CS 188 Summer 2021 — Introduction to Artificial Intelligence — Final

- You have approximately 170 minutes.
- The exam is open book, open calculator, and open notes.
- For multiple choice questions,
  - ☐ means mark **all options** that apply
  - ○ means mark a **single choice**

First name	
Last name	
SID	

For staff use only:

Q1.	Potpourri	/20
Q2.	Model-Based RL with Function Approximation	/14
Q3.	Naive Bayes and Perceptron	/18
Q4.	Backpropagation with Activation Checkpointing	/18
Q5.	Ace King Queen	/16
Q6.	Pure Romance	/20
Q7.	Games	/18
Q8.	Hidden Markov Models and Particle Filtering	/15
	Total	/139

THIS PAGE IS INTENTIONALLY LEFT BLANK

Q1. [20 pts] Potpourri

(a) [3 pts] Which of the following statements are always true?

☐ $P(X,Y) = \sum_a P(X,a) \sum_b P(Y,b)$

■ $P(X) = \sum_a \sum_b \sum_c \sum_d P(X,a,b,c,d)$

☐ $P(X_1, X_2, ..., X_n) = P(X_1) \prod_{i=2}^{n} P(X_i | X_{i-1})$

☐ $P(X) \propto \sum_Y P(X|Y)$

☐ $P(X|Y) = \frac{P(X,Y)}{\sum_Y P(X,Y)}$

■ $P(X|y) \propto P(y|X)P(X)$

(b) [1 pt] Oski trains a neural network to classify whether or not a student is from Stanfurd. He notices that his classifier gets high accuracy when he tests it on his friends at Berkeley, but low accuracy when he visits Stanfurd. Which of the following is the best reason for why this is happening?

◯ Oski used a learning rate that was too low which led classifier to be stuck in a local minimum

◯ Oski added too much regularization when training his model

● Oski's training data has disproportionately more Berkeley examples than Stanfurd examples

◯ Oski is incorrectly calculating the accuracy

(c) [1 pt] Regina is trying to perform gradient descent on a function $f(x)$ using the following update rule:

$$x = x - \frac{\partial f}{\partial x}(x)$$

Is gradient descent guaranteed to converge to the global minimum for any $f(x)$?

◯ Yes, since she's updating using the gradient of x.

◯ Yes, but not for the reason above.

◯ No, since she is updating x in the wrong direction.

● No, but not for the reason above.

(d) [3 pts] Which of the following statements regarding VPI are always true?

☐ $VPI(E'|E=e) - VPI(F'|F=f) \geq 0$

■ $VPI(E'|E=e) * VPI(F'|F=f) \geq 0$

■ $VPI(E_h, E_i, E_j|E=e) = VPI(E_h|E=e) + VPI(E_i|E=e, E_h) + VPI(E_j|E=e, E_h, E_i)$

☐ $VPI(E_h, E_i, E_j|E=e) = VPI(E_h|E=e) + VPI(E_i|E=e) + VPI(E_j|E=e)$

☐ $VPI(E_j, E_k|E=e) = VPI(E_j|E=e) + VPI(E_k|E=e)$

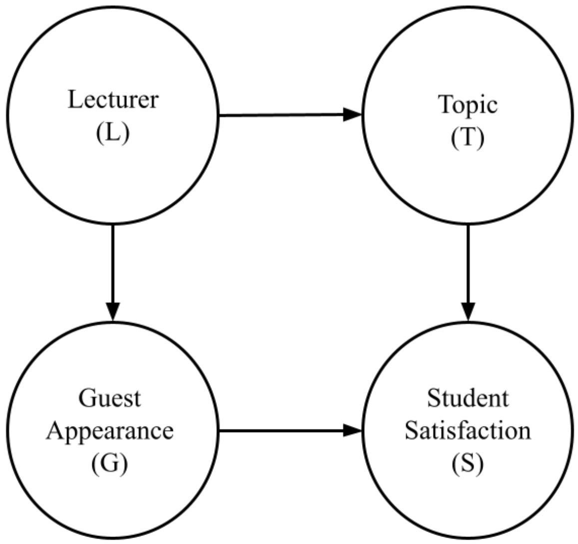
(e) [3 pts] Seth tries to generate samples using a modified version of prior sampling. Half of the time, he follows the normal prior sampling procedure. The other half of the time, he randomly generates a sample where he gives all of the variable assignments an equal chance. What is the probability of a certain sample following this procedure? Let the variables be $x_1, x_2, ..., x_n$, each of which can take on k possible values.

$$P(sample) = \frac{1}{2}(A)(B) + \frac{1}{2}(C)$$

- **(A):** ● $\prod_{i=1}^{n}$ ◯ $\sum_{i=1}^{n}$ ◯ $max_{i=1...n}$
- **(B):** ◯ $P(x_i)$ ● $P(x_i|parents(x_i))$ ◯ $P(x_i|children(x_i))$
- **(C):** ● $(\frac{1}{k})^n$ ◯ $(\frac{1}{n})^k$ ◯ $\frac{1}{n}$

(f) Although both great lecturers, Carl (Ca) and Mesut (Me) want to formalize this by seeing student satisfaction based off each of their lectures. They both lecture about two different topics, Reinforcement Learning (RL) or Game Trees (GT). They also both have different guest appearances in lecture, John Denero (JD), who Carl is closer to, and Carol Christ (CC), who Mesut is closer to.

P(L)	
Me	x
Ca	y



P(T|L)		
RL	Me	0.8
GT	Me	0.2
RL	Ca	0.1
GT	Ca	0.9

P(G|L)		
JD	Me	0.3
CC	Me	0.7
JD	Ca	0.9
CC	Ca	0.1

P(S|T,G)			
Sat	RL	JD	0.9
Unsat	RL	JD	0.1
Sat	RL	CC	0.5
Unsat	RL	CC	0.5
Sat	GT	JD	0.8
Unsat	GT	JD	0.2
Sat	GT	CC	0.5
Unsat	GT	CC	0.5

(i) [2 pts] What's the probability that the students are satisfied with a lecture where Carol Christ makes an appearance?

$P(Sat \mid CC) =$ $\boxed{0.5}$

By observation, you can see in the $P(S \mid T, G)$ table that all lectures where Carol Christ is the guest have a 0.5/0.5 chance of being satisfying or not satisfying.

(ii) [2 pts] Given that the students were satisfied with a lecture, what's the probability that the lecture was given by Carl? You should use $x = 0.5$ and $y = 0.5$ for this question if needed.

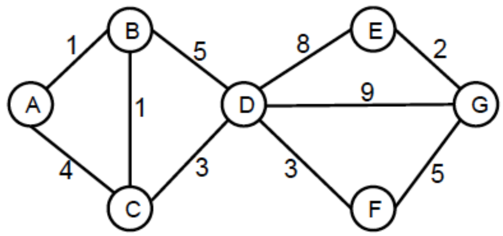
$P(Ca \mid Sat) =$ $\boxed{0.55}$

$P(Ca \mid Sat) = \frac{P(Ca \text{ and } Sat)}{P(Sat)}$

$P(Ca \text{ and } Sat) = 0.5 * (0.9 * (0.1 * 0.9 + 0.9 * 0.8) + 0.1 * (0.1 * 0.5 + 0.9 * 0.5)) = 0.3895$

$P(Me \text{ and } Sat) = 0.5 * (0.3 * (0.8 * 0.9 + 0.2 * 0.8) + 0.7 * (0.8 * 0.5 + 0.2 * 0.5)) = 0.307$

$P(Ca \mid Sat) = \frac{P(Ca \text{ and } Sat)}{P(Sat)} = \frac{0.3895}{0.6965} = 0.56$



Node	h_1	h_2
A	9.5	10
B	9	12
C	8	10
D	7	8
E	1.5	1
F	4	4.5
G	0	0

(g) Consider the state space graph shown above. A is the start state and G is the goal state. The costs for each edge are shown on the graph. Each edge can be traversed in both directions. Note that the heuristic h_1 is consistent but the heuristic h_2 is not consistent.

For each of the following graph search strategies (do not answer for tree search), mark which, if any, of the listed paths it could return. Note that for some search strategies the specific path returned might depend on tie-breaking behavior. In any such cases, make sure to mark all paths that could be returned under some tie-breaking scheme.

(i) [1 pt] DFS

■ A-B-D-G
■ A-C-D-G
■ A-B-C-D-F-G
☐ None of the above

(ii) [1 pt] BFS

■ A-B-D-G
■ A-C-D-G
☐ A-B-C-D-F-G
☐ None of the above

(iii) [1 pt] A^* Search with h_2

☐ A-B-D-G
☐ A-C-D-G
■ A-B-C-D-F-G
☐ None of the above

DFS can return any path. BFS will return all the shallowest paths, i.e. A-B-D-G and A-C-D-G. A-B-C-D-F-G is the optimal path for this problem, so that UCS and A^* using consistent heuristic h_1 will return that path. Although, h_2 is not consistent, it will also return this path.

Suppose you are completing the new heuristic function h_3 shown below. All the values are fixed except $h_3(B)$

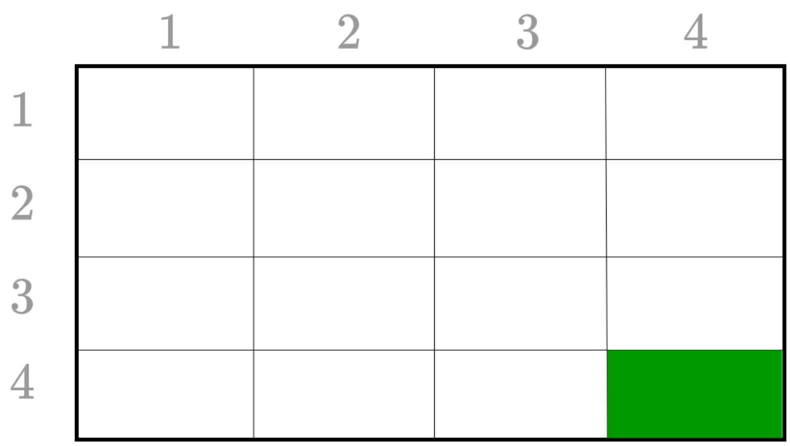
Node	A	B	C	D	E	F	G
h_3	10	?	9	7	1.5	4.5	0

(iv) [1 pt] What values of $h_3(B)$ make h_3 admissible?

$\alpha \leq h_3 \leq \beta$
$\alpha = 0$ $\beta = 12$

(v) [1 pt] What values of $h_3(B)$ make h_3 consistent?
$\gamma \le h_3 \le \lambda$
$\gamma = 9$ $\lambda = 10$

Q2. [14 pts] Model-Based RL with Function Approximation



Consider a robot navigating in the above grid world with walls around the edges as shown above. The robot's state is represented by $(row, column)$, with the starting position at $(1, 1)$

The robot cannot leave the grid. The robot can move into the wall, but its state will stay the same after the action. The allowed actions are $a \in \{\text{up}, \text{down}, \text{left}, \text{right}\}$ in all states except the sink state shaded in green In this sink state, only the exit action is available, which takes the agent to a terminal state in which it can no longer take actions or receive rewards.

When the robot transitions into the terminal state, it receives a reward of 10. For all other states, the robot receives a living reward of -1 for transitioning into the state.

The robot wants to learn a policy to maximize its reward, but unfortunately does not know the transition model of the MDP exactly. However, the robot does know that any given state s, there is an associated (unknown) probability s_p that any action it takes will be flipped (eg. up becomes down, right becomes left).

(a) [3 pts] To estimate the MDP transition model, the robot executes some policy π in the grid world and collects a set of N transitions $\{(s_i, a_i, s_i', r_i)\}_{i=1}^N$. The robot then decides to use this to create a dataset $\mathcal{D} = \{(s_i, y_i)\}_{i=1}^N$ of states in which action flips occur, where $y_i = 1$ if a flip happened at s_i and -1 otherwise.

2 The robot could use the number of action flips at each state to estimate the empirical probability of an action flip at each individual state. However, the robot is also considering training a logistic regression classifier $g : \mathcal{S} \mapsto \{0, 1\}$ which uses $\mathcal{D}$ to learn whether an action flip will occur at a given state. Which of the following is a possible advantage of learning a classifi er? Select all that apply.

- [x] The classifier will use less memory if the state space dimension is much lower than the number of possible states
- [x] The classifier will give us information about action flip probabilities even on states not visited in $\mathcal{D}$.
- [ ] The classifier will always be more accurate.
- [x] The classifier will be able to reuse information about the probability of action flips across different states.

(b) [2 pts] You decide to use logistic regression to estimate the probability of an action flip at a given state ($P(y_i|s_i, w)$). Define the logistic function $\phi(z) = \frac{1}{1+e^{-z}}$. Which of the following is a correct expression for $P(y_i|s_i, w)$?

○ $1 + \phi(y_i w \cdot s_i)$
○ $1/\phi(y_i w \cdot s_i)$
○ $1/\phi(1 - y_i w \cdot s_i)$
○ $1/\phi(1 + y_i w \cdot s_i)$
● $\phi(y_i w \cdot s_i)$
○ $\phi(1 + y_i w \cdot s_i)$

(c) Suppose the robot now wants to update the weights for logistic regression with gradient ascent to maximize the likelihood of the N transitions in $\mathcal{D}$.

Directly maximizing the likelihood of transitions in $\mathcal{D}$ gives the following udpate:

$$w \leftarrow w + \eta \nabla_w \prod_{i=1}^{N} P(y_i|s_i, w)$$

Suppose the dataset has $N = 64$ transitions and we are computing our weights on a computer which can only store variables x in memory if $|x| > 10^{-32}$.

(i) [2 pts] If $P(y_i|s_i, w) \leq \alpha \; \forall y_i, s_i$, what is the largest value of α such that $\prod_{i=1}^{N} P(y_i|s_i, w)$ will **not** fit in memory?

$$\alpha^N = 10^{-32} \implies \alpha = 1/\sqrt{10} = 0.31$$

(ii) [2 pts] If $N = 64$ and $\alpha = 0.1$, will value of $\prod_{i=1}^{N} P(y_i|s_i, w)$ fit in memory?

◯ Yes
● No

$$|0.1^{64}| < 10^{-32}$$

(d) Concerned about memory issues, you decide to instead maximize the log-likelihood of the transitions in $\mathcal{D}$ as follows:

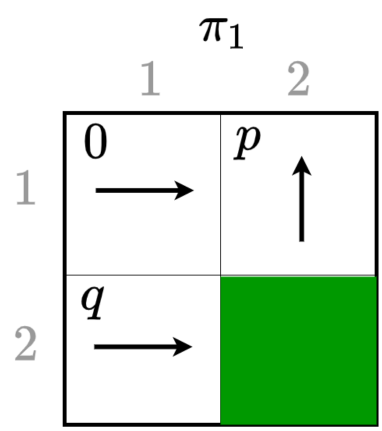
$$w \leftarrow w + \eta \nabla_w \sum_{i=1}^{N} \log P(y_i|s_i, w)$$

For this question, assume that log is in base 10.

(i) [2 pts] For $N = 64$ and $\alpha = 0.1$, will $\sum_{i=1}^{N} \log P(y_i|s_i, w)$ fit in memory?

● Yes
◯ No

$$|-64| > 10^{-32}$$



(e) Having estimated w by running logistic regression, we can now use the resulting classifier to estimate the value of different policies in the environment. For this problem, consider a simplified 2×2 grid world where the action flip probability is included in the top left of each grid cell. For policy π_1 illustrated above:

(i) [3 pts] What is $V^{\pi_1}((1,1))$ if $p = q = 0.3$? Here p and q are the flip probabilities (probabilities of choosing the opposite action) for the respective states.

We have the following system of equations for V^{π_1}:

$$\begin{aligned}
V^{\pi_1}((1,1)) &= -1 + V^{\pi_1}((1,2)) \\
V^{\pi_1}((1,2)) &= -1 + pV^{\pi_1}((2,2)) + (1-p)V^{\pi_1}((1,2)) \\
V^{\pi_1}((2,1)) &= -1 + (1-q)V^{\pi_1}((2,2)) + qV^{\pi_1}((2,1)) \\
V^{\pi_1}((2,2)) &= 10
\end{aligned}$$

Solving the above system gives:

$$\begin{aligned}
V^{\pi_1}((1,2)) &= -1 + 10p + (1-p)V^{\pi_1}((1,2)) \implies \\
pV^{\pi_1}((1,2)) &= 10p - 1 \implies \\
V^{\pi_1}((1,2)) &= \frac{10p-1}{p} \implies \\
V^{\pi_1}((1,1)) &= -1 + V^{\pi_1}((1,2)) \implies \\
V^{\pi_1}((1,1)) &= \frac{9p-1}{p} = \frac{9 * 0.3 - 1}{0.3} = 5.67
\end{aligned}$$

Q3. [18 pts] Naive Bayes and Perceptron

Pacman has received a ton of spam lately. He decides to use some machine learning techniques to filter his emails.

(a) Pacman first tries using Naive Bayes. For some reason, he chooses the words "buy", "discount", and "dollar" as features during classification. Below is the training dataset:

"buy" (W_1)	"discount" (W_2)	"dollar" (W_3)	label (E)
1	1	0	spam
1	0	1	spam
1	0	0	spam
0	1	0	spam
0	0	0	ham
0	1	1	ham

Under the assumptions of Naive Bayes, work out the following probabilities.

(i) [2 pts] $\mathbb{P}(E = ham) =$ $\frac{1}{3}$ $\quad \mathbb{P}(E = spam) =$ $\frac{2}{3}$

(ii) [4 pts] $\mathbb{P}(W_1 = 1 | E = spam) =$ $\frac{3}{4}$ $\quad \mathbb{P}(W_1 = 0 | E = ham) =$ $\frac{1}{1}$

$\mathbb{P}(W_3 = 1 | E = spam, W_2 = 0) =$ $\frac{1}{4}$ $\quad \mathbb{P}(W_2 = 1, W_3 = 0 | E = ham) =$ $\frac{1}{4}$

Notice that it's under the assumptions of Naive Bayes, which is different from the normal "inference by enumeration" technique for the last two questions.

(iii) [2 pts] After filling out the probability table, Pacman found that one probability gets value zero, so he decided to use Laplace smoothing with $k = 1$. However, his roommate said that it's better to use $k = 2$. What k value should Pacman choose?

- ○ He should pick the value that works best in training data
- ● He should pick the value that works best in validation data
- ○ He should pick the value that works best in testing data
- ○ He should pick the average number of samples per class, which is 3 in this case.

(iv) [3 pts] Right after implementing Laplace smoothing with $k = 1$, Pacman receives an email which includes all three feature words. How would the model classify this email?

$\mathbb{P}(E = spam, W_1 = 1, W_2 = 1, W_3 = 1) =$ $\frac{5}{72}$

$\mathbb{P}(E = ham, W_1 = 1, W_2 = 1, W_3 = 1) =$ $\frac{3}{128}$

○ ham ● spam

$$\mathbb{P}(E = spam) = \frac{4+1}{6+2} = \frac{5}{8} \qquad \mathbb{P}(E = ham) = \frac{2+1}{6+2} = \frac{3}{8}$$

$$\mathbb{P}(W_1 = 1 | E = spam) = \frac{3+1}{4+2} = \frac{2}{3} \qquad \mathbb{P}(W_1 = 1 | E = ham) = \frac{0+1}{2+2} = \frac{1}{4}$$

$$\mathbb{P}(W_2 = 1 | E = spam) = \frac{2+1}{4+2} = \frac{1}{2} \qquad \mathbb{P}(W_2 = 1 | E = ham) = \frac{1+1}{2+2} = \frac{1}{2}$$

$$\mathbb{P}(W_3 = 1 | E = spam) = \frac{1+1}{4+2} = \frac{1}{3} \qquad \mathbb{P}(W_3 = 1 | E = ham) = \frac{1+1}{2+2} = \frac{1}{2}$$

$$\mathbb{P}(E = spam, W_1 = 1, W_2 = 1, W_3 = 1) = \frac{5}{8}\frac{2}{3}\frac{1}{2}\frac{1}{3} = \frac{5}{72} \qquad \mathbb{P}(E = ham, W_1 = 1, W_2 = 1, W_3 = 1) = \frac{3}{8}\frac{1}{4}\frac{1}{2}\frac{1}{2} = \frac{3}{128}$$

$$\hat{E} = \arg\max_e \mathbb{P}(E = e, W_1 = 1, W_2 = 1, W_3 = 1) = spam$$

(b) Pacman is unhappy with the performance of Naive Bayes, so he decided to switch to using a Linear Perceptron with bias, with new features: number of "$" sign, and number of digits. Below is the training dataset:

number of "\$" sign (n_1)	number of digits (n_2)	label (E)
7	3	spam (1)
0	2	ham (-1)
5	5	spam (1)
1	1	ham (-1)

(i) [2 pts] Pacman is confident that the Perceptron will correctly classify all incoming emails after it converges. Is that true?

- ○ Yes, because the training data is linearly separable
- ○ Yes, but not the reason above
- ○ No, because no decision boundary that goes through the origin can separate the data
- ● No, but not the reason above

The Perceptron might not correctly classify unseen data.

(ii) [3 pts] Starting from initial weight $w = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix}$ (the last entry being the bias weight), determine the weight for the first few iterations. **Leave all answers in form of "[a,b,c]"**

After seeing the first data: $\begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} + \begin{bmatrix} 7 \\ 3 \\ 1 \end{bmatrix} = \begin{bmatrix} 7 \\ 3 \\ 0 \end{bmatrix}$

After seeing the second data: $\begin{bmatrix} 7 \\ 3 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 7 \\ 1 \\ -1 \end{bmatrix}$

After seeing the third data: $\begin{bmatrix} 7 \\ 1 \\ -1 \end{bmatrix}$

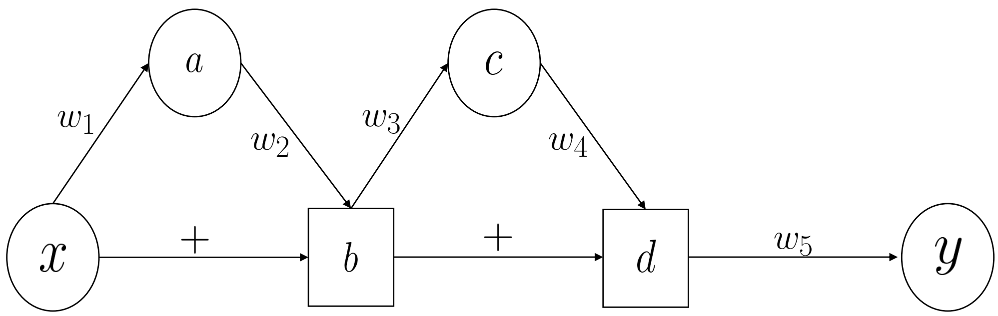
(iii) [2 pts] To decouple from the above, suppose Pacman now uses a Linear Perceptron **without bias**, and the current weight is $\begin{bmatrix} \frac{1}{2} \\ -\frac{1}{2} \end{bmatrix}$, which is in the fourth quadrant. The next training example has 1 occurrence of "\$" sign, and 1 occurrence of digits. Which quadrant could the weight be in after training using this sample?

- ■ The first quadrant
- □ The second quadrant
- ■ The third quadrant
- ■ The fourth quadrant
- □ The x or y axis

If the sample is classified correctly, then the weight would still be in the fourth quadrant. Otherwise, if the sample is a ham, then the weight would be subtracted by [1,1] and end up in the third quadrant. If the sample is a spam, then the weight would be added to [1,1], and end up in the first quadrant.

Q4. [18 pts] Backpropagation with Activation Checkpointing

Below is a neural network with residual connections (square nodes) whose weights are w_1, w_2, w_3, w_4, w_5. The neural network takes x as input and outputs y.



The outputs at each node are computed as the following:

$o_a = \text{ReLU}(z_a)$ where $z_a = x \cdot w_1$
$o_b = \text{ReLU}(z_b) + x$ where $z_b = o_a \cdot w_2$
$o_c = \text{LeakyReLU}(z_c)$ where $z_c = o_b \cdot w_3$
$o_d = \text{LeakyReLU}(z_d) + o_b$ where $z_d = o_c \cdot w_4$
$y = o_d * w_5$

Let $\text{ReLU}(z) = \max(z, 0)$ while $\text{LeakyReLU}(z) = \begin{cases} z, & \text{if } z > 0 \\ \gamma * z, & \text{otherwise} \end{cases}$

Suppose the network has input $x = 2$ and $\gamma = 0.1$.
The weight values are $w_1 = 1$, $w_2 = 2$, $w_3 = 1$, $w_4 = -5$, $w_5 = 3$

(a) [2 pts] Perform forward propagation on the neural network.

$o_a =$ 2
$o_b =$ 6
$o_c =$ 6
$o_d =$ 3
$y =$ 9

(b) [3 pts] Run backpropagation to calculate the following partial derivatives. Express the values of partial derivatives using only input (x), activations (o_a, o_b, o_c, o_d), and constants. Do not write as a single number (must be an expression using x and/or o_i, $i \in \{a, b, c, d\}$).

Input o_a, o_b, o_c, o_d as "o_a", "o_b", "o_c", "o_d" respectively.

$\frac{\partial y}{\partial w_5} =$ o_d

$\frac{\partial y}{\partial w_4} =$ $0.3 \cdot o_c$

$\frac{\partial y}{\partial w_3} =$ $-1.5 \cdot o_b$

$\frac{\partial y}{\partial w_2} =$ $1.5 \cdot o_a$

$\frac{\partial y}{\partial w_1} =$ $3 \cdot x$

$$\frac{\partial y}{\partial w_5} = \frac{\partial(o_d \cdot w_5)}{\partial w_5}$$
$$= o_d$$

$$\frac{\partial y}{\partial w_4} = \frac{\partial o_d}{\partial w_4}\frac{\partial y}{\partial o_d}$$
$$= \frac{\partial(\gamma \cdot o_c \cdot w_4 + o_b)}{\partial w_4}\frac{\partial(o_d \cdot w_5)}{\partial o_d}$$
$$= \gamma \cdot o_c \cdot w_5$$
$$= 0.3 \cdot o_c$$

$$\frac{\partial y}{\partial w_3} = \frac{\partial o_c}{\partial w_3}\frac{\partial y}{\partial o_c}$$
$$= \frac{\partial o_c}{\partial w_3}\frac{\partial o_d}{\partial o_c}\frac{\partial y}{\partial o_d}$$
$$= o_b \cdot (\gamma \cdot w_4) \cdot w_5$$
$$= -1.5 \cdot o_b$$

$$\frac{\partial y}{\partial w_2} = \frac{\partial o_b}{\partial w_2}\frac{\partial y}{\partial o_b}$$
$$= \frac{\partial o_b}{\partial w_2}\frac{\partial o_d}{\partial o_b}\frac{\partial y}{\partial o_d}$$
$$= o_a \cdot (\gamma \cdot w_4 \cdot w_3 + 1) \cdot w_5$$
$$= 1.5 \cdot o_a$$

$$\frac{\partial y}{\partial w_1} = \frac{\partial o_a}{\partial w_1}\frac{\partial y}{\partial o_a}$$
$$= \frac{\partial o_a}{\partial w_1}\frac{\partial o_b}{\partial o_a}\frac{\partial y}{\partial o_b}$$
$$= x \cdot w_2 \cdot (\gamma \cdot w_4 \cdot w_3 + 1) \cdot w_5$$
$$= 3 \cdot x$$

(c) [1 pt] Let's say storing the value of a single activation o_i, $i \in \{a, b, c, d\}$) costs 1 memory unit. What is the maximum number of memory units used while running the backpropagation above? ____4____
All activations must be known to compute all the partial derivatives. This means that a maximum of 4 memory units had to be used.

(d) As we try to train this neural network, we get an out-of-memory error and it turns out the main culprit is the cost of storing activations. To address this issue, we explore **activation checkpointing** where *only a subset of activations are stored/checkpointed during forward propagation*. This means that we may need to re-run parts of the forward propagation in order to re-compute activations that are missing, but needed during backpropagation.

During backpropagation, you may *use more memory units* to re-compute and store additional activations needed for computing a particular partial derivative, but they *must be released* either when they are no longer needed for computing that partial derivative or if that partial derivative is successfully computed. Checkpointed activations are never released.

As an example, suppose o_a is stored/checkpointed after forward propagation (memory: $[o_a]$) and we are interested in knowing the partial derivative which can be computed using just o_c. The neural network re-computes o_b based on x and o_a (memory: $[o_a, o_b]$). Then, the neural network uses o_b to re-compute o_c (memory: $[o_a, o_b, o_c]$). o_b is released as it's no longer needed in computing the partial derivative of interest (memory: $[o_a, o_c]$). Once the partial derivative in interest is computed using o_c, o_c is released (memory: $[o_a]$).

Now, suppose we only checkpointed o_b during forward propagation.

(i) [1 pt] Can you compute $\frac{\partial y}{\partial w_5}$?

- ◯ Yes, and without needing any additional computation than what is required in the vanilla backpropagation (without activation checkpointing).
- ● Yes, but requiring more computation than what is used in the vanilla backpropagation (without activation checkpointing).
- ◯ No

Computing this partial derivative requires the value of o_d. Since we only stored o_b, we need to re-compute o_c using o_b, and then use o_b and o_c to re-compute o_d. So you can compute the partial derivative, but need extra computation.

(ii) [1 pt] Can you compute $\frac{\partial y}{\partial w_4}$?

- ◯ Yes, and without needing any additional computation than what is required in the vanilla backpropagation (without activation checkpointing).
- ● Yes, but requiring more computation than what is used in the vanilla backpropagation (without activation checkpointing).
- ◯ No

Computing this partial derivative requires the value of o_c. Since we only stored o_b, we need to re-compute o_c using o_b. So you can compute the partial derivative, but need extra computation.

(iii) [1 pt] Can you compute $\frac{\partial y}{\partial w_3}$?

- ● Yes, and without needing any additional computation than what is required in the vanilla backpropagation (without activation checkpointing).
- ◯ Yes, but requiring more computation than what is used in the vanilla backpropagation (without activation checkpointing).
- ◯ No

Computing this partial derivative requires the value of o_b, which we already have. So you can compute the partial derivative without needing additional computation.

(iv) [2 pts] What is the maximum number of memory units used while computing the above partial derivatives? If a partial derivative cannot be computed, assume no additional memory unit was used. ____3____

In the beginning of backpropagation, o_b is the only activation that is stored (memory: $[o_b]$).

To compute $\frac{\partial y}{\partial w_5}$, o_d is required, so we re-compute o_c first (memory: $[o_b, o_c]$). Using o_c and o_b, we re-compute o_d (memory: $[o_b, o_c, o_d]$). Since o_c is no longer needed, it is released (memory: $[o_b, o_d]$). After computing $\frac{\partial y}{\partial w_5}$, o_d is released (memory: $[o_b]$).

To compute $\frac{\partial y}{\partial w_4}$, we re-compute o_c (memory: $[o_b, o_c]$). Once $\frac{\partial y}{\partial w_4}$ is computed, o_c is released (memory: $[o_b]$).

Finally, to compute $\frac{\partial y}{\partial w_3}$ we use o_b (memory: $[o_b]$).

As a result, the maximum number of memory units used is 3.

Now, suppose we only checkpoint o_c during forward propagation.

(v) [1 pt] Can you compute $\frac{\partial y}{\partial w_5}$?

- ◯ Yes, and without needing any additional computation than what is required in the vanilla backpropagation (without activation checkpointing).
- ● Yes, but requiring more computation than what is used in the vanilla backpropagation (without activation checkpointing).
- ◯ No

Computing this partial derivative requires the value of o_d. Since we only stored o_c, we need to re-compute o_a, and then o_b. Using o_b and o_c, o_d is re-computed. So you can compute the partial derivative, but need extra computation.

(vi) [1 pt] Can you compute $\frac{\partial y}{\partial w_4}$?

- ● Yes, and without needing any additional computation than what is required in the vanilla backpropagation (without activation checkpointing).
- ○ Yes, but requiring more computation than what is used in the vanilla backpropagation (without activation checkpointing).
- ○ No

Computing this partial derivative requires the value of o_c which is checkpointed. So you can compute the partial derivative without extra computation.

(vii) [1 pt] Can you compute $\frac{\partial y}{\partial w_3}$?

- ○ Yes, and without needing any additional computation than what is required in the vanilla backpropagation (without activation checkpointing).
- ● Yes, but requiring more computation than what is used in the vanilla backpropagation (without activation checkpointing).
- ○ No

Computing this partial derivative requires the value of o_b. We need to re-compute o_a and then use it to re-compute o_b. So you can compute the partial derivative, but need additional computation.

(viii) [2 pts] What is the maximum number of memory units used while computing the above partial derivatives? If a partial derivative cannot be computed, assume no additional memory unit was used. ____3____

In the beginning of backpropagation, o_c is the only activation that is stored (memory: $[o_c]$).

To compute $\frac{\partial y}{\partial w_5}$, o_d is required. We re-compute o_a first (memory: $[o_a, o_c]$). Using x and o_a, we re-compute o_b (memory: $[o_a, o_b, o_c]$). Since o_a is no longer needed, it is released (memory: $[o_b, o_c]$). Re-compute o_d using o_b and o_c (memory: $[o_b, o_c, o_d]$). o_b is released as it's no longer needed (memory: $[o_c, o_d]$). After computing $\frac{\partial y}{\partial w_5}$, o_d is released (memory: $[o_c]$).

To compute $\frac{\partial y}{\partial w_4}$, we use the checkpointed o_c (memory: $[o_c]$).

Finally, to compute $\frac{\partial y}{\partial w_3}$, o_b is required. We re-compute o_a first (memory: $[o_a, o_c]$). Using x and o_a, we re-compute o_b (memory: $[o_a, o_b, o_c]$). Since o_a is no longer needed, it is released (memory: $[o_b, o_c]$). After computing $\frac{\partial y}{\partial w_3}$ using o_b, it is released (memory: $[o_c]$).

As a result, the maximum number of memory units used is 3.

(ix) [2 pts] Which one is a better checkpoint between o_b and o_c?

- ○ Both are the same.
- ○ o_b because it has a lower peak memory usage.
- ○ o_c because it has a lower peak memory usage.
- ● o_b because it requires less additional compute.
- ○ o_c because it requires less additional compute.

The peak memory usage for both activation checkpoints are the same (3), yet checkpointing o_c requires more additional compute because, for instance, computing o_d to get $\frac{\partial y}{\partial w_5}$ involves re-running the forward propagation all the way from the input node.

Q5. [16 pts] Ace King Queen

Your friend, Trevor, proposes a simplified game of poker with three cards - an Ace, a King, and Queen (best to worst in that order).

In the game, each player gets a card (drawn from the 3 total cards without replacement) face down and puts in a mandatory $10 ante. The first player can either "bet" (put in another) $20 or "check" which ends the game (causing the player with the better card to take the $20 in the middle).

If the first player bets, the second player can either "call", matching first player's bet (and have the player with the highest card win the now $60 pot), or "fold" (let the first player win without seeing his card).

An example round would be the first player drawing an Ace and second player drawing a King. The first player could either "bet" or "check". Say the first player "bets", then the second player could either "call" or "fold". If the second player "calls" he would lose $30 to the first player (they would reveal cards and first player would have a higher card).

(a) Now, let's model this game as a Bayes net.

(i) [2 pts] Is the first player's card independent from the second player's card?

◯ Yes, because they are drawn separately.
◯ Yes, but for another reason.
◯ No, because the card may affect the player's strategy.
● No, but for another reason.

Knowing the first player's card eliminates the second player from drawing it and affect their card distribution.

(ii) [1 pt] Is the first player's card independent from the second player's card given the last (third) card in the deck?

◯ Yes
● No

Knowing the last card doesn't eliminate the dependence.

(b) Suppose you're the second player, and the opponent has just bet, so now it's your turn to decide on an action.

Furthermore, you know that the opponent will bet a third of his Queens, two-thirds of his Kings, and all of his Aces.

(i) [1 pt] If you have a Queen, what is the expected utility of "calling" in your spot?

EU = -30

You have a 100% chance of losing as the second player with a Queen. By "calling" you lose an extra $20 on top of the ante.

(ii) [1 pt] If you have a Queen, what is the expected utility of "folding" in your spot?

EU = -10

Same as above except "folding" stops you from losing the extra $20.

(iii) [1 pt] Say you have a Queen as the second player, and the first player bets, what is the optimal move?

● Fold
◯ Call

Action with MEU is "Fold" as seen from parts i) and ii).

(iv) [1 pt] If you now have an Ace, what is the expected utility of "calling" in your spot?

EU = 30

Same as queen, except you now always win.

(v) [1 pt] What is the expected utility of the game for the first player if he chooses to always check?

Value = 0

$EU = P(\text{having a higher card}) * \$10 + (1 - P(\text{having a higher card})) * -\$10 = \frac{1}{2} * \$10 + \frac{1}{2} * -\$10 = 0$. The game is symmetric if no betting happens.

(c) Suppose you're the second player with a King, and the opponent has just bet, so now it's your turn to decide on an action.

You still know that the opponent will bet a third of his Queens, two-thirds of his Kings, and all of his Aces.

(i) [1 pt] What is probability that your opponent is holding an Ace?

P = $\frac{3}{4}$

$P(\text{opponent has an Ace}|\text{opponent bet, we have a King}) = \frac{\frac{1}{3}*\frac{1}{2}*1}{\frac{1}{3}*\frac{1}{2}*1+\frac{1}{3}*\frac{1}{2}*\frac{1}{3}} = \frac{3}{4}$

(ii) [1 pt] What is probability that your opponent is holding an King?
P = ____0____
The opponent cannot hold a King if you're holding one.

(iii) [1 pt] What is probability that your opponent is holding an Queen?
P = ____$\frac{1}{4}$____
$1 - P(\text{opponent has an Ace}|\text{opponent bet, we have a King}) = 1 - \frac{3}{4} = \frac{1}{4}$

(iv) [3 pts] Now, the dealer tells you that if you pay him, he will tell you what card your opponent is holding. The dealer never lies.
What is the maximum expected utility of this state if you were to find out your opponent's card?
MEU = ____0____
If we know the opponent's card, we can act optimally, which will be folding if they have an Ace and calling if they have a Queen.
MEU $= -\$10 * P(\text{opponent has an Ace}|\text{opponent bet, we have a King}) + \$30 * P(\text{opponent has a Queen}|\text{opponent bet, we have a}$
$-10\frac{3}{4} + 30\frac{1}{4} = 0$

(v) [2 pts] Suppose the dealer doesn't always tell you the truth, even if you pay him. Can we still model how much to pay him with value of perfect information?
● Yes
○ No
You can use an additional node to represent the information that the dealer tells you and model it's relation to the true value.

(d) Bonus! Only attempt if you have extra time. Worth 0 points. Not on the examtool for confusion reasons.

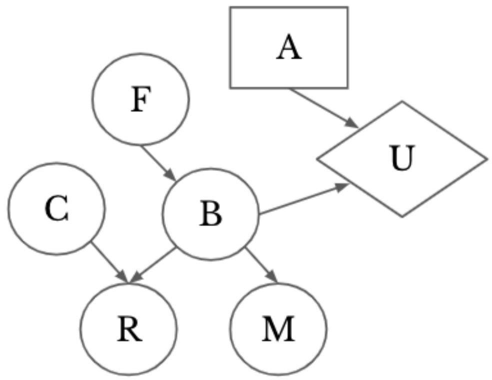
(i) [0 pts] As the first player, with what probability should you bet when holding a King?
P = ____0____

(ii) [0 pts] As the first player, with what probability should you bet when holding a Queen?
P = ____0____

(iii) [0 pts] As the mandatory contribution to the pot increases, how does the optimal strategy with a Queen change as the first player?
○ Bet more
● Check more
○ Optimal strategy doesn't change

Q6. [20 pts] Pure Romance

Andy's got a problem: he has a crush on Brianna and doesn't know if he should ask her out. One day, Andy watches lecture and realizes something — he can model his current worry as a decision net! He drew up the following net with utility function $U(A, B)$:



- $A \in \{yes, no\}$: Whether Andy asks Brianna out
- B: How Brianna feels towards Andy
- F: How Brianna's best friend thinks of Andy
- M: Brianna's mood when Andy is around her
- R: The time it takes for Brianna to reply to Andy's messages
- C: The courseload that Brianna is taking this semester

(a) For the following relations, indicate whether it is always, sometimes, or never true.

(i) [2 pts] $VPI(R|M) > 0$

○ Always true
● Sometimes true
○ Never true

There is a possibility that knowing R after knowing M will increase the MEU.

(ii) [2 pts] $VPI(C) <= 0$

● Always true
○ Sometimes true
○ Never true

VPI of C alone is always equal to 0, since C is independent of B when R is not known.

(iii) [2 pts] $VPI(F, B) > VPI(B)$

○ Always true
○ Sometimes true
● Never true

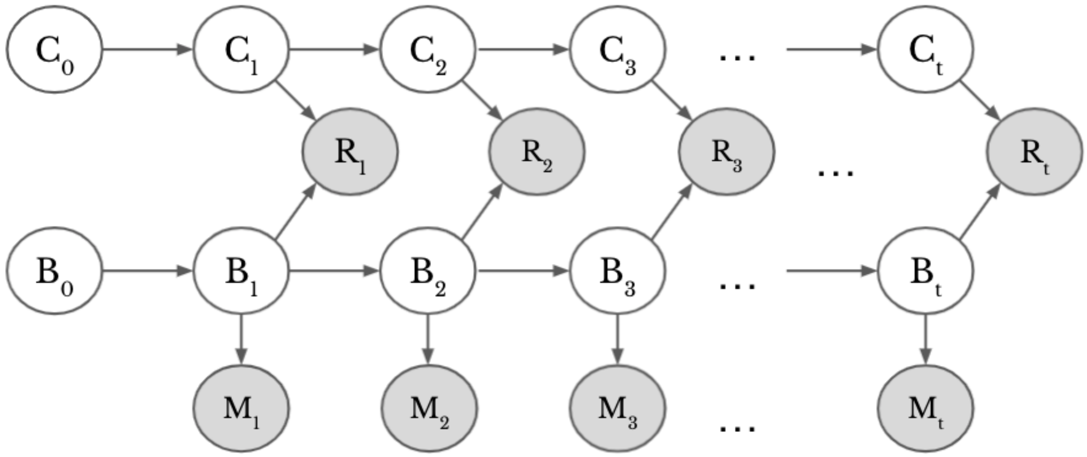
Knowing B already gives us the maximum information, since it is the only non-action input into the utility node. So VPI(F, B) will always be exactly equal to VPI(B)

(iv) [2 pts] $VPI(R, M|B) = VPI(R|B) + VPI(M|B)$

● Always true
○ Sometimes true
○ Never true

Though VPI usually is not additive, it is in this case because both $VPI(R|B)$ and $VPI(M|B)$ are zero (because of the reasoning in the subquestion above)

(b) Andy realizes he can observe R and M. He's not sure that his estimates of how Brianna feels about him (B) and her courseload (C) are close to reality, and so he decides to take his time to estimate these two variables using a Hidden Markov Model. He decides to disregard F in his HMM.



He plans to spend t days collecting evidence, and use his belief of B and C on day t to inform his decision net.

(i) [2 pts] How should Andy solve for these beliefs?

- ○ Particle filtering, because particle weights are useful in decision nets.
- ○ Particle filtering, because we don't know how long the time horizon t will be.
- ○ Exact inference, since it is always more accurate than particle filtering
- ○ Exact inference if the time horizon t is short and particle filtering if the time horizon t is long
- ● Unable to answer with the information provided

We don't know the size of the domain of the state variables. B could have a tiny domain , for example like, dislike. Or it could have an extremely large number of possible values.

(ii) [2 pts] We want to develop a model for this HMM. Which of the following are equivalent to an observation model $P(M_i, R_i|B_i, C_i)$, where $3 \leq i \leq t$?

- ■ $P(M_i|B_i)P(R_i|B_i, C_1)$
- □ $P(M_i|B_i)P(M_i|C_i)P(R_i|B_i)P(R_i|C_i)$
- ■ $P(M_i|B_i, C_i)P(R_i|B_i, C_i)$
- □ $P(M_i|B_i)P(M_i|C_i)P(R_i|B_i)$
- ■ $P(M_i|R_i, B_i, C_i)P(R_i|B_i, C_i)$
- □ None of the above

$P(M_i, R_i|B_i, C_i) = P(M_i|R_i, B_i, C_i)P(R_i|B_i, C_i)$ through chain rule, and the other two options come from applying conditional independences.

(iii) [2 pts] Which of the following are equivalent to $P(B_i, C_i|B_{i-1}, C_{i-1}, R_{i-1})$, where $3 \leq i \leq t$?

- ■ $P(B_i|B_{i-1}, R_{i-1})P(C_i|C_{i-1}, R_{i-1})$
- ■ $P(B_i|B_{i-1}, R_{i-1})P(C_i|B_i, C_{i-1}, R_{i-1})$
- ■ $P(B_i|B_{i-1}, C_i, R_{i-1})P(C_i|B_i, C_{i-1}, R_{i-1})$
- □ $P(C_i|B_i, C_{i-1})$
- □ $P(B_i|B_{i-1}, C_{i-1})$
- □ None of the above

We can expand this probability into a product of two through the chain rule, and simplify using conditional independences.

(c) [2 pts] An issue is that running this HMM takes time. And every day Andy puts off the decision to ask Brianna out makes him miserable. What of the following ways can he incorporate this logic into his algorithm?

- ☐ Conduct Laplace Smoothing on $B_t(B)$ and $B_t(C)$, using t as the hyperparameter k
- ☐ Replace $U(A, B)$ with $U_1(A, B, t)$, where $U_1(A, B, t) = U(A, B) + t$
- ■ Replace $U(A, B)$ with $U_2(A, B, t)$, where $U_2(A, B, t) = \frac{U(A,B)}{t}$
- ☐ None of the above

Higher values of t will result in lower overall utility, so we want to replace the utility function with U_2. Laplace smoothing doesn't make sense on a probability distribution that we've found from an HMM.

(d) [2 pts] Andy's friend Joe thinks he might be able to break this HMM into two separate HMMs, one with state variable B and evidence variables R and M, and one with state variable C and evidence variable R. Joe says Andy can then calculate his beliefs of C and B separately. Is this approach valid?

- ○ Yes, because C doesn't affect the final utility
- ○ Yes, but not for the reason above
- ○ No, because B and C are conditionally independent
- ● No, but not for the reason above

B and C are not conditionally independent because there exists an active path $B_i to M_i to C_i$.

(e) [2 pts] Andy solved for his belief of his HMM at a time t. He's ready to make his decision. Does he need any other factors to decide on asking Brianna out?

- ○ He needs the factor $P(F)$
- ○ He needs the factors $P(F)$ and $P(C|R, M)$
- ○ He needs factors other than the options above
- ● He doesn't need any additional factors, but he would have to do additional computation before using the belief in the decision net.
- ○ He doesn't need any additional factors, and he would not have to do additional computation before using the belief in the decision net

The belief at time t is $P(B, C|R, M)$. To solve the decision net, he just needs $P(B|R, M)$ since B is the only non-deterministic input to U and R and M are the only evidence. So he would have to marginalize before using the output of the HMM in the decision net.

Q7. [18 pts] Games

(a) In the following problems please choose all the answers that apply. You may pick more than one answer.

(i) [2 pts] In the context of adversarial search, $\alpha - \beta$ pruning

■ can reduce computation time by pruning portions of the game tree

☐ is generally faster than minimax, but loses the guarantee of optimality

■ always returns the same value as minimax for the root of the tree

■ always returns the same value as minimax for all nodes on the leftmost (first to be explored) edge of the tree, assuming successor game states are expanded from left to right

☐ always returns the same value as minimax for all nodes of the tree

(ii) [2 pts] Consider an adversarial game in which each state s has minimax value v(s). Assume that the maximizer plays according to the optimal minimax policy π, but the opponent (the minimizer) plays according to an unknown, possibly suboptimal policy π'. Which of the following statements are true?

■ The score for the maximizer from a state s under the maximizer's control could be greater than v(s)

☐ The score for the maximizer from a state s under the maximizer's control could be less than v(s).

☐ Even if the opponent's strategy π' were known, the maximizer should play according to π.

■ If π' is optimal and known, the outcome from any s under the maximizer's control will be v(s).

(iii) [3 pts] Consider a very deep game tree where the root node is a maximizer, and the complete-depth minimax value of the game is known to be v_∞. Similarly, let π_∞ be the minimax-optimal policy. Also consider a depth-limited version of the game tree where an evaluation function replaces any tree regions deeper than depth 10. Let the minimax value of the depth-limited game tree be v_{10} for the current root node, and let π_{10} be the policy which results from acting according to a depth 10 minimax search at every move. Which of the following statements are true?

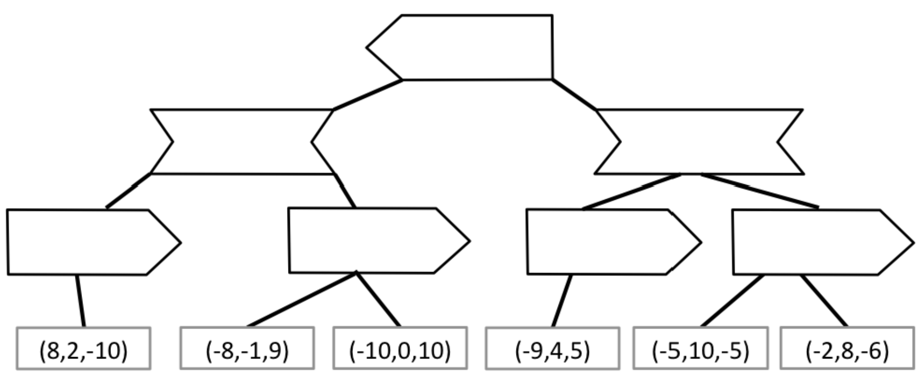
■ v_∞ may be greater than or equal to v_{10}

■ v_∞ may be less than or equal to v_{10}

☐ Against a perfect opponent, the actual outcome from following π_{10} may be greater than v_∞.

■ Against a perfect opponent, the actual outcome from following π_{10} may be less than π_∞. This assumes that the perfect opponent is playing with infinite depth lookahead.

(b) **(i)** [2 pts] Consider the 3-player game shown below. The player going first (at the top of the tree) is the Left player, the player going second is the Middle player, and the player going last is the Right player, optimizing the left, middle and right components respectively of the utility vectors shown. Fill in the values at all nodes. Note that all players maximize their own respective utilities.

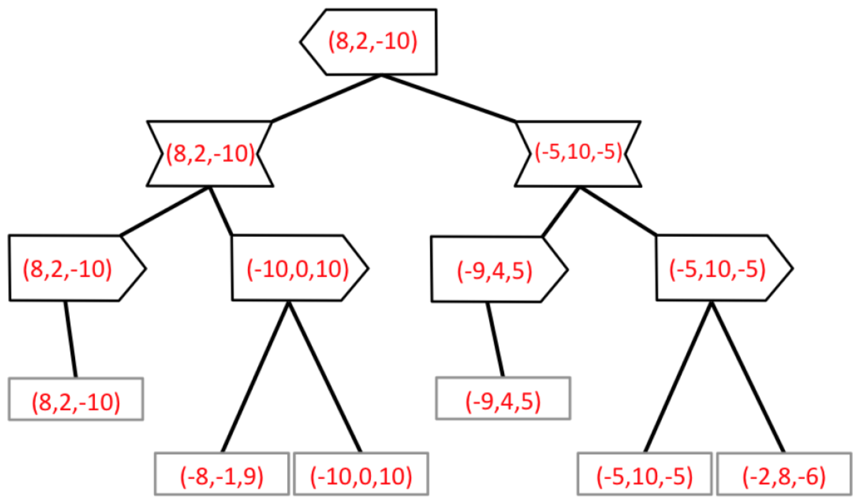


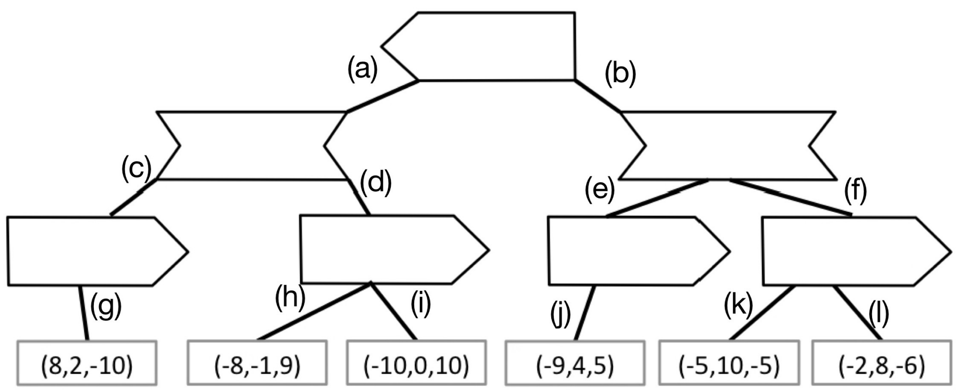
Solution:

(ii) [6 pts]

We have the knowledge that the sum of the utilities of all 3 players is always zero.

Select all edges for which observing the node will not affect the top-level decision (edges that can be pruned). To clarify, we only care about the left player's value.





If you prune a parent branch, do not mark any downstream children as pruned.

- [ ] (f)
- [ ] (i)
- [x] (l)
- [ ] None of the above

(l) can be pruned: The node farthest on the right can be pruned. This is because by the time search reaches this node, we know that the first player can achieve a utility of 8, the second player can achieve a utility of 4, and the third player can achieve a utility of -5. For the third player to pick the node connected to branch (l) it needs to be a value > -4. Since we know this is a zero sum game, if the value of third player is > -4 then the sum of player 1 and player 2 has to be < 4.

Let the node be x,y,z. We know z > -4, hence x + y < 4.

Case 1. x <= 8 and y <= 4: in this case neither players would pick this node, since both have preferable values already.

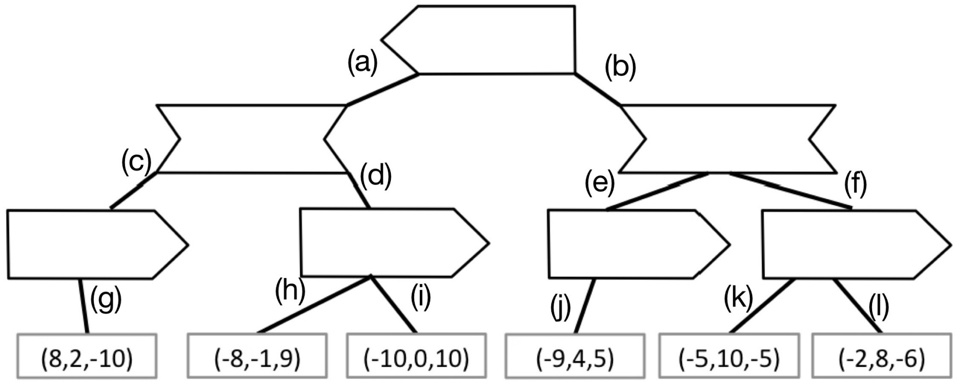
Case 2. y > 4. Hence, x < -1; This node might be preferred by player 2. However, player 1 will HAVE to have a value < 8 in order for the sum to 0. Hence, player 1 will end up rejecting this node.

Case 3. x > 8. Hence, y < -5. In this case player 2 will reject the node, because it's value would definitely have to be < 4 to get the sum to be 0 and player 2 will choose the node that gives it a value of 4.

(i) cannot be pruned: Each player is trying to maximize their own value. Since there is no upper limit on the value that a node can take, there isn't enough information yet to prune (i). This is because even if the 3rd player has a value > 10, the second player could still have a value > 2 and this node could end up being propagated by the middle player. Hence, we cannot prune (i)

(f) cannot be pruned for the same reason as (i), since there's no defined range of values that each value can take on. Hence, it's possible that the leaf nodes in subtree under (f) contain the nodes (9;5;-14) and (10;6;-16). In this case the right player will propagate (9;5;-14) and this is preferable for both the middle player and the left player and will end up being the top value.

(iii) [3 pts]



If we assume more about a game, additional pruning may become possible. Now, in addition to assuming that the sum of the utilities of all 3 players is still zero, we also assume that all utilities are in the interval [-10, 10]. Select all edges that would be pruned under these assumptions.

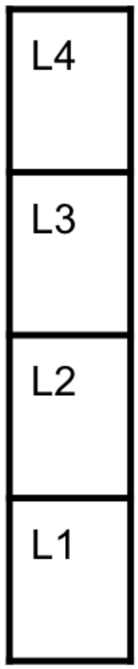
If you prune a parent branch, do not mark any downstream children as pruned.

- ■ (f)
- ■ (i)
- ☐ (l)
- ☐ None of the above

Pruning is now possible, for we can bound the best case scenario for the player above us. For the first pruning at (-10; 0; 10), the Right player sees a value of 9. This means that Right can get at least a 9, no matter what other values are explored next. If Right gets a 9, then Middle can only get at best a 1 (calculated from $C - v_r = 10 - 9$), which corresponds to the utility triple $(-10; 1; 9)$. Middle however has an option for a 2 above, so Middle would never prefer this branch so we can prune. Similar reasoning holds for the pruning on the right Middle sees a 4, which means Left would get at best a 6, but Left already has an option for an 8, so all other parts of this tree will never be returned upwards.

Q8. [15 pts] Hidden Markov Models and Particle Filtering

The elevator in the Tower of Terror moves up and down to the other floors (L1, L2, L3, L4). The location of the elevator at time t is X_t. At the beginning of each timestep,
(i) the elevator goes upwards with a probability of 0.4. It may go to any floor above its current position with equal probability.
(ii) the elevator goes downwards with a probability of 0.4. It may go to any floor below its current position with equal probability.
(iii) the elevator stays where it is with a probability of 0.2. If the elevator is on floor L4, it goes down with probability 0.8 and stays in position with probability 0.2. Similarly, if the elevator is on floor L1, it goes up with probability 0.8 and stays in position with probability 0.2.



X_0	$P(X_0)$
L4	0.2
L3	0.2
L2	0.3
L1	0.3

(a) [3 pts] Fill in the table below with the distribution of the elevator's location at time t = 1.

X_1	$P(X_1)$
L4	0.26
L3	
L2	
L1	

X_1	$P(X_1)$
L4	0.26
L3	0.2*0.2+0.2*0.8/3+0.3*0.4/2+0.3*0.8/3 = 0.23
L2	0.3*0.2+0.3*0.8/3+0.2*0.4/2+0.2*0.8/3 = 0.233
L1	0.3*0.2+0.3*0.4+0.2*0.4/2+0.2*0.8/3 = 0.273

X_1	P(X_1)
L4	0.26 (Which is wrong. Should have been 0.3)
L3	0.2*0.8/3 + 0.2*0.4/2 + 0.3*0.2 + 0.3*0.8/3 = 0.233
L2	0.2*0.8/3 + 0.2*0.2 + 0.3*0.4/2 + 0.3*0.8/3 = 0.233
L1	0.2*0.2 + 0.2*0.8/3 + 0.3*0.4/2 + 0.3*0.8/3 = 0.233 or 1 - 0.26 - 0.233 - 0.233 = 0.273 (if using P(X_1 = L4) = 0.26)

X_0	P(X_0)
L1	0.2
L2	0.2
L3	0.3
L4	0.3

Figure 1: Solutions for an alternative initial distribution X_0 (shown on the right). Note that this question is initially not intended to have an alternative initial distribution, so this figure is added after final exam grade release.

(b) Calculate the stationary distribution for the tower states by filling the unknown values in the matrix below

$$\begin{bmatrix} 0.2 & (i) & 0.2 & 0.266 \\ (ii) & (iii) & 0.2 & 0.266 \\ 0.266 & 0.2 & 0.2 & 0.266 \\ 0.266 & (iv) & 0.4 & 0.2 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} P(X_\infty = L1) \\ P(X_\infty = L2) \\ P(X_\infty = L3) \\ P(X_\infty = L4) \end{bmatrix} = \begin{bmatrix} P(X_\infty = L1) \\ P(X_\infty = L2) \\ P(X_\infty = L3) \\ P(X_\infty = L4) \\ 1 \end{bmatrix}$$

Fill in the missing values in the stationary system of equations for the following subparts.

(i) [1 pt] 0.4

(ii) [1 pt] 0.266, or 0.268 (to make the first column a probability distribution

(iii) [1 pt] 0.2

(iv) [1 pt] 0.2

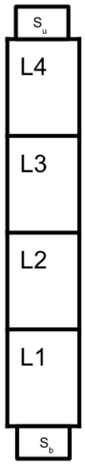
To keep track of the position of the elevator, a sound sensor S_u is installed on the top of the tower and a sound sensor S_b is installed in the basement. Both sensors detect the excited sounds of the passengers, ($+s$), or no sound at all, -s. The distribution of sensor measurements is determined by d, the number of floors between the elevator and the respective sensor. For example, if the elevator is on floor L3, then $d_b = 2$ because there are two floors (L2 and L1) between floor L3 and the bottom and $d_u = 1$ because there is one floor (L4) between floor L3 and the top. The prior of the both sensors' outputs are identical and listed below.

(c) [2 pts] You decide to track the elevator's position by particle filtering with 3 particles. At the end of time t = 1, the particles are at positions p_1 = L1, p_2 = L2 and p_3 = L3. Without incorporating any sensory information, what is the probability that the particles will be resampled as p_1 = L3, p_2 = L2, and p_3 = L4, at the end of time t=1?

$P(p_1 = L3|p_1 = L1) * P(p_2 = L2|p_2 = L2) * P(p_3 = L4|p_3 = L3)$

$0.8/3 * 0.2 * 0.4$

0.0213



S_u	$P(S_u \mid d_u)$
+s	$0.2/(d_u+1)$
-s	$1 - 0.2/(d_u+1)$

S_b	$P(S_b \mid d_b)$
+s	$1 - 0.2*d_b$
-s	$0.2*d_b$

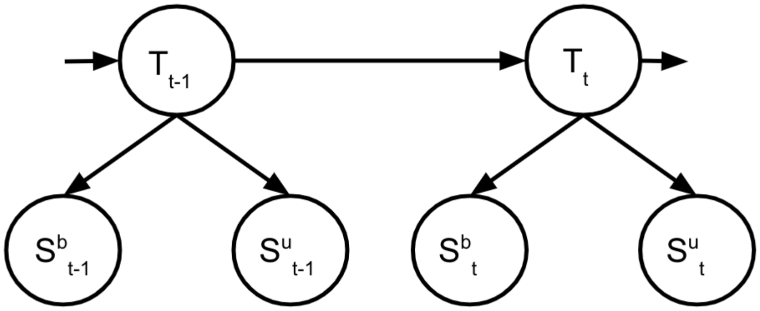
(d) [3 pts] To decouple this from the previous question, assume the particles after time elapsing are p_1 = L3, p_2 = L2, p_3 = L1, and the sensors observe S_u = +s and S_b = -s. What are the particle weights given these observations?

Particle	Weight
X_1 = L3	
X_2 = L2	
X_3 = L1	

Particle	Weight
p_1 = L3	$P(S_u=+s \mid d_u=1)\ P(S_b=-s \mid d_b=2) = 0.2/2 * 0.2*2 = 0.04$
p_2 = L2	$P(S_u=+s \mid d_u=2)\ P(S_b=-s \mid d_b=1) = 0.2/3 * 0.2*1 = 0.0133$
p_3 = L1	$P(S_u=+s \mid d_u=3)\ P(S_b=-s \mid d_b=0) = 0.2/4 * 0.2*0 = 0$

(e) [3 pts] Note: the u and b subscripts from before will be written here as superscripts. Part of the expression for the forward algorithm update for Hidden Markov Models is given below. $s^u_{0:t}$ are all the measurements from the roof sensor s^u_0, s^u_1, s^u_2, ... , s^u_t. $s^b_{0:t}$ are all the measurements from the roof sensor s^b_0 , s^b_1 , s^b_{12} , ... , s^b_t.

Choose all the correct options for the blank given



$P(x_t|s^u_{0:t}, s^b_{0:t}) \propto P(x_t, s^u_{0:t}, s^b_{0:t})$

$= \sum_{x_{t-1}} P(x_{t-1}, x_t, s^u_{0:t}, s^b_{0:t})$

$= \sum_{x_{t-1}} P(x_{t-1}, x_t, s^u_{0:t-1}, s^b_{0:t-1}, s^u_t, s^b_t)$

$\propto \sum_{x_{t-1}}$ ______

- [x] $P(s^u_t, s^b_t|x_{t-1}, x_t, s^u_{0:t-1}, s^b_{0:t-1})$
- [x] $P(s^u_t|x_t)P(s^b_t|x_t)$
- [ ] $P(s^u_t|x_{t-1})P(s^b_t|x_{t-1})$
- [ ] $P(s^u_t|s^u_{t-1})P(s^b_t|s^b_{t-1})$
- [ ] None of the above