INSTRUCTIONS

- Exam length: 180 minutes
- You are permitted to have three handwritten 8.5”x11” pages of notes, double-sided
- No calculators or other electronic devices allowed

Name	
Andrew ID	

For staff use only

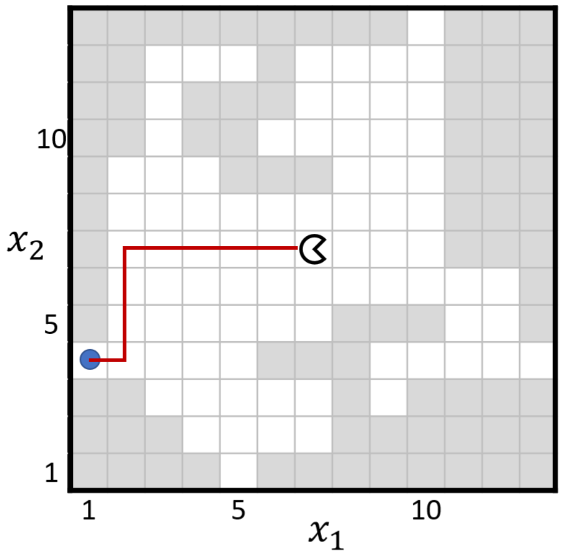
Q1. Pacman Search	/ 30
Q2. Game Trees	/ 21
Q3. Encrypted Knowledge Base	/ 14
Q4. Linear and Integer Programming	/ 25
Q5. MDPs	/ 25
Q6. Learning Games	/ 20
Q7. Bayes Nets and Sampling	/ 14
Q8. Hidden Markov Models	/ 35
Q9. Game Theory	/ 16
Total	/200

Q1. [35 pts] Pacman Search

There is a slight tilt to Pacman's grid and all the food has rolled to the lowest position on the grid that is not occupied by a wall, $\boldsymbol{x}_{low}$. The elevation of each grid location is given by $\boldsymbol{c}^T\boldsymbol{x}$, where:

$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ is a grid location with x_1 representing the horizontal component and x_2 the vertical component

$\boldsymbol{c} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$ is a vector to be defined later



(a) [5 pts] Given the vector $\boldsymbol{c} = [1, 0]^T$ and the grid below, where is the location of all the food, $\boldsymbol{x}_{low}$? In other words, what grid location (x_1, x_2) has lowest elevation and is not occupied by a wall.

$\boldsymbol{x}_{low}$?
(1,4)

Pacman's available actions are *Up*, *Right*, *Down*, *Left* when there is not a wall in the neighboring location in that direction. These actions simply move Pacman one space in that direction.

Pacman is trying to find the shortest path to the food, where path length is defined by the number of actions taken.

When searching, Pacman will use the following heuristic function:

$$h(\boldsymbol{x}) = \boldsymbol{c}^T\boldsymbol{x} - \boldsymbol{c}^T\boldsymbol{x}_{low}$$

(b) [5 pts] Assume Pacman is using graph search with a greedy priority defined by $h(\boldsymbol{x}) = \boldsymbol{c}^T\boldsymbol{x} - \boldsymbol{c}^T\boldsymbol{x}_{low}$. *Any ties* encountered during search are broken by the following order of actions: *Up*, then *Right*, then *Down*, then *Left*.

Given the vector $\boldsymbol{c} = [1, 0]^T$ and Pacman's starting location on the grid above, determine the path on the grid above that Pacman will travel along after this greedy graph search. What is the first action that Pacman will take along his path?

First Action?
Left

(c) [5 pts] What grid location (x_1, x_2) will he stop taking that action, pivot and begin a new one? (i.e., if he goes Right 3 cells in a row, we would expect you to write (10,7).

Pivot point?
(2,7)

(d) [5 pts] Is the $h(\boldsymbol{x}) = \boldsymbol{c}^T\boldsymbol{x} - \boldsymbol{c}^T\boldsymbol{x}_{low}$ with $\boldsymbol{c} = [1, 0]^T$ admissible? Is it consistent?

Admissible?	**Consistent?**
Yes	Yes

(e) [5 pts] What specific restrictions do we need to place on $\boldsymbol{c}$ to guarantee admissibility of $h(\boldsymbol{x})$?

Answer:
The magnitude of each component c_i must be at most one.

(f) For any $\boldsymbol{c}$ such that $h(\boldsymbol{x}) = \boldsymbol{c}^T\boldsymbol{x} - \boldsymbol{c}^T\boldsymbol{x}_{low}$ is both admissible and consistent, we now consider a new heuristic:

$$h_s(\boldsymbol{x}) = s(h(\boldsymbol{x}))\; h(\boldsymbol{x})$$

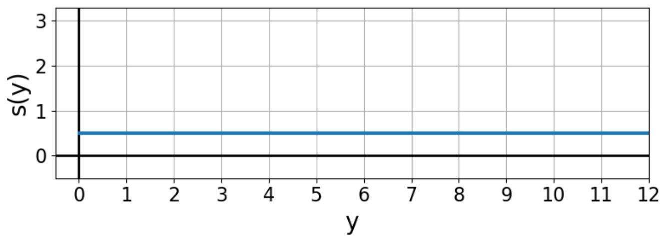
where $s(y)$ may be any one of the following functions.

For each of the $s(y)$ functions below, circle Yes or No to indicate if the resulting $h_s(\boldsymbol{x})$ is admissible and/or consistent.

We have provided plots of $s(y)$ for your convenience. *Note:* These are plots of $s(y)$, not plots of $h_s(\boldsymbol{x})$.

(i) [2 pts]

$s(y) = 1/2$

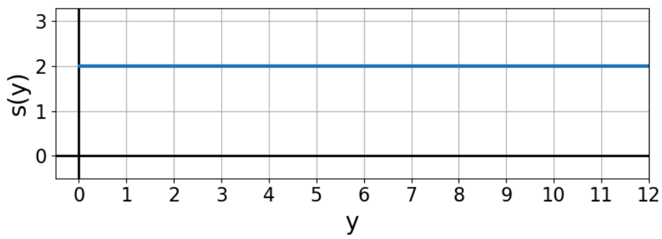


Admissible? **Yes** No

Consistent? **Yes** No

(ii) [2 pts]

$s(y) = 2$

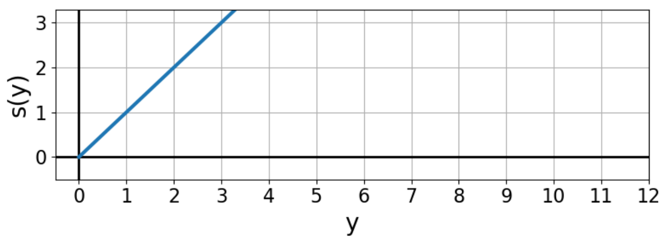


Admissible? Yes **No**

Consistent? Yes **No**

(iii) [2 pts]

$s(y) = y$

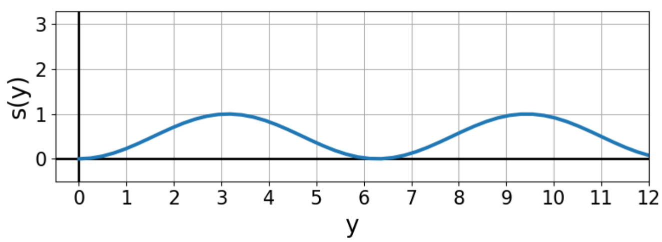


Admissible? Yes **No**

Consistent? Yes **No**

(iv) [2 pts]

$s(y) = -\frac{1}{2}\cos(y) + \frac{1}{2}$

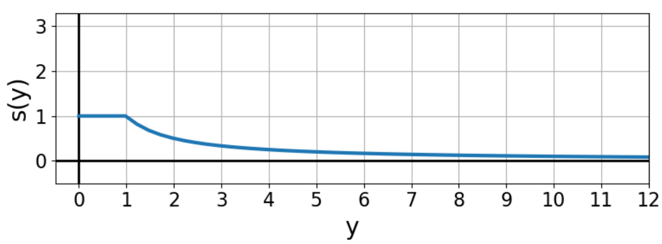


Admissible? **Yes** No

Consistent? Yes **No**

(v) [2 pts]

$s(y) = \min(1, 1/y)$



Admissible? **Yes** No

Consistent? **Yes** No

Q2. [21 pts] Game Trees

(a) Alpha-beta pruning true/false

(i) [3 pts] Minimax search with alpha-beta pruning may not find a minimax-optimal strategy.

○ True ● False

False. Alpha-beta will always find the optimal strategy for players playing optimally.

(ii) [3 pts] Alpha-beta pruning prunes the same number of subtrees independent of the order in which successor states are expanded.

○ True ● False

False. If a heuristic is available, we can expand nodes in an order that maximizes pruning.

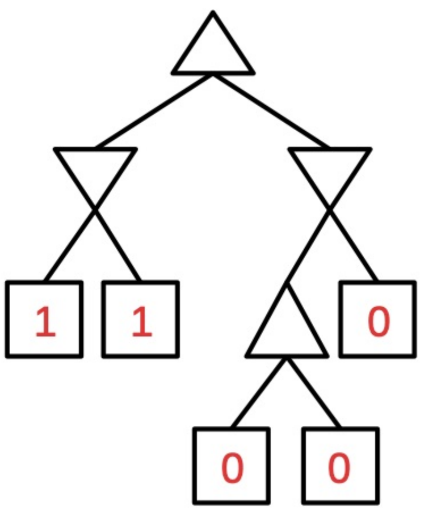
(iii) [3 pts] Minimax search with alpha-beta pruning generally requires more computation time than minimax without pruning on the same game tree.

○ True ● False

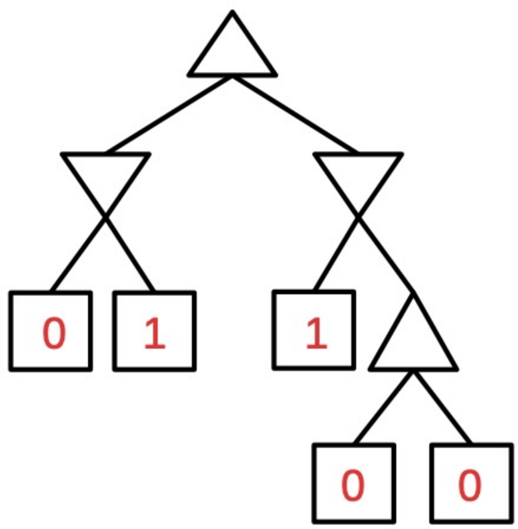
False. Alpha-beta will require less computation time than minimax except in contrived cases, which is the whole point of pruning.

(b) [12 pts] For each of the following minimax game trees (max is at the root), determine whether no pruning is done, the maximum amount of pruning is done, or neither (some but not all leaves that could be pruned are pruned). Note that the two trees are slightly different.

Assume (1) left to right traversal while pruning, and (2) leaf utility values are potentially drawn from $[-\infty, \infty]$ (not from [0, 1]).



○ No Pruning
● Maximum Pruning
○ Neither



● No Pruning
○ Maximum Pruning
○ Neither

Q3. [14 pts] Encrypted Knowledge Base

We have a propositional logic knowledge base, but unfortunately, it is encrypted. The only information we have is that:

- Each of the following 12 boxes contains a propositional logic symbol (A, B, C, D, or E) or a propositional logic operator and
- Each line is a valid propositional logic sentence.

□ 1 □ 2

□ 3 □ 4 □ 5

□ 6

□ 7 □ 8 □ 9

□ 10 □ 11 □ 12

(a) [6 pts] We are going to implement a constraint satisfaction problem solver to find a valid assignment to each box from the domain $\{A, B, C, D, E, \wedge, \vee, \neg, \Rightarrow, \Leftrightarrow\}$.

Propositional logic syntax imposes constraints on what can go in each box. What values are in the domain of boxes 1-6 after enforcing the unary syntax constraints?

Box	Remaining Values
1	$\neg$
2	$A\ B\ C\ D\ E$
3	$A\ B\ C\ D\ E\ \neg$
4	$\wedge\ \vee\ \neg\ \Rightarrow\ \Leftrightarrow$
5	$A\ B\ C\ D\ E$
6	$A\ B\ C\ D\ E$

(b) [4 pts] You are given the following assignment as a solution to the knowledge base CSP on the previous page:

$$\neg A$$
$$B \Rightarrow A$$
$$D$$
$$C \vee B$$
$$D \vee E$$

Now that the encryption CSP is solved, we have an entirely new CSP to work on: finding a model. In this new CSP the variables are the symbols $\{A, B, C, D, E\}$ and each variable could be assigned to $true$ or $false$.

We are going to run CSP backtracking search with forward checking to find a propositional logic model M that makes all of the sentences in this knowledge base true.

After choosing to assign C to false, what values are removed by running forward checking? Based on the table of remaining values below, list one of the values that were removed.

Symbol	Remaining Values
A	F
B	T ~~F~~
C	F
D	T
E	T F

Variable:	**Value that was removed**
B	F

Forward checking removes the value false from the domain of B. Forward checking does not continue on to make any other arcs consistent.

(c) [4 pts] We eventually arrive at the model $M = \{A = False, B = False, C = True, D = True, E = True\}$ that causes all of the knowledge base sentences to be true. We have a query sentence α specific as $(A \vee C) \Rightarrow E$. Our model M also causes α to be true. Can we say that the knowledge base entails α? Explain briefly (in one sentence) why or why not.

○ Yes ● No

Explain
No, the knowledge base does not entail α. There are other models for which the knowledge base could be true and the query be false. Specifically $\{A = False, B = False, C = True, D = True, E = False\}$ satisfies the knowledge base but causes the query α to be false.

Q4. [25 pts] Linear and Integer Programming

(a) Multiple Choice

(i) [3 pts] Given a two-dimensional linear program in inequality form with objective $\boldsymbol{c}^T\boldsymbol{x}$:
As the magnitude of the cost vector $\boldsymbol{c}$ increases, the distance between objective value contour lines:

○ Increases ● Decreases

(ii) [3 pts] Given a two-dimensional linear program in inequality form with objective $\boldsymbol{c}^T\boldsymbol{x}$:
True/False: The contour line where the objective value equals zeros always passes through the origin.

● True ○ False

(iii) [3 pts] Let $\boldsymbol{x}^*_{IP}$ and y^*_{IP} be the optimal solution point and optimal objective value of a linear program. Let $\boldsymbol{x}^*_{LP}$ and y^*_{LP} be the optimal solution point and optimal objective value of the corresponding relaxed linear program.

Select ALL of the following that are true:

- ☐ $\boldsymbol{x}^*_{IP} = \boldsymbol{x}^*_{LP}$
- ☐ $y^*_{IP} \leq y^*_{LP}$ if it is a minimization problem
- ☒ $y^*_{IP} \geq y^*_{LP}$ if it is a minimization problem
- ☒ $y^*_{IP} \leq y^*_{LP}$ if it is a maximization problem
- ☐ $y^*_{IP} \geq y^*_{LP}$ if it is a maximization problem
- ☐ None of the above

(b) [16 pts] Branch and Bound

Consider the following three-dimensional integer program in inequality form:

$$\begin{aligned} \min_{\boldsymbol{x}}.\ & \boldsymbol{c}^T\boldsymbol{x} \\ \text{s.t.}\ & A_0\boldsymbol{x} \preceq \boldsymbol{b}_0 \\ & \boldsymbol{x} \in \mathbb{Z}^3 \end{aligned} \qquad A_0 = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \\ a_{4,1} & a_{4,2} & a_{4,3} \end{bmatrix} \qquad \boldsymbol{b}_0 = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$$

We will use branch and bound to solve for the optimal IP solution. We have access to an inequality-form LP solver, `solveLP`, that takes in a matrix, A, a vector $\boldsymbol{b}$, and a cost vector $\boldsymbol{c}$, and returns the optimal LP solution point and objective value.

At the root node of branch and bound, `solveLP`$(A_0, \boldsymbol{b}_0, \boldsymbol{c})$ returns the point:

$$\boldsymbol{x} = [x_1, x_2, x_3]^T = [2.5, 5.3, -10.1]^T$$

Because this is not an integer solution we choose to continue by branching on x_1, calling `solveLP` for these two branches.

Specify the modified A matrix and modified $\boldsymbol{b}$ vector required to call `solveLP` for these two branches.

Note: You will need to reuse entries from A_0 ($a_{i,j}$) and $\boldsymbol{b}_0$ (b_i). *Note*: Let the left branch be the $\leq$ branch.

Select all the rows in the modified A matrix after **left** branching on x_1.

- ☒ $a_{1,1}\ a_{1,2}\ a_{1,3}$
- ☒ $a_{2,1}\ a_{2,2}\ a_{2,3}$
- ☒ $a_{3,1}\ a_{3,2}\ a_{3,3}$
- ☒ $a_{4,1}\ a_{4,2}\ a_{4,3}$
- ☐ 1 1 1
- ☒ 1 0 0
- ☐ 0 0 1
- ☐ -1 0 0

After **left** branching on x_1, our modified $\boldsymbol{b}$ vector will have one additional value. What is the new added value?

What is the new added value?
2

Select all the rows in the modified A matrix after **right** branching on x_1.

- [x] $a_{1,1}$ $a_{1,2}$ $a_{1,3}$
- [x] $a_{2,1}$ $a_{2,2}$ $a_{2,3}$
- [x] $a_{3,1}$ $a_{3,2}$ $a_{3,3}$
- [x] $a_{4,1}$ $a_{4,2}$ $a_{4,3}$
- [ ] 1 1 1
- [ ] 1 0 0
- [ ] 0 0 1
- [x] -1 0 0

After **right** branching on x_1, our modified $\boldsymbol{b}$ vector will have one additional value. What is the new added value?

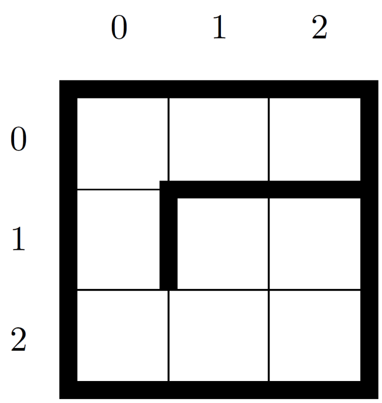
What is the new added value?
-3

Q5. [25 pts] MDPs

Pacman is in a maze where he gets a reward every time he visits state (0,0). This setup is a bit different from the one you've seen before: Pacman can get the reward multiple times; these rewards do not get "used up" like food pellets and there are no "living rewards". As usual, Pacman can not move through walls and may take any of the following **deterministic** actions: go North (↑), South (↓), East (→), West (←), or stay in place (∘). State (0,0) gives a total reward of 1 every time Pacman takes an action in that state regardless of the outcome, and all other states give no reward. To be precise, the reward function is: $R_{(0,0),a} = 1$ for any action a and $R_{s',a} = 0$ for all $s' \neq (0,0)$

You should not need to use any other complicated algorithm/calculations to answer the questions below. We remind you that geometric series converge as follows: $1 + \gamma + \gamma^2 + \cdots = 1/(1-\gamma)$.

(a) [14 pts] Assume finite horizon of $h = 10$ (so Pacman takes exactly 10 steps) and no discounting ($\gamma = 1$). We want to fill in an optimal policy and value function for the following maze.



Optimal policy: Fill in Pacman's optimal action for each of the following grid spaces with either N, S, E, W, or X. Note: (2,0) corresponds to the top right corner and (0,2) corresponds to the bottom left corner.

(0, 0)	**(0, 2)**	**(1, 1)**	**(2, 0)**	**(2, 2)**
X	N	S	W	W

Value Function: Fill in the value for each of the following grid spaces.

(0, 0)	**(0, 1)**	**(0, 2)**	**(1, 0)**	**(1, 1)**
10	9	8	9	6

(1, 2)	**(2, 0)**	**(2, 1)**	**(2, 2)**
7	8	5	6

(b) Assume finite horizon of $h = 10$, no discounting, but the action to stay in place is temporarily (for this sub-point only) unavailable. Actions that would make Pacman hit a wall are not available. Specifically, Pacman can not use actions North or West to remain in state $(0, 0)$ once he is there.

(i) [3 pts] True/False: There is just one optimal action at state $(0, 0)$

○ True ● False

East and South are both optimal actions

(ii) [3 pts] The value of state $(0, 0)$ is: 5

Since the "stay action" is no longer available, Pacman needs to exit state $(0, 0)$ at even time steps

(c) [5 pts] Assume infinite horizon, discount factor $\gamma = 0.9$, and the "stay action" is available.

The value of state $(0, 0)$ is: $1/(1-\gamma) = 10$

Q6. [20 pts] Learning Games

(a) [5 pts] **Q-Learning to Play an Adversarial Game.** Pacman does exact Q-learning (where every state-action pair has its own Q-value) to figure out how to play a game against adversarial ghosts that are trying to minimize Pacman's score. As he likes to explore, Pacman always plays a random action. After enough time has passed, every state-action pair is visited infinitely often. The learning rate decreases as needed. For any game state s, the value $\max_a Q(s, a)$ for the learned $Q(s, a)$ is equal to (for complete search trees) which one of the following:

● The minimax value where Pacman maximizes and ghosts minimize.
○ The expectimax value where Pacman maximizes and ghosts act uniformly at random.
○ The expectimax value where Pacman plays uniformly at random and ghosts minimize.
○ The expectimax value where both Pacman and ghosts play uniformly at random.
○ None of the above.

Only minimax search correctly models the adversarial game of Pacman's learned policy: although the acting policy is random, the learned policy is the optimal policy for max.

Tabular Q-learning and full-depth minimax search both compute the exact value of all states, since Q-learning has a value for every state-action (and thus every state) and the conditions are right for convergence.

(b) [5 pts] **Approximate Q-Learning for an Adversarial Game** Pacman now runs feature-based Q-learning. The Q-values are equal to the evaluation function $\sum_{i=1}^{n} w_i f_i(s, a)$ for weights w and features f. The number of features is much less than the number of states. As he likes to explore, Pacman always plays a random action. After enough time has passed, every state-action pair is visited infinitely often. The learning rate decreases as needed. The value $\max_a Q(s, a)$ for the learned $Q(s, a)$ is equal to (for complete search trees) which one of the following:

○ The minimax value where Pacman maximizes and ghosts minimize and the same evaluation function is used at the leaves.

○ The expectimax value where Pacman maximizes and ghosts act uniformly at random and the same evaluation function is used at the leaves.

○ The expectimax value where Pacman plays uniformly at random and ghosts minimize and the same evaluation function is used at the leaves.

○ The expectimax value where both Pacman and ghosts play uniformly at random and the same evaluation function is used at the leaves.

● None of the above.

Full-depth minimax search computes the approximate value of all the leaves by the evaluation function and then exactly propagates these values up the search tree. Feature-based Q-learning approximates the value of all states with the evaluation function and not only the leaves. Since there are fewer features than states, the approximation is not expressive enough to capture the true values of all the states.

(c) [5 pts] **A Costly Game.** Pacman is now stuck playing a new game with only costs and no payoff. Instead of maximizing expected utility $V(s)$, he has to minimize expected costs $J(s)$. In place of a reward function, there is a cost function $C(s, a, s')$ for transitions from s to s' by action a. We denote the discount factor by $\gamma \in (0, 1)$. $J^*(s)$ is the expected cost incurred by the optimal policy. Which one of the following equations is satisfied by J^*?

○ $J^*(s) = \min_a \sum_{s'} [C(s, a, s') + \gamma \max_{a'} T(s, a', s') * J^*(s')]$
○ $J^*(s) = \min_{s'} \sum_a T(s, a, s')[C(s, a, s') + \gamma * J^*(s')]$
○ $J^*(s) = \min_a \sum_{s'} T(s, a, s')[C(s, a, s') + \gamma * \max_{s'} J^*(s')]$
○ $J^*(s) = \min_{s'} \sum_a T(s, a, s')[C(s, a, s') + \gamma * \max_{s'} J^*(s')]$
● $J^*(s) = \min_a \sum_{s'} T(s, a, s')[C(s, a, s') + \gamma * J^*(s')]$
○ $J^*(s) = \min_{s'} \sum_a [C(s, a, s') + \gamma * J^*(s')]$

Minimum expected cost has the same form as maximum expected utility except that the optimization is in the opposite direction and costs replace rewards.

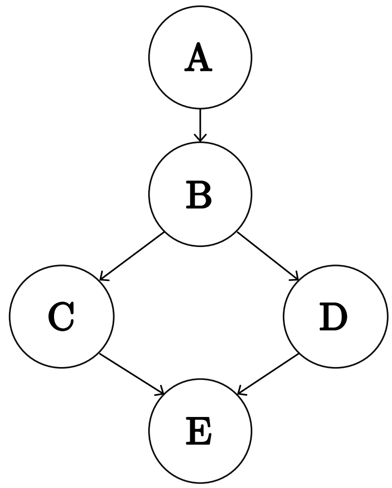
(d) [5 pts] **It's a conspiracy!** The ghosts have rigged the costly game so that once Pacman takes an action they can pick the outcome from all states $s' \in S'(s,a)$, the set of all s' with non-zero probability according to $T(s,a,s')$. Choose the correct Bellman-style equation for Pacman against the adversarial ghosts.

- ○ $J^*(s) = \min_a \max_{s'} T(s,a,s')[C(s,a,s') + \gamma * J^*(s')]$
- ○ $J^*(s) = \min_{s'} \sum_a T(s,a,s')[\max_{s'} C(s,a,s') + \gamma * J^*(s')]$
- ○ $J^*(s) = \min_a \min_{s'} [C(s,a,s') + \gamma * \max_{s'} J^*(s')]$
- ● $J^*(s) = \min_a \max_{s'} [C(s,a,s') + \gamma * J^*(s')]$
- ○ $J^*(s) = \min_{s'} \sum_a T(s,a,s')[\max_{s'} C(s,a,s') + \gamma * \max_{s'} J^*(s')]$
- ○ $J^*(s) = \min_a \min_{s'} T(s,a,s')[C(s,a,s') + \gamma * J^*(s')]$

Pacman is still minimizing cost, but instead of expected cost it is worst-case (maximum) cost among all possible successors s'. The transition probability $T(s,a,s')$ is dropped since the worst-case outcome is selected with certainty.

Q7. [14 pts] Bayes Nets and Sampling

(a) [5 pts] Given the following Bayes net, write an expression for the probability distribution $P(A|e)$ using only the conditional probability distributions associated with this Bayes net, e.g. $P(A)$, $P(C|B)$, etc. Normalization constants, such as α or Z are not permitted.



Which of the following is your expression for $P(A|e)$?

○ $\sum_{b,c,d} P(A)P(b \mid A)P(c \mid b)P(d \mid b)P(e \mid c,d)$

○ $\frac{\sum_{b,c,d,e} P(A)P(b|A)P(c|b)P(d|b)P(e|c,d)}{\sum_{a,b,c,d} P(a)P(b|a)P(c|b)P(d|b)P(e|c,d)}$

○ $\frac{\sum_{a} P(A)P(b|A)P(c|b)P(d|b)P(e|c,d)}{\sum_{a,b,c,d} P(a)P(b|a)P(c|b)P(d|b)P(e|c,d)}$

○ $\frac{\sum_{b,c,d} P(A)P(B)P(C)P(D)P(E)}{\sum_{a,b,c,d} P(A)P(B)P(C)P(D)P(E)}$

● $\frac{\sum_{b,c,d} P(A)P(b|A)P(c|b)P(d|b)P(e|c,d)}{\sum_{a,b,c,d} P(a)P(b|a)P(c|b)P(d|b)P(e|c,d)}$

(b) Alita is using likelihood weighted sampling to answer various queries on the Bayes net from part (d).

(i) [3 pts] True/False: For the query $P(E|b)$, the value sampled for variable A will have no effect on the weight of the complete sample.

○ True ● False

False. The weight on the sample will be $P(b|a)$, which could definitely be different for different values a.

Alita has implemented a simpler version of likelihood weighted sampling to answer the query $P(E|b)$. Alita's method skips sampling variable A and skips incorporating the weight associated with B and proceeds to sample values for C, D, and then E from $P(C|B)$, $P(D|B)$, and $P(E|C,D)$, respectively.

(ii) [3 pts] True/False: Alita's simpler sampling method will converge to the same answer for $P(E|b)$ as standard likelihood weighted sampling.

● True ○ False

True. Alita is sampling from $P(C \mid B)$, $P(D \mid B)$, and $P(E \mid C,D)$, so the resulting counts when you multiply these three together will approximate $P(C \mid B)P(D \mid B)P(E \mid C,D) = P(C,D,E \mid b)$. She will then compute the correct query be summing over C and D: $P(E \mid b) = \sum_c \sum_d P(c,d,E \mid b)$.

The multiplication above relied on the assumption that 1) E is independent of B given C, D and 2) C is independent from D given B.

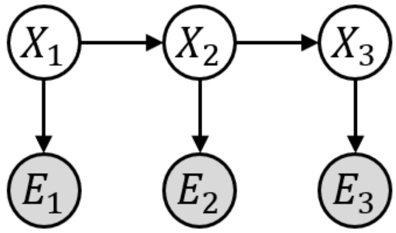
(iii) [3 pts] Is it possible to make a similar simplification to likelihood weighted sampling to make it more efficient (but still accurate in the limit) when answering the query $P(E|c)$. Briefly describe the simplification or why it is not possible.

○ Yes ● No

Explanation:
No, it is not possible. There are no independence relationships that can allow us to get around the need to sample A, B, D, and E, and compute the weight associated with C.

Q8. [35 pts] Hidden Markov Models

(a) For each of the following queries on the three time step HMM below, select the minimum set of the factors and the minimum set of the summations needed to compute that query. Also select whether or not we need to normalize after all factors have been multiplied and summed; do not include any items required only for normalization, i.e. those that only appear in the denominator.



Select all that apply, but not more than are necessary.

(i) [4 pts] $P(X_1 \mid e_1) =$

☒ Normalize (α)

☐ $\sum_{X_1}$ ☐ $\sum_{X_2}$ ☐ $\sum_{X_3}$ ☐ $\sum_{E_1}$ ☐ $\sum_{E_2}$ ☐ $\sum_{E_3}$

☒ $P(X_1)$ ☐ $P(X_2 \mid X_1)$ ☐ $P(X_3 \mid X_2)$

☒ $P(E_1 \mid X_1)$ ☐ $P(E_2 \mid X_2)$ ☐ $P(E_3 \mid X_3)$

(ii) [4 pts] $P(X_3 \mid e_1, e_2) =$

☒ Normalize (α)

☒ $\sum_{X_1}$ ☒ $\sum_{X_2}$ ☐ $\sum_{X_3}$ ☐ $\sum_{E_1}$ ☐ $\sum_{E_2}$ ☐ $\sum_{E_3}$

☒ $P(X_1)$ ☒ $P(X_2 \mid X_1)$ ☒ $P(X_3 \mid X_2)$

☒ $P(E_1 \mid X_1)$ ☒ $P(E_2 \mid X_2)$ ☐ $P(E_3 \mid X_3)$

(iii) [4 pts] $P(X_2, e_1, e_2) =$

☐ Normalize (α)

☒ $\sum_{X_1}$ ☐ $\sum_{X_2}$ ☐ $\sum_{X_3}$ ☐ $\sum_{E_1}$ ☐ $\sum_{E_2}$ ☐ $\sum_{E_3}$

☒ $P(X_1)$ ☒ $P(X_2 \mid X_1)$ ☐ $P(X_3 \mid X_2)$

☒ $P(E_1 \mid X_1)$ ☒ $P(E_2 \mid X_2)$ ☐ $P(E_3 \mid X_3)$

(iv) [4 pts] $P(X_1 \mid e_1, e_2, e_3) =$

☒ Normalize (α)

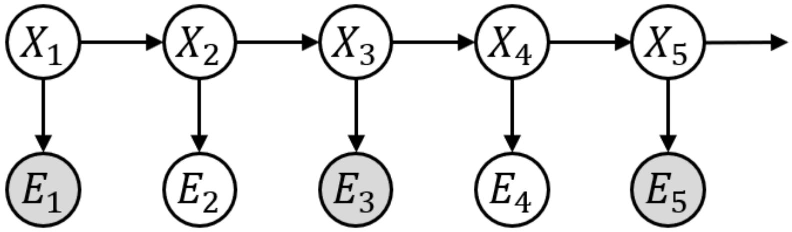
☐ $\sum_{X_1}$ ☒ $\sum_{X_2}$ ☒ $\sum_{X_3}$ ☐ $\sum_{E_1}$ ☐ $\sum_{E_2}$ ☐ $\sum_{E_3}$

☒ $P(X_1)$ ☒ $P(X_2 \mid X_1)$ ☒ $P(X_3 \mid X_2)$

☒ $P(E_1 \mid X_1)$ ☒ $P(E_2 \mid X_2)$ ☒ $P(E_3 \mid X_3)$

(b) [9 pts] Modified Forward Algorithm

We have been busy studying for finals and only had time to observe the evidence node on the odd days:



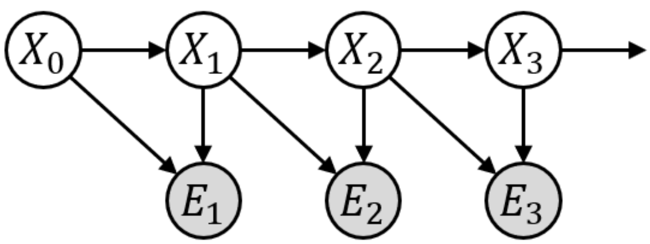
If we are given the probability distribution for the *third day* given evidence for days *one* and *three*, $P(X_3 \mid e_1, e_3)$, use this distribution and the standard HMM transition and emission probabilities to write the formula for the probability distribution for the *fifth* day given evidence for days *one*, *three*, **and** *five*, $P(X_5 \mid e_1, e_3, e_5)$. You may use α to indicate the need to normalize if necessary.

Which of the following can represent the specified probability distribution? Select all that apply.
$P(X_5 \mid e_1, e_3, e_5) =$

- [x] $\alpha P(e_5 \mid X_5) \sum_{x_4} P(X_5 \mid x_4) \sum_{x_3} P(x_4 \mid x_3) P(x_3 \mid e_1, e_3)$
- [ ] $\alpha P(e_5|X_5)P(e_3|X_3)P(e_1|X_1)$
- [ ] $\alpha P(e_5 \mid X_5) \sum_{x_4} P(X_5 \mid x_4) \sum_{x_3} P(x_4 \mid x_3) P(e_3|x_3)$
- [ ] $P(e_5 \mid X_5) \sum_{x_4} P(X_5 \mid x_4) \sum_{x_3} P(x_4 \mid x_3) P(x_3 \mid e_1, e_3)$
- [x] $\alpha P(e_5 \mid X_5) \sum_{x_4} P(X_5 \mid x_4) \sum_{x_3} P(x_4 \mid x_3) P(x_3 \mid e_1, e_3) \sum_{e_4} P(e_4 \mid x_4)$

(c) Modified Particle Filtering

As shown in the modified HMM below, our model for states and evidence actually have more dependencies than a standard HMM:



If we still want to do particle filtering on this HMM, we need to figure out the predict and update steps of this modified particle filtering algorithm.

(i) [5 pts] From what probability table do we sample to move a particle from time t to time $t+1$?

● $P(X_{t+1} \mid X_t)$ ○ $P(X_t|X_{t+1})$ ○ $P(e_t|X_t)$ ○ $P(e_{t+1}|X_t)$

(ii) [5 pts] From what probability table do we look up the weights for a particle at time t given new evidence at time t, e_t?

○ $P(X_t \mid e_t)$ ○ $P(e_t \mid X_{t-1})$ ○ $P(e_t \mid X_t)$ ● $P(e_t \mid X_{t-1}, X_t)$

Q9. [16 pts] Game Theory

When a soccer player is kicking a penalty kick, she will choose to kick either to the goalie's left or to his right. The goalie will leap to one side or the other in an attempt to block the kick; she will leap before determining to which side the kick will come, but too late for the kicker to change direction, For many professional kicker vs. goalie match-ups the following table gives a good approximation to the **probabilities that the kicker will score/miss a goal**, as a function of the two players' choices.

		Goalie	
		Left	Right
Kicker	Left	0.3, 0.7	0.9, 0.1
	Right	0.6, 0.4	0.1, 0.9

(a) [8 pts] Suppose that the kicker and the goalie make their move at the same time. What are the mixed Nash equilibrium strategies for the kicker and goalie?

Kicker Left:	**Kicker Right**	**Goalie Left**	**Goalie Right**
$\frac{5}{11}$	$\frac{6}{11}$	$\frac{8}{11}$	$\frac{3}{11}$

(b) [8 pts] If the kicker chooses her side first, and the goalie will respond to the kicker's action, what is the best strategy for the kicker? (Hint: Strong Stackelberg Equilibrium)

Left	**Right**
$\frac{5}{11}$	$\frac{6}{11}$

THIS PAGE INTENTIONALLY LEFT BLANK