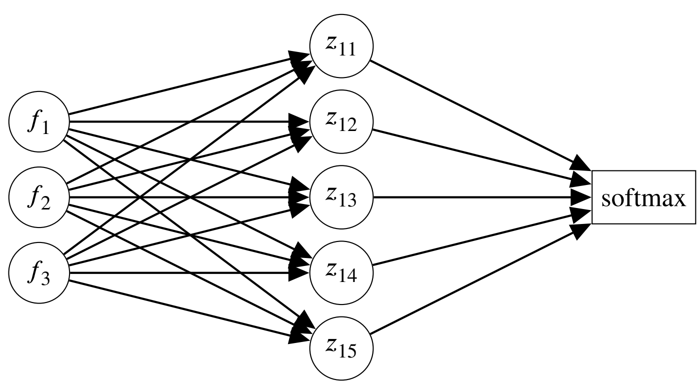
(a) Back Propagation.
(b) Output layer
(c) Hidden unit
(d) Stochastic Gradient Descent
(e) Batch Gradient Descent
(f) Activation function.
(g) Gradient

Match them to the most related concept or procedure in the list.

(1) tanh can be used as the _____
(2) weight vectors are used in a _____
(3) a single data point is used in _____ to compute a(n) _____ using _____
(4) many data points are used in _____ to compute a(n) _____ using _____
(5) _____ can compute a probability distribution over classes

But Pacman decided to start simple.

- f_1, f_2, f_3 are the (normalized) frequencies of "a", "o", "u", respectively
- $z_{1i} = w_{1i} f + b_{1i}$, where $f = [f_1\ f_2\ f_3]^\top$ and each w_{1j} is a 1×3 vector



He built Neural Net A as above.

(d) [1 pt] Pacman used the set up for Project 5 to implement neural network A. He trained it for 1 epoch and got a training accuracy of 50%, but he forgot to save the model. The TA suggests that Pacman can train neural network A from the start again for 1 epoch, passing in the same data in the same order, and Pacman will have the same weights and the 50% training accuracy.

(e) Pacman trained Neural Net A with some data. He tried to classify some new data with the trained model, but the test accuracy was low. What can he do to improve the performance?

(i) [1 pt] Pacman can replace the softmax layer with sigmoid
☐ if the training accuracy is low ☐ if the training accuracy is high ◯ None of the above

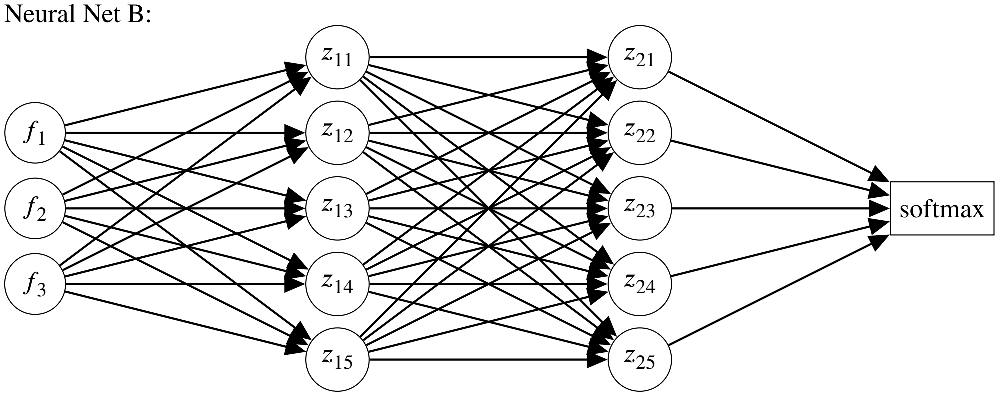
(ii) [1 pt] Pacman can add more nodes ($z_{16}, z_{17}, \ldots$) to the layer

☐ if the training accuracy is low ☐ if the training accuracy is high ◯ None of the above

(iii) [1 pt] Pacman can add more training data

☐ if the training accuracy is low ☐ if the training accuracy is high ◯ None of the above

Neural Net B:



(f) [1 pt] Pacman would like to try adding a layer on Neural Net A to get Neural Net B. Which of the following would lead you to expect Neural Net B to have better training accuracy than Neural Net A?

☐ $z_{2i} = w_{2i} \cdot z_1 + b_{2i}$, where $z_1 = [z_{11}, z_{12}, z_{13}, z_{14}, z_{15}]^\top$

☐ $z_{2i} = w_{2i} \cdot z_1 + b_{2i}$, where $z_1 = [z_{11}, z_{12}, z_{13}, z_{14}, z_{15}]^\top$, and change $z_{1i} = w_{1i} f + b_{1i}$ to $z_{1i} = \text{ReLU}\left(w_{1i} f\right) + b_{1i}$

☐ $z_{2i} = w_{2i} \cdot \text{ReLU}\left(z_1\right) + b_{2i}$, where $z_1 = [z_{11}, z_{12}, z_{13}, z_{14}, z_{15}]^\top$

◯ None of the above

(g) [2 pts] Suppose the activation function for z_{1i} is g, then we can represent the NN in the graph as

☐ $\text{softmax}\left(g\left(w_2(w_1 f + b_1) + b_2\right)\right)$,

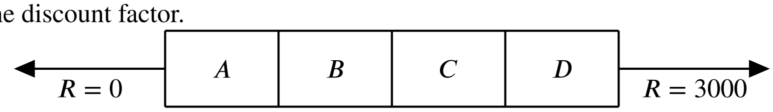
☐ $\text{softmax}\left(w_2 \cdot g\left(w_1 f + b_1\right) + b_2\right)$,

◯ both are incorrect

with dimensions: w_1: ___ × ___, b_1: ___ × ___, w_2: ___ × ___, b_2: ___ × ___.

Q5. [13 pts] Pacman Loses Control

Pacman finds himself inside the grid world Markov Decision Process (MDP) depicted below. Each rectangle represents a possible state. Pacman has two possible actions, left or right. However, these movement actions only work with probability p. With probability q Pacman moves in the opposite direction. Otherwise Pacman stays in the same state. I.E. $T(B, right, C) = p$, $T(B, right, A) = q$ and $T(B, right, B) = 1 - q - p$. If Pacman physically moves right from state D (with probability p if he chooses right from D, or with probability q if he chooses left from D) he earns a reward of 3000 and enters the terminal state where he can no longer perform actions. Similarly if Pacman physically moves left from A he earns a reward of 0 and enters the terminal state. Note γ is the discount factor.



(a) Assume $p = .5$, $q = 0$, and $\gamma = \frac{2}{3}$ for part (a)

(i) [1 pt] What is the value of $V^*(D)$ (the expected value of total discounted rewards pacman can get from state D)

$V^*(D) =$

(ii) [1 pt] What is the value of $V^*(C)$ (the expected value of total discounted rewards pacman can get from state C)

$V^*(C) =$

(b) For each subpart in part (b) you will be given two sets of parameters. Select which set of parameters results in a greater value of $V^*(D)$ (or whether they are the same).

(i) [1 pt] Set I: $\{p = .5, q = 0, \gamma = .5\}$ Set II: $\{p = .6, q = 0, \gamma = .5\}$

○ Set I's $V^*(D)$ is greater ○ Set II's $V^*(D)$ is greater ○ They are equal

(ii) [1 pt] Set I: $\{p = .5, q = 0, \gamma = 1\}$ Set II: $\{p = .6, q = 0, \gamma = 1\}$

○ Set I's $V^*(D)$ is greater ○ Set II's $V^*(D)$ is greater ○ They are equal

(iii) [1 pt] Set I: $\{p = .6, q = .1, \gamma = .5\}$ Set II: $\{p = .1, q = .6, \gamma = .5\}$

○ Set I's $V^*(D)$ is greater ○ Set II's $V^*(D)$ is greater ○ They are equal

(iv) [1 pt] Set I: $\{p = .1, q = 0, \gamma = 1\}$ Set II: $\{p = .8, q = .1, \gamma = 1\}$

○ Set I's $V^*(D)$ is greater ○ Set II's $V^*(D)$ is greater ○ They are equal

(v) [1 pt] Set I: $\{p = .5, q = 0, \gamma = .5\}$ Set II: $\{p = .5, q = .1, \gamma = .5\}$

○ Set I's $V^*(D)$ is greater ○ Set II's $V^*(D)$ is greater ○ They are equal

(c) Assume $p = 0$, $q = .5$, and $\gamma = .5$ for part (c). Pacman decides to use policy iteration to figure out the optimal policy for this MDP. He starts with this policy:

$\pi(A)$	$\pi(B)$	$\pi(C)$	$\pi(D)$
right	left	left	left

(i) [1 pt] Pacman uses policy evaluation to evaluate his policy and then he uses policy improvement to update his policy. Select which states (if any) have a different action according to the policy after improvement is over.

☐ A ☐ B ☐ C ☐ D ○ None changed

(ii) [1 pt] After part (i) we run policy evaluation and policy improvement infinitely more times. How many more times will the policy change (not including a potential change in part (i))?

○ 0 ○ 1 ○ 2 ○ 3 ○ 4 ○ 16 ○ ∞

(d) After some practice runs, Pacman realizes what he thought about $T(s, a, s')$ and $R(s, a, s')$ might be wrong. A policy π is **strictly greedy** with respect to a set of Q-values as long as $\forall s\ \forall a \neq \pi(s)\ Q(s, \pi(s)) > Q(s, a)$ I.E. the Q value for the action chosen by the policy must be strictly greater than the Q value for all other actions. Pacman decides to use approximate q-learning to come up with a strictly greedy policy.

Pacman has 4 feature functions available.

$$f_1(s,a) = \begin{cases} 1 & \text{if s = C} \\ 0 & else \end{cases} \qquad f_2(s,a) = \begin{cases} 1 & \text{if (a = right)} \land \text{(s = A)} \\ 0 & else \end{cases}$$

$$f_3(s,a) = \begin{cases} 1 & \text{if a = left} \\ 0 & else \end{cases} \qquad f_4(s,a) = \begin{cases} 1 & \text{if (a = left)} \land \text{((s = A)} \lor \text{(s = B))} \\ 0 & else \end{cases}$$

He wants to choose weights w_1, w_2, w_3, and w_4 that he can use to calculate $Q(s, a)$ by multiplying each weight by its respective feature and adding the products together. For each policy below select which weights **must be non-zero** (weights can be negative or positive) for the calculated Q-values to generate the policy as a strictly greedy policy. If it is not possible to generate the strictly greedy policy with the given features mark "Not Possible".

(i) [1 pt]

$\pi(A)$	$\pi(B)$	$\pi(C)$	$\pi(D)$
right	right	right	right

☐ w_1 ☐ w_2 ☐ w_3 ☐ w_4 ○ Not Possible

(ii) [1 pt]

$\pi(A)$	$\pi(B)$	$\pi(C)$	$\pi(D)$
right	left	left	left

☐ w_1 ☐ w_2 ☐ w_3 ☐ w_4 ○ Not Possible

(iii) [1 pt]

$\pi(A)$	$\pi(B)$	$\pi(C)$	$\pi(D)$
left	right	left	right

☐ w_1 ☐ w_2 ☐ w_3 ☐ w_4 ○ Not Possible

(iv) [1 pt]

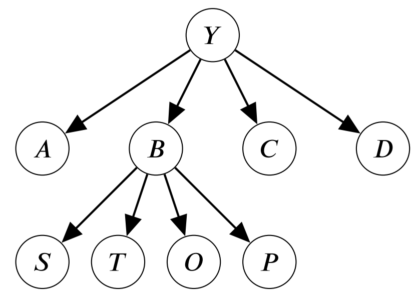
$\pi(A)$	$\pi(B)$	$\pi(C)$	$\pi(D)$
right	left	right	right

☐ w_1 ☐ w_2 ☐ w_3 ☐ w_4 ○ Not Possible

Q6. [9 pts] A Nonconvolutional Nontrivial Network

You have a robotic friend MesutBot who has trouble passing Recaptchas (and Turing tests in general). MesutBot got a 99.99% on the last midterm because he could not determine which squares in the image contained stop signs. To help him ace the final, you decide to design a few classifiers using the below features.

- $A = 1$ if the image contains an octagon, else 0.
- $B = 1$ if the image contains the word STOP, else 0.
  - $S = 1$ if the image contains the letter S, else 0.
  - $T = 1$ if the image contains the letter T, else 0.
  - $O = 1$ if the image contains the letter O, else 0.
  - $P = 1$ if the image contains the letter P, else 0.
- $C = 1$ if the image is more than 50% red in color, else 0.
- $D = 1$ if the image contains a post, else 0.



(a) First, we use a Naive Bayes-inspired approach to determine which images have stop signs based on the features and Bayes Net above. We use the following features to predict $Y = 1$ if the image has a stop sign anywhere, or $Y = 0$ if it doesn't.

(i) [1 pt] Using the independence assumptions encoded in the Bayes Net, which of the following are true?

☐ If we know whether the picture has the word "STOP" (B), the appearance of the letter "S" is independent from the appearance of the letter "T" in the image.

☐ If we know whether the picture has a STOP sign (Y), the appearance of the letter "S" is independent from the appearance of the letter "T" in the image.

☐ $S \perp\!\!\!\perp D | Y$

☐ $A \perp\!\!\!\perp B | \{S, T, O, P\}$

☐ The 7 features (A, S, T, O, P, C, D) satisfy the Naive Bayes independence assumptions.

○ None

(ii) [1 pt] Which expressions would a Naive Bayes model use to predict the label for B if given the values for features $S = s, T = t, O = o, P = p$? Choose all valid expressions.

☐ $b = \arg\max_b P(b)P(s|b)P(t|b)P(o|b)P(p|b)$

☐ $b = \arg\max_b P(s|b)P(t|b)P(o|b)P(p|b)$

☐ $b = \arg\max_b P(b|s, t, o, p)$

☐ $b = \arg\max_b P(b, s, t, o, p)$

☐ $b = \arg\max_b P(s, t, o, p|b)$

○ None

(iii) [1 pt] Which expressions would we use to predict the label for Y with our Bayes Net above? Assume we are given all features except B. So $A = a, S = s, T = t$, etc. For the below choices, the underscore means we are dropping the value of that variable. So $y, \_\_ = (0, 1)$ would mean $y = 0$.

☐ $y, \_\_ = \arg\max_{y,b} P(y)P(a|y)P(b|y)P(c|y)P(d|y)P(s|b)P(t|b)P(o|b)P(p|b)$

☐ $y, \_\_ = \arg\max_{y,b} P(s)P(t)P(o)P(p)P(a)P(b|s, t, o, p)P(c)P(d)P(y|a, b, c, d)$

☐ First compute $b' = \arg\max_b$ of the formula chosen in part (ii).
Then compute $y = \arg\max_y P(y)P(a|y)P(b'|y)P(c|y)P(d|y)$

☐ First compute $b' = \arg\max_b$ of the formula chosen in part (ii).
Then compute $y = \arg\max_y P(y|a, b', c, d)$

☐ $y = \arg\max_y \sum_{b'} P(y)P(a|y)P(b'|y)P(c|y)P(d|y)P(s|b')P(t|b')P(o|b')P(p|b')$

○ None

(iv) [1 pt] One day MesutBot got allergic from eating too many cashews. The incident broke his letter S detector, so that he no longer gets reliable S features. Now what expressions would we use to predict the label for Y? Assume all features except B, S are given. So $A = a, T = t, O = o$, etc.

☐ $y = \arg\max_y P(y)P(a|y)P(c|y)P(d|y)$

☐ $y, \_\_, \_\_ = \arg\max_{y,b,s} P(y)P(a|y)P(b|y)P(c|y)P(d|y)P(s|b)P(t|b)P(o|b)P(p|b)$

☐ $y, \_\_ = \arg\max_{y,s} P(y)P(a|y)P(b|y)P(c|y)P(d|y)P(s|b)P(t|b)P(o|b)P(p|b)$

☐ $y, \_\_ = \arg\max_{y,b} P(y)P(a|y)P(b|y)P(c|y)P(d|y)P(t|b)P(o|b)P(p|b)$

☐ $y, \_\_ = \arg\max_{y,b} P(y)P(a|y)P(b|y)P(c|y)P(d|y)P(s|b)P(t|b)P(o|b)P(p|b)$

☐ $y, \_\_ = \arg\max_{y,b} P(y|a, b, c, d)$

☐ $y = \arg\max_y P(y)P(a|y)P(c|y)P(d|y) \sum_{b',s'} P(b'|y)P(s'|b')P(t|b')P(o|b')P(p|b')$

○ None

(b) [1 pt] You decide to try to output a probability $P(Y|features)$ of a stop sign being in the picture instead of a discrete ± 1 prediction. We denote this probability as $P(Y|\vec{f}(x))$. Which of the following functions return a **valid** probability distribution for $P(Y = y|\vec{f}(x))$? Recall that $y \in \{-1, 1\}$.

☐ $\frac{e^{y \cdot \vec{w}^T \vec{f}(x)}}{e^{-y \cdot \vec{w}^T \vec{f}(x)} + e^{y \cdot \vec{w}^T \vec{f}(x)}}$

☐ $\frac{1}{2}$

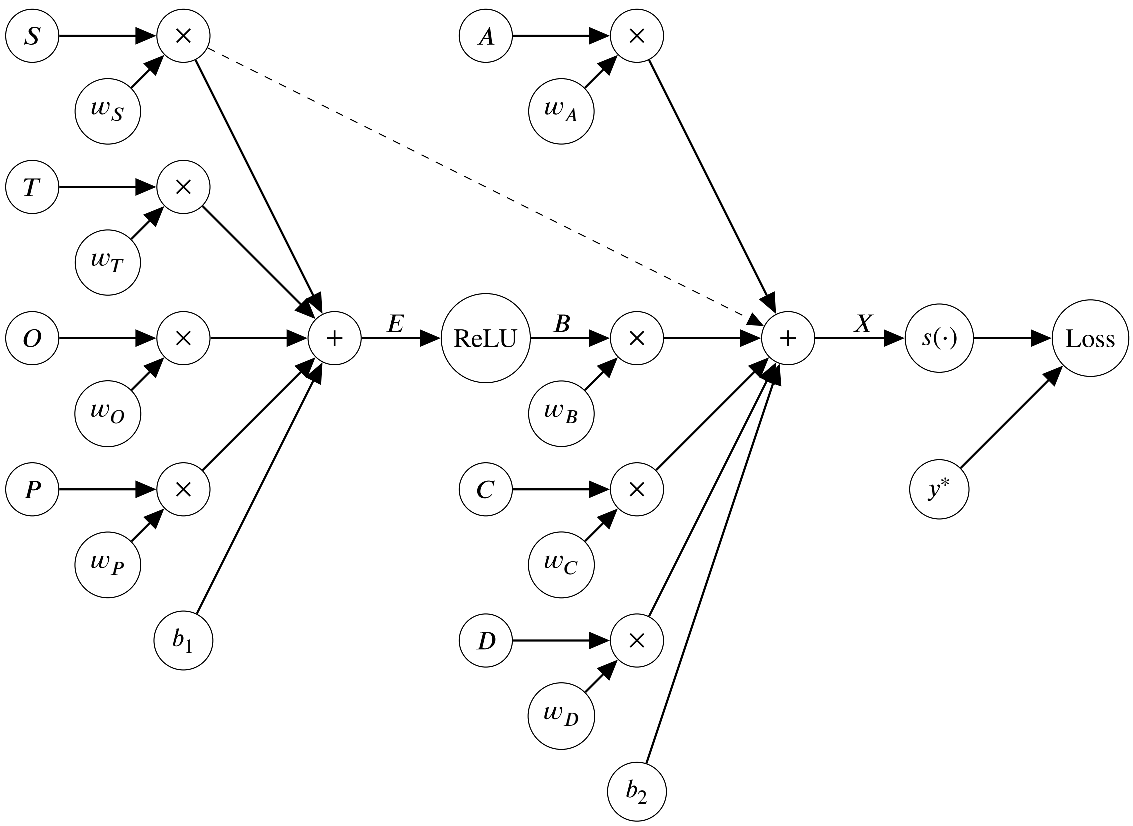
☐ $\frac{0.5}{1+e^{-\vec{w}^T \vec{f}(x)}}$

☐ $\frac{-1}{1+e^{\vec{w}^T \vec{f}(x)}} + 1$

○ None

SID: ____________________________

Unimpressed by the perceptron, you note that features are inputs into a neural network and the output is a label, so you modify the Bayes Net from above into a Neural Network computation graph. Recall the logistic function $s(x) = \frac{1}{1+e^{-x}}$ has derivative $\frac{\partial s(x)}{\partial x} = s(x)[1 - s(x)]$



(c) For this part, ignore the dashed edge when calculating the below.

(i) [1 pt] What is $\frac{\partial Loss}{\partial w_A}$?

◯ $\frac{\partial Loss}{\partial s(X)} \cdot [s(X) \cdot (1 - s(X))] \cdot A$

◯ $2(s(X) - y^*) \cdot [s(X) \cdot (1 - s(X))] \cdot A$

◯ $\frac{\partial Loss}{\partial s(X)} \cdot [s(X) \cdot (1 - s(X))] \cdot 2A + 1$

◯ $\frac{\partial Loss}{\partial s(X)} \cdot [s(X) \cdot (1 - s(X))] \cdot 2A$

◯ $2(s(X) - y^*) \cdot [s(X) \cdot (1 - s(X))] \cdot A + 1$

◯ $\frac{\partial Loss}{\partial s(X)} \cdot [s(X) \cdot (1 - s(X))] \cdot A + 1$

◯ None

(ii) [1 pt] What is $\frac{\partial Loss}{\partial w_S}$? Keep in mind we are still ignoring the dotted edge in this subpart.

- ◯ $\frac{\partial Loss}{\partial s(X)} \cdot [s(X) \cdot (1 - s(X))] \cdot w_B \cdot \left(\begin{cases} 1 & E \geq 0 \\ 0 & E < 0 \end{cases} \right) \cdot S$
- ◯ $2(s(X) - y^*) \cdot [s(X) \cdot (1 - s(X))] \cdot w_B \cdot \left(\begin{cases} 1 & E \geq 0 \\ 0 & E < 0 \end{cases} \right) \cdot S$
- ◯ $\frac{\partial Loss}{\partial s(X)} \cdot [s(X) \cdot (1 - s(X))] \cdot w_B \cdot \left(\begin{cases} 1 & E \geq 0 \\ 0 & E < 0 \end{cases} \right) \cdot 2S + S$
- ◯ $\frac{\partial Loss}{\partial s(X)} \cdot [s(X) \cdot (1 - s(X))] \cdot w_B \cdot \left(\begin{cases} 1 & E \geq 0 \\ 0 & E < 0 \end{cases} \right) \cdot 2S$
- ◯ $2(s(X) - y^*) \cdot [s(X) \cdot (1 - s(X))] \cdot w_B \cdot \left(\begin{cases} 1 & E \geq 0 \\ 0 & E < 0 \end{cases} \right) \cdot S + S$
- ◯ $\frac{\partial Loss}{\partial s(X)} \cdot [s(X) \cdot (1 - s(X))] \cdot w_B \cdot \left(\begin{cases} 1 & E \geq 0 \\ 0 & E < 0 \end{cases} \right) \cdot S + S$
- ◯ None

(d) MesutBot is having trouble paying attention to the S feature because sometimes it gets zeroed out by the ReLU, so we connect it directly to the input of $s(\cdot)$ via the dotted edge. For the below, treat the dotted edge as a regular edge in the neural net.

(i) [1 pt] Which of the following is equivalent to $\frac{\partial Loss}{\partial w_A}$?

- ◯ $\frac{\partial Loss}{\partial s(X)} \cdot [s(X) \cdot (1 - s(X))] \cdot A$
- ◯ $2(s(X) - y^*) \cdot [s(X) \cdot (1 - s(X))] \cdot A$
- ◯ $\frac{\partial Loss}{\partial s(X)} \cdot [s(X) \cdot (1 - s(X))] \cdot 2A + A$
- ◯ $\frac{\partial Loss}{\partial s(X)} \cdot [s(X) \cdot (1 - s(X))] \cdot 2A$
- ◯ $2(s(X) - y^*) \cdot [s(X) \cdot (1 - s(X))] \cdot A + A$
- ◯ $\frac{\partial Loss}{\partial s(X)} \cdot [s(X) \cdot (1 - s(X))] \cdot A + A$
- ◯ None

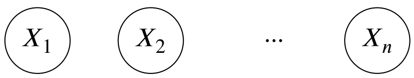
(ii) [1 pt] Which of the following is equivalent to $\frac{\partial Loss}{\partial w_S}$? Keep in mind we are still treating the dotted edge as a regular edge.

- ◯ $\frac{\partial Loss}{\partial s(X)} \cdot [s(X) \cdot (1 - s(X))] \cdot w_B \cdot \left(\begin{cases} 1 & E \geq 0 \\ 0 & E < 0 \end{cases} \right) \cdot S$
- ◯ $2(s(X) - y^*) \cdot [s(X) \cdot (1 - s(X))] \cdot w_B \cdot \left(\begin{cases} 1 & E \geq 0 \\ 0 & E < 0 \end{cases} \right) \cdot S$
- ◯ $\frac{\partial Loss}{\partial s(X)} \cdot [s(X) \cdot (1 - s(X))] \cdot w_B \cdot \left(\begin{cases} 1 & E \geq 0 \\ 0 & E < 0 \end{cases} \right) \cdot 2S + S$
- ◯ $\frac{\partial Loss}{\partial s(X)} \cdot [s(X) \cdot (1 - s(X))] \cdot w_B \cdot \left(\begin{cases} 1 & E \geq 0 \\ 0 & E < 0 \end{cases} \right) \cdot 2S$
- ◯ $2(s(X) - y^*) \cdot [s(X) \cdot (1 - s(X))] \cdot w_B \cdot \left(\begin{cases} 1 & E \geq 0 \\ 0 & E < 0 \end{cases} \right) \cdot S + S$
- ◯ $\frac{\partial Loss}{\partial s(X)} \cdot [s(X) \cdot (1 - s(X))] \cdot w_B \cdot \left(\begin{cases} 1 & E \geq 0 \\ 0 & E < 0 \end{cases} \right) \cdot S + S$
- ◯ None

Q7. [13 pts] Searching for a Bayes Network

Your friend gives you a joint distribution $P(X_1, X_2, ..., X_n)$ and wants you to find the structure of a Bayes Network that best represents this data. This can be formulated as a search problem.

Initial state:



Legal actions: Add an edge between any pair of nodes $X_i \rightarrow X_j$ so long as it does not violate the structure of a Bayes Network (remember, no cycles!).

Goal state: We don't know this but our friend does and can tell us if a given state is the goal state or not.

Answer the following questions regarding the search tree for this problem setup.

(a) [1 pt] Considering the set of legal actions, what is the branching factor of the search tree at depth 1? That is, the branching factor of the node representing the initial state. Write your answer in terms of the number of nodes, N.

☐

(b) [1 pt] Will any states be repeated in our search tree? ○ Yes ○ No

(c) Assume infinite computational resources, and that the goal state exists somewhere in the search tree.

(i) [1 pt] DFS is guaranteed to return the solution

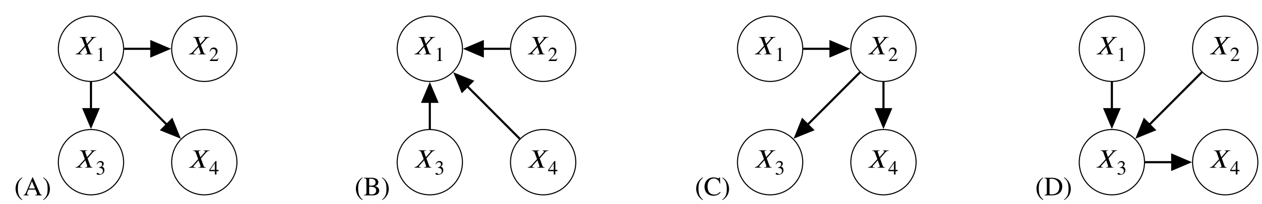
☐ if $N = 10000$ ☐ if $N = 10000$ and deleting edges are also legal actions ○ None of the above

(ii) [1 pt] BFS is guaranteed to return the solution

☐ if $N = 10000$ ☐ if $N = 10000$ and deleting edges are also legal actions ○ None of the above

Now your friend gives you the joint distribution $P(X_1, X_2, X_3, X_4)$ and wants you to run A* search to find the right structure. Recall that A* search expands states with the lowest estimated total cost, where total cost is equal to the sum of backward cost (sum of edge weights in the path to the state) and estimated forward cost (heuristic value).

(d) For the following heuristics, select the state (or multiple states, if there are ties) that would be expanded by A* search if the backward cost for all of the states are the same. **Assume that X_1, X_2, X_3 and X_4 are binary variables.**



(i) [1 pt] Number of nodes with no ancestors.

☐ A ☐ B ☐ C ☐ D

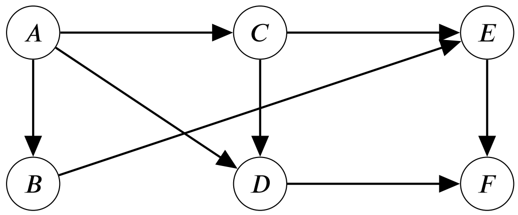
(ii) [1 pt] Number of parameters in Bayes Net representation.

☐ A ☐ B ☐ C ☐ D

(iii) [1 pt] Number of pairs of nodes that are not independent. Note that in each pair, the 2 nodes are unordered, i.e., the set of all pairs of nodes = $\{X_1X_2, X_1X_3, X_1X_4, X_2X_3, X_2X_4, X_3X_4\}$).

☐ A ☐ B ☐ C ☐ D

You found a Bayes Net that represents a joint distribution $P(A, B, C, D, E, F)$, as below.



A	P(A)
a_1	0.2
a_2	0.3
a_3	0.5

B	A	P(B|A)
b_1	a_1	0
b_2	a_1	1
b_1	a_2	0.4
b_2	a_2	0.6
b_1	a_3	0.9
b_2	a_3	0.1

C	A	P(C|A)
c_1	a_1	0.2
c_2	a_1	0.8
c_1	a_2	0.4
c_2	a_2	0.6
c_1	a_3	0.5
c_2	a_3	0.5

D	A	C	P(D|A, C)
d_1	a_1	c_1	0.7
d_2	a_1	c_1	0.3
d_1	a_2	c_1	0.4
d_2	a_2	c_1	0.6
d_1	a_3	c_1	0.8
d_2	a_3	c_1	0.2
d_1	a_1	c_2	0.6
d_2	a_1	c_2	0.4
d_1	a_2	c_2	0.5
d_2	a_2	c_2	0.5
d_1	a_3	c_2	0.9
d_2	a_3	c_2	0.1

E	B	C	P(E|B, C)
e_1	b_1	c_1	0.3
e_2	b_1	c_1	0.4
e_3	b_1	c_1	0.3
e_1	b_2	c_1	0.1
e_2	b_2	c_1	0.8
e_3	b_2	c_1	0.1
e_1	b_1	c_2	0.6
e_2	b_1	c_2	0.3
e_3	b_1	c_2	0.1
e_1	b_2	c_2	0.5
e_2	b_2	c_2	0.2
e_3	b_2	c_2	0.3

F	D	E	P(F|D, E)
f_1	d_1	e_1	0.5
f_2	d_1	e_1	0.5
f_1	d_2	e_1	0.8
f_2	d_2	e_1	0.2
f_1	d_1	e_2	0.6
f_2	d_1	e_2	0.4
f_1	d_2	e_2	0.4
f_2	d_2	e_2	0.6
f_1	d_1	e_3	0.1
f_2	d_1	e_3	0.9
f_1	d_2	e_3	0.3
f_2	d_2	e_3	0.7

(e) [1 pt] Starting with the original 6 CPTs, you eliminated D and got a factor. What is the size of the factor?

(f) You would like to find $P(c_1|e_1, f_1)$. For each ordering i of variable elimination, we denote $S(i)$ to be the size of the largest factor that gets generated during the variable elimination process following the ordering i.

(i) [1 pt] Among the 4 orderings below, select the ordering(s) i with the largest $S(i)$.
☐ A, B, D ☐ A, D, B ☐ B, A, D ☐ B, D, A

(ii) [1 pt] Among the 4 orderings below, select the ordering(s) i with the smallest $S(i)$.
☐ A, B, D ☐ A, D, B ☐ B, A, D ☐ B, D, A

(g) You'd like to try out sampling to find $P(c_1|e_1, f_1)$. For each question below, select the sampling method(s) that may generate the specified data point.

(i) [1 pt] $a_2, b_1, c_1, d_2, e_1, f_1$
☐ prior sampling ☐ rejection sampling ☐ likelihood sampling ☐ gibbs sampling ○ None

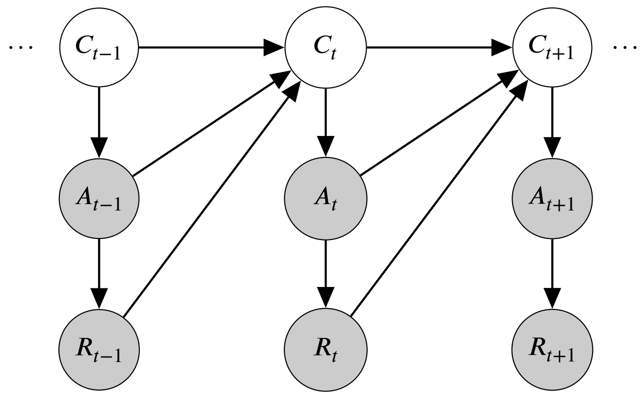
(ii) [1 pt] $a_1, b_2, c_2, d_2, e_3, f_1$
☐ prior sampling ☐ rejection sampling ☐ likelihood sampling ☐ gibbs sampling ○ None

(iii) [1 pt] $a_1, b_1, c_2, d_2, e_1, f_1$
☐ prior sampling ☐ rejection sampling ☐ likelihood sampling ☐ gibbs sampling ○ None

Q8. [11 pts] Particle Madness

Humans are finicky and have beliefs, and robots often get annoyed at dealing with us. Ideally, the human will have fully rational conscious beliefs that drive their actions.

(a) C represents the human's conscious beliefs, A represents human actions, and R represents reward distribution. Let's consider this model that makes the robots happy, in which actions and rewards directly influence conscious beliefs, and conscious beliefs influence actions.



(i) [1 pt] Select all variables that exhibit the Markov (memoryless) property in this model

☐ C ☐ A ☐ R ○ None

All variables are binary and have their probabilities specified in the table below. **Note the first table doesn't contain all the rows.**

C_{t+1}	A_t	R_t	C_t	$P(C_{t+1}\|C_t, A_t, R_t)$
+c	+a	+r	+c	.25
-c	+a	+r	+c	.75
+c	+a	+r	-c	.2
-c	+a	+r	-c	.8
+c	+a	-r	+c	.4
+c	-a	-r	-c	.9

A_t	C_t	$P(A_t\|C_t)$
+a	+c	.8
-a	+c	.2
+a	-c	.3
-a	-c	.7

R_t	A_t	$P(R_t\|A_t)$
+r	+a	.5
-r	+a	.5
+r	-a	.4
-r	-a	.6

We are trying to run the forward algorithm to obtain a belief distribution at time i, $B(C_i) = P(C_i|a_1, .., a_i, r_1, ..., r_i)$. We have $B(C_{i-1} = +c) = .4$ and $B(C_{i-1} = -c) = .6$ and we know that $A_i = +a$ and $R_i = +r$.

(ii) [1 pt]

If possible, calculate $B'(C_i = +c)$ where $B'(C_i) = P(C_i|a_1, .., a_{i-1}, r_1, ..., r_{i-1})$, the belief distribution after the time elapse update but before including the new evidence.

$B'(C_i = +c) =$ ☐ ○ Not Possible with the given information

(iii) [1 pt] Now it is time to calculate $B(C_i = +c)$ by performing the observation update. If possible, fill in the blank with a number that will be multiplied by $B'(C_i = +c)$ to create an expression that is equal to $B(C_i = +c)$ after normalization.

$$B(C_i = +c) = \frac{B'(C_i{=}+c) * \boxed{}}{B(C_i{=}+c) + B(C_i{=}+c)}$$ ○ Not Possible with the given information

(iv) [1 pt] Your friend claims that using particle filtering with 100 particles to estimate the belief distribution for this model will result in more accurate results than using the forward algorithm. Is your friend correct?

◯ Yes ◯ No

(v) [1 pt] Your friend also claims that using particle filtering with 100 particles to estimate the belief distribution for this model will require less computation and time than using the forward algorithm. Is your friend correct?

◯ Yes ◯ No

You decide to listen to your friend and run particle filtering. At time i you have a particle that is in state $+c$. You also know that $A_i = +a$, $R_i = -r$, $A_{i+1} = -a$, and $R_{i+1} = -r$.

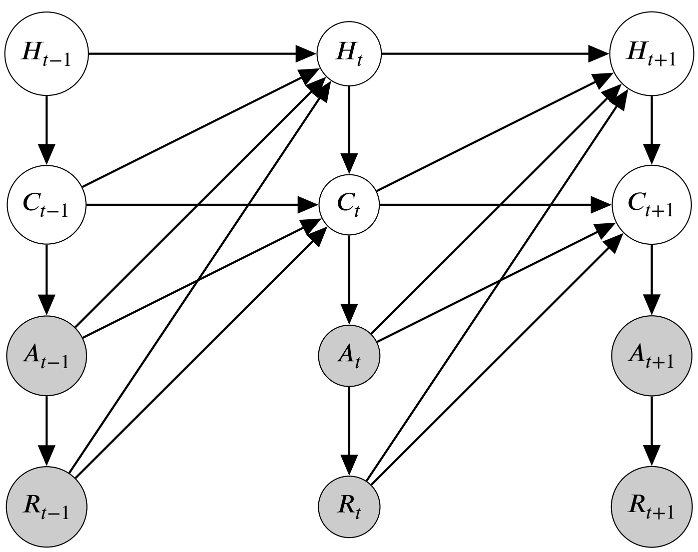
(vi) [1 pt] If possible, what is the probability that that particle is sampled into state $-c$ after the time elapse update?

◯ Not possible with the given information

(vii) [1 pt] If possible, assume that particle did get sampled into state $-c$. What is its weight during the observation update?

◯ Not possible with the given information

(b) Humans have unconscious preferences that can influence their beliefs. We incorporate another hidden Markov layer, where H_t represents unconscious preferences that influence C_t. Our revised model is shown below.



(i) [1 pt] From the model above, indicate which independence relations hold true.

- ☐ $R_t \perp\!\!\!\perp C_t | A_t$
- ☐ $C_{t+1} \perp\!\!\!\perp C_{t-1} | C_t, H_t$
- ☐ $C_{t+1} \perp\!\!\!\perp C_{t-1} | C_t$
- ☐ $C_{t+1} \perp\!\!\!\perp C_{t-1} | C_t, H_t, H_{t+1}$
- ☐ $C_{t+1} \perp\!\!\!\perp C_{t-1} | C_t, H_t, A_t$
- ☐ $C_{t+1} \perp\!\!\!\perp C_{t-1} | C_t, H_t, A_t, R_t$

The computation is difficult, so we use particle filtering with n particles. Particles can be defined in a few different ways. For the next three subparts, indicate whether the proposed particle definition is valid, which means it could be used to simulate a true distribution of the hidden variables while also incorporating observed A and R values.

(ii) [1 pt] $n/2$ of the particles code for H, and $n/2$ of the particles code for C.

◯ Yes ◯ No

(iii) [1 pt] All n particles code for an (H, C) pair.

◯ Yes ◯ No

(iv) [1 pt] All n particles code for H, and each particle generates an additional particle sampled only from $P(C_t | H_t, C_{t-1}, A_{t-1}, R_{t-1})$ to represent C only when needed

◯ Yes ◯ No