CS 188 Spring 2024 — Introduction to Artificial Intelligence — Final

Solutions last updated: Thursday, April 19, 2024

- You have 170 minutes.
- The exam is closed book, no calculator, and closed notes, other than two double-sided cheat sheets that you may reference.
- Anything you write outside the answer boxes or you ~~cross out~~ will not be graded. If you write multiple answers, your answer is ambiguous, or the bubble/checkbox is not entirely filled in, we will grade the worst interpretation.

For questions with **circular bubbles**, you may select only one choice.

○ Unselected option (completely unfilled)

● Only one selected option (completely filled)

✓ Don't do this (it will be graded as incorrect)

For questions with **square checkboxes**, you may select one or more choices.

■ You can select

■ multiple squares (completely filled)

First name	
Last name	
SID	
Name of person to the right	
Name of person to the left	
Discussion TAs (or None)	

Honor code: "As a member of the UC Berkeley community, I act with honesty, integrity, and respect for others."

By signing below, I affirm that all work on this exam is my own work, and honestly reflects my own understanding of the course material. I have not referenced any outside materials (other than my cheat sheets), nor collaborated with any other human being on this exam. I understand that if the exam proctor catches me cheating on the exam, that I may face the penalty of an automatic "F" grade in this class and a referral to the Center for Student Conduct.

Signature: ______________________

Point Distribution

Q1.	Potpourri	16
Q2.	Search & Games: Shelving Books	15
Q3.	Logic: Easter Island Revisited	15
Q4.	MDPs: Blinky's Hamster	10
Q5.	VPI: Cost of Extra Samples	16
Q6.	RL: Optimistic Q-Learning	11
Q7.	ML: Spam Filter	17
	Total	100

We hope you set a new high score on this exam!

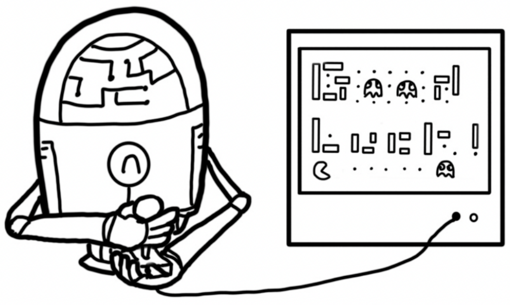


Image credit: Samantha Huang

Q1. [16 pts] Potpourri

(a) [2 pts] Which of the following statements about heuristics are true?

- ☐ The trivial heuristic can sometimes (but rarely) speed up A* search, compared to uniform cost search.
- ☐ If a heuristic is admissible, it is guaranteed to be consistent.
- [✓] If a heuristic is consistent, it is guaranteed to be admissible.
- ☐ Greedy tree search with a consistent heuristic is complete, but not optimal.
- ○ None of the above.

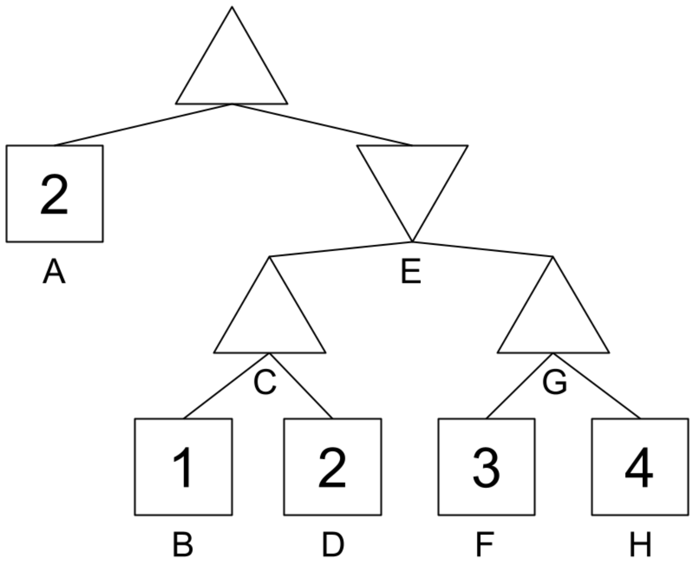
(b) [1 pt] Which of the following statements means that ϕ **is satisfiable**, where ϕ is an arbitrary logical expression and $\bot$ is always false?

○ $\phi \models \bot$ [✓ $\phi \not\models \bot$] ○ $\neg\phi \models \bot$ ○ $\neg\phi \not\models \bot$

For subparts (c)-(e), consider the following game tree. Assume that alpha-beta pruning visits nodes from left to right. Assume pruning on equality.

(c) [1 pt] Fill out the four missing values in the game tree below.

Node C is 2; node G is 4; node E is 2, and the topmost root node is 2.



(d) [2 pts] Select all terminal nodes that are never explored as a result of alpha-beta pruning.

☐ A ☐ B ☐ D [✓ F] [✓ H]

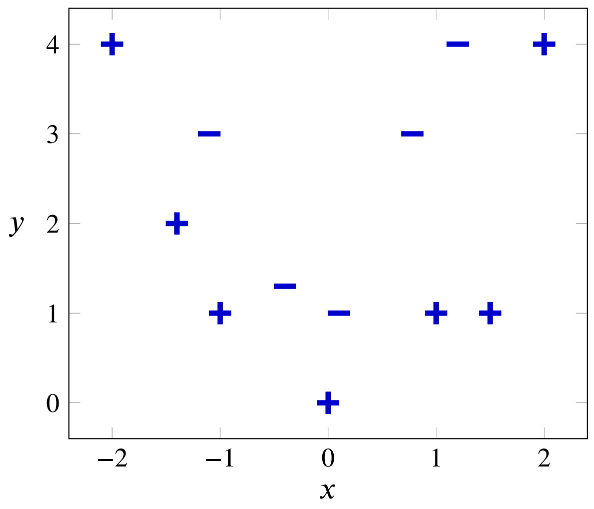
○ None of the above

(e) [2 pts] Which of the following pairs of nodes, when their values are swapped, will produce a tree where zero nodes are pruned during alpha-beta pruning?

- ○ A and B
- ○ B and H
- [✓ D and F]
- ○ F and H

The simplest solution is to place the numbers in ascending order; 1, 2, 2, 3, 4, or BADFH. However, we can note that the second and third leaf will never be pruned. Then, all we need to is to provide the E minimizer node a high enough value, such that the last two leaves won't be pruned. We choose the most alphabetical version of this setup, ABF, then DH for the last two leaves.

For subparts (f)-(g), consider the following dataset. Points represented by the plus symbol (+) correspond to a label of +1, and points represented by the minus symbol (−) correspond to a label of −1.



(f) [3 pts] Assuming that we fit a linear perceptron to this dataset, select all sets of features that would allow such a perceptron to perfectly classify the dataset. Assume points on the decision boundary are classified as label +1.

- ☐ x, y
- ■ x^2, y
- ☐ x^3, y
- ■ x^2, y^2, x, y
- ☐ $x, y, 1$
- ■ $x^2, y^2, y, 1$
- ○ None of the above.

The function that this graph should bring to mind is a parabola, $y = x^2$. Note that it is centered at zero (the x-coordinate of the vertex is zero). Any answer choice containing (x^2, y) in the features work. Adding additional features is OK; bias is not needed.

(g) [2 pts] Suppose we use x, y as the features, and the initial weight vector of the perceptron is $[0, 0, -1]$. In other words, the feature weights are $[0, 0]$ and the bias is -1.

After running 1 iteration of the perceptron training algorithm with the point at $x = -1, y = 1$, what is the new weight vector (including the bias term)?

$$\boxed{[-1, 1, 0]}$$

With the initial weights, 0, 0, -1, the sample at (-1, 1) would get a negative score and thus a label of -1, which is wrong. We update the weights with true label +1:

$$\begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} + (1) \cdot \begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}$$

(h) [2 pts] Recall Monte Carlo Tree Search, where UCT (upper confidence bound applied to trees) is used to determine the next best rollout. Which of the following terms, in combination, are necessary to determine the upper confidence bound at a particular node s? Select all that apply.

- ■ Number of rollouts $N(s)$ at node s.
- ■ Number of wins $U(s)$ from node s.
- ■ Number of rollouts $N(p)$ at the parent p of node s.
- ☐ The number of rollouts $N(c_i)$ from **each** child c_i of node s.
- ○ None of the above.

(i) [1 pt] Suppose we use gradient descent to find the value of x that minimizes the loss function $L(x) = x^2$. The gradient is $\frac{\partial L}{\partial x} = 2x$. What is the smallest value of the learning rate for which gradient descent does **not** converge (assuming we do not start at the minimum $x = 0$)?

$\boxed{1}$

Q2. [15 pts] Search & Games: Shelving Books

At the library, Carolyn plans to place 36 books on a bookshelf. The bookshelf consists of 6 rows, labeled Row 0 to Row 5, and each row can hold exactly 6 books.

The 36 books are labeled from b_0 to b_{35}. Furthermore, each spot on the bookshelf is labeled from 0 to 35. Here is a diagram of what the bookshelf looks like with all the books in their correct positions:

Row 0	b_0	b_1	b_2	b_3	b_4	b_5
Row 1	b_6	b_7	b_8	b_9	b_{10}	b_{11}
Row 2	b_{12}	b_{13}	b_{14}	b_{15}	b_{16}	b_{17}
Row 3	b_{18}	b_{19}	b_{20}	b_{21}	b_{22}	b_{23}
Row 4	b_{24}	b_{25}	b_{26}	b_{27}	b_{28}	b_{29}
Row 5	b_{30}	b_{31}	b_{32}	b_{33}	b_{34}	b_{35}

Carolyn first places all 36 books onto the bookshelf in a random order.

Carolyn can move books around using these three actions (each action costs 1):

1. Shift all books in any row to the right by 1, with the rightmost book moving to the first spot on that row.
   - For instance, using this action on Row 3, $\{b_{18}, b_{19}, b_{20}, b_{21}, b_{22}, b_{23}\}$, turns the row into $\{b_{23}, b_{18}, b_{19}, b_{20}, b_{21}, b_{22}\}$.
2. Shift all books in any row to the left by 1, with the leftmost book moving to the last spot on that row.
   - For instance, using this action on Row 4, $\{b_{24}, b_{25}, b_{26}, b_{27}, b_{28}, b_{29}\}$, turns the row into $\{b_{25}, b_{26}, b_{27}, b_{28}, b_{29}, b_{24}\}$.
3. Swap any two books that are on **different** rows.
   - For instance, the books at positions 6 and 22 can be swapped, and so can the books at positions 25 and 31, but the books at positions 24 and 26 **cannot** be swapped.

Carolyn's goal is to **sort** the books by putting all the books in their correct positions. In other words, for $0 \leq i \leq 35$, Carolyn wants book b_i to be placed on the ith position on the bookshelf.

Carolyn decides to model this as a search problem.

(a) [2 pts] Carolyn claims that the size of the state space for this problem is 2^{36}, while a coworker Julia claims that it is 36! instead. Who is correct, and why? Explain your choice in two sentences or fewer.

○ Carolyn is correct [✓ Julia is correct] ○ Neither are correct

Julia is correct, every book's specific position must be known in order to perform the right actions.

(b) [4 pts] Derive the maximum branching factor for this problem. Your answer must written as a single integer.

Hint 1: How many ways are there to swap two books on different rows? How many possible ways are there to shift any of the rows?

Hint 2: The choose function, $\binom{n}{k} = \frac{n!}{k!(n-k)!}$, may be useful here (though you can also solve this question without it).

552

One way to go about this problem is by first accounting for all possible swaps. The way this is done is by computing all possible swaps between two books, $\binom{36}{2} = 630$, and then subtracting all the ways we can swap two books from the same row, $6 \cdot \binom{6}{2} = 90$, which will give us all possible swaps between two books on **different** rows, resulting in 540. Then, we add 12, accounting for all the shifts that can be done (one forward and one backward shift for each of the 6 rows), resulting in 552.
Furthermore, you could also approach this problem by choosing any book, so 36 choices, and then choosing any of the 30 books that book can successfully swap with (because we don't want our book to swap with a book from the same row), so 1080. We then divide by 2 to prevent double-counting, and adding 12 for the same reason as before yields 552.

(c) [1 pt] Another coworker Nawoda claims that the following goal test is valid for this search problem.
(Reminder: the goal state is when every book is at its correct position on the bookshelf.)

- For the leftmost book b_i of every row, book b_{i+1} is to its direct right.
- For the rightmost book b_i of every row, book b_{i-1} is to its direct left.
- For each book b_i on **neither** end of a row, book b_{i-1} is to its direct left, and book b_{i+1} is to its direct right.

○ This is a valid goal test. [✓ This is **not** a valid goal test.]

Consider the case in which almost every book is sorted, but two complete rows of books are swapped. In the goal test stated, it will incorrectly say that we have reached the goal state.

(d) [3 pts] Which of the following heuristics are admissible for this search problem? Select all that apply.

- ☐ The minimum number of **swaps** needed to sort all the books, assuming you can swap **any** two books, regardless of whether they are on the same row.
- ☐ The number of books that are **not** in their correct position.
- ☐ For some book b_i, define dist(b_i) to be the number of rows plus the number of columns book b_i is from its correct position. The heuristic is max(dist(b_i)) for all books b.
- ● None of the above.

The first heuristic is inadmissible, as we can consider the case in which almost all the books are at their correct position, but the first row's books are all shifted one to the right of where they're supposed to be. The minimum number of swaps would give a cost larger than the true cost, which would just be shifting the books in the first row to the left once.
The second heuristic is also inadmissible. We can picture the exact case talked about above. If one row is shifted to the right and everything else is correct, the number of books that aren't in their correct shelf position would be an overestimate of the true cost (just 1, shift all the books in that row to the left by 1).
Finally, the third is inadmissible. Manhattan distance is irrelevant here, as you can swap two books from different rows that are really far away from each other at only the cost of 1, so the distance a book is from where it's supposed to be does not matter for this search problem.

Now consider the following game: Julia and Carolyn take turns making moves on the same bookshelf until all the books are in their correct position. The person who makes the move that results in the bookshelf being completely sorted wins. Assume both players are playing optimally.

(e) [2 pts] Julia and Carolyn decide to model this game with a game tree. Select all true statements.

- ■ This game tree is infinite in depth.
- ☐ This game tree requires expectation nodes.
- ■ This is a zero-sum game.
- ■ Every non-terminal node's branching factor will be equal to the game's maximum branching factor.
- ○ None of the above.

This game tree is infinite in depth because since both players are acting optimally, no player will ever make a move that causes the bookshelf to be ONE move away from being completely sorted, because then the other player automatically wins the game.
The game tree does NOT require expectation nodes, as minimizer, maximizer, and terminal nodes are all we need.
This indeed is a zero-sum game, as each player's gain is directly equivalent to the other player's loss.
Since there are no state-dependent restrictions on what moves can be played, we can perform any of the possible actions on any of the non-terminal states.

(f) [3 pts] Julia is trying to maximize her utility. Fill in the boxes below with integer values such that $f(s)$ is an evaluation function that causes Julia to act optimally.

$$f(s) = \begin{cases} \boxed{-1} & \text{if state } s \text{ is } \boxed{1} \text{ move(s) away from a state where all the books are sorted.} \\ \boxed{0} & \text{otherwise.} \end{cases}$$

The main idea for this problem is that since Carolyn is playing optimally, Julia should NEVER move into a state that is one move away from being a terminal state, because then Carolyn will make the move that causes the bookshelf to be completely sorted. Therefore, we want our evaluation function to equate to some value a when we are at a state that is one move away from being terminal, and some other value b otherwise, where $a < b$. Therefore, -1 and 0 could both work, but all we need is that the first value is strictly less than the second.

Q3. [15 pts] Logic: Easter Island Revisited

To recap from the midterm: Matei is an elf on Easter Island who is discontent with the current leadership of Pietru the Rotund. In preparation for the election outcome on May 9th, Matei has decided to predict the politics of Easter Island using logic.

Matei has quickly gathered that there are only 5 voting members on Easter Island—Abby (A), Brad (B), Charles (C), Datsu (D), Ershawn (E)—who determine the outcome of the election.

Let the proposition P_d represent the decision d made by voter P in an election. A voter may either vote YES, NO, or ABSTAIN, i.e. P_y = True, P_n = True, or P_a = True, respectively.

(a) [3 pts] What is the constraint that represents the following statement?

"Brad must either vote YES, NO, or ABSTAIN, **and can only choose one of these options**."

Write your answer in Disjunctive Normal Form (DNF) using a combination of the following symbols: $\wedge$ (and), $\vee$ (or), $\neg$ (not), B_y, B_n, B_a. Parentheses—"(" and ")"—may also be used.

$$(\neg B_y \wedge \neg B_n \wedge B_a) \vee (B_y \wedge \neg B_n \wedge \neg B_a) \vee (\neg B_y \wedge B_n \wedge \neg B_a)$$

(b) [4 pts] Convert the following statement to a logical expression:

"If Brad votes NO, then either Charles abstains or Ershawn votes YES, but not both."

Write your answer in Conjunctive Normal Form (CNF) using a combination of the following symbols: $\wedge$ (and), $\vee$ (or), $\neg$ (not), A_y, A_n, A_a, ..., E_y, E_n, E_a. Parentheses—"(" and ")"—may also be used.

Hint 1: $A \bigoplus B$ is a function that evaluates to True if A and B are opposite truth values, or False if they are the same.

Hint 2: $A \bigoplus B$, can be represented as $(A \vee B) \wedge (\neg A \vee \neg B)$.

$$(\neg B_n \vee C_a \vee E_y) \wedge (\neg B_n \vee \neg C_a \vee \neg E_y)$$

Subparts (c)-(f) are unrelated to the previous subparts.

(c) [2 pts] Which options should be picked in the square brackets below to make the description true?

The DPLL algorithm is used for checking whether a sentence in CNF form has any assignment of values to variables that makes the sentence [*true / false*]. As the algorithm progresses, we evaluate partial models with [*more / fewer*] assignments of values to variables.

[✓ true; more] ○ false; more ○ true; fewer ○ false; fewer

Depending on the sentence and the partial model, we might employ different techniques for speeding up the DPLL algorithm.

(d) [2 pts] Select the CNF sentence and partial model such that the DPLL algorithm can output True early.

- [✓ $(A \vee \neg B) \wedge (\neg B \vee C) \wedge (\neg A \vee C)$. Partial model: {B: false, C: true}]
- ○ $(A \vee \neg B) \wedge (\neg B \vee C) \wedge (\neg A \vee C)$. Partial model: {B: false}
- ○ $(A \vee B) \wedge (B \vee C) \wedge (A \vee C)$. Partial model: {B: true}
- ○ $(A \vee \neg A) \wedge (B \vee \neg B) \wedge (C \vee \neg C)$. Partial model: {A: true}

(e) [2 pts] Select the CNF sentence and partial model such that the DPLL algorithm can use the **Pure Symbol** trick to update the partial model for the next iteration.

- ○ $(A \vee \neg B) \wedge (B \vee \neg C) \wedge (\neg A \vee C)$. Partial model: {}
- [✓ $(A \vee \neg B) \wedge (\neg B \vee C) \wedge (\neg A \vee C)$. Partial model: {C: true}]
- ○ $(A \vee \neg A) \wedge (B \vee \neg B) \wedge (C \vee \neg C)$. Partial model: {C: true}
- ○ $(A \vee B \vee C \vee D) \wedge (\neg A \vee \neg B) \wedge (\neg C \vee \neg D)$. Partial model: {A: true, C: true}

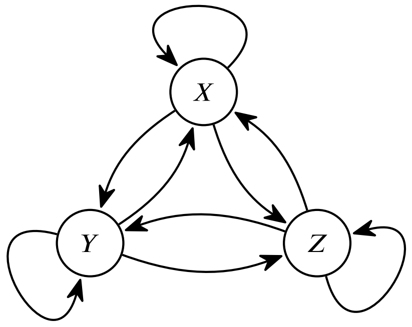
(f) [2 pts] Select the CNF sentence and partial model such that the DPLL algorithm can use the **Unit Clause** trick to update the partial model for the next iteration.

- ○ $(\neg A \vee \neg B \vee \neg C) \wedge (A \vee \neg D \vee \neg E)$. Partial model: {}
- ○ $(A \vee B \vee C \vee D \vee E \vee F) \wedge (\neg A \vee \neg B)$. Partial model: {}
- ○ $(A \vee B \vee C) \wedge (\neg A \vee \neg B \vee \neg C) \wedge (A \vee B \vee \neg C)$. Partial model: {A: true}
- [✓ $(A \vee \neg B) \wedge (\neg B \vee C \vee D) \wedge (\neg A \vee C \vee \neg D)$. Partial model: {B: true}]

Q4. [10 pts] MDPs: Blinky's Hamster

Blinky has a hamster that runs between 3 hamster wheels, labeled X, Y, and Z. At each timestep t, the hamster is at one of the wheels. Between timestep t and $t+1$, the hamster either stays at its current wheel or moves to one of the other two wheels.

We can model the hamster's movement as a Markov process, where S_t represents the hamster's location at timestep t.



	$S_{t+1}=X$	$S_{t+1}=Y$	$S_{t+1}=Z$
$S_t=X$	0.25	0.50	0.25
$S_t=Y$	0.25	0.25	0.50
$S_t=Z$	0.50	0.25	0.25

For the next two subparts, use the above table of transition probabilities for $T(S_{t+1}|S_t)$. Suppose we know $S_2 = X$, i.e. the hamster is in location X at $t = 2$.

(a) [1 pt] Calculate $P(S_3 = Y)$.

0.5 or 1/2

Note: During the exam, a clarification was given that the exact expression we were looking for was $P(S_3 = Y|S_2 = X)$.

Solution: $P(S_t + 1 = Y|S_t = X) = 0.5$.

(b) [2 pts] Calculate $P(S_4 = Y)$.

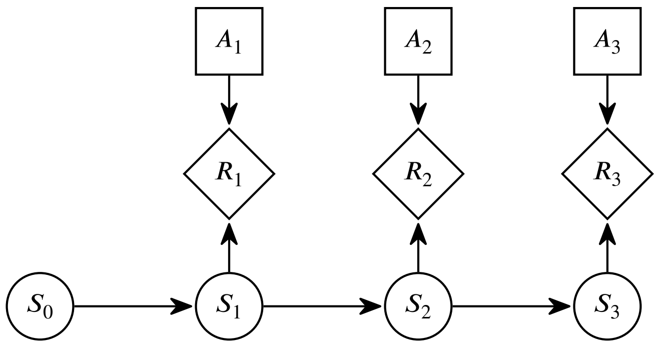
0.3125 or 5/16

Note: During the exam, a clarification was given that the exact expression we were looking for was $P(S_4 = Y|S_2 = X)$.

Solution: $P(S_4 = Y|S_2 = X) = \sum_w P(S_4 = Y|P_3 = w)P(S_3 = w|S_2 = X)$.

$S_3 = X : 0.25 \times 0.5$,
$S_3 = Y : 0.5 \times 0.25$,
$S_3 = Z : 0.25 \times 0.25$,
Summed together, 0.3125.

Consider the following game. At every timestep, Blinky chooses to look inside wheel X, Y, or Z. Let A_t represent Blinky's choice of wheel at time t. At every timestep t, if Blinky finds the hamster, Blinky gets a reward of 1 (i.e. $R_t = 1$ if $A_t = S_t$). Otherwise, the reward is zero.



(c) [1 pt] Blinky's observation at time t consists of A_t (Blinky's choice of wheel) and R_t (the reward at that time step).

Does this observation qualify as a state that obeys the Markov property? In other words, is it true that (A_t, R_t) is conditionally independent of $(A_{0:t-2}, R_{0:t-2})$, given (A_{t-1}, R_{t-1})?

○ Yes [✓ No]

No, this state does not obey the Markov property. If our current reward $R_t = 0$, then the past will give additional information (for the belief of future states). Knowing that two time steps ago, we recieved reward $R_{t-2} = 1$, lets us solve equations set up the same as in part (b) (and then incorporating the evidence that we see at our current timestep) to have a more informed belief. This is like solving an HMM.

For the rest of the question, assume Blinky does not know the hamster's transition probabilities $T(s_{t+1}|s_t)$.

(d) [3 pts] Suppose Blinky chooses a wheel and receives a reward at each timestep from 1 to K, where $K >> 1$, to learn about the game.

After these initial K steps, Blinky will leave the game for a long time (while the hamster continues to move between wheels). Then, far in the future, at timestep $C >> K$, Blinky returns and wants to find the hamster.

Select all strategies that maximize the probability of choosing the wheel with the hamster at timestep C.

☐ Treat the game as a bandit game with one arm per wheel. Play by **UCB1** for the initial K steps, keeping track of the Q-value for each wheel. Then, at time step C, look inside the wheel with the highest Q-value.

☐ Treat the game as a bandit game with one arm per wheel. Play by **ϵ-greedy with constant ϵ** for the initial K steps, keeping track of the Q-value for each wheel. Then, at time step C, look inside the wheel with the highest Q-value.

[✓] Select wheels randomly for the initial K steps, keeping track of the average reward for each wheel. At time step C, look inside the wheel with the highest average reward.

[✓] Use the initial K steps to estimate $T(s_{t+1}|s_t)$. Solve the steady state equations to find the most likely hamster wheel. At time step C, choose the corresponding wheel.

○ None of the above.

Note that our goal is to essentially find the steady state distribution of the hamster's location, and choose the most common state.

(a) and (b) are incorrect. This problem is not a bandit problem, because the arms are not independent; UCB1 has no guarantees on finding an accurate Q-value. Similarly, epsilon greedy will not find an accurate Q-value.

(c) is true. This models our random pulling at timestep C.

(d) is true. Solving steady state equations for the equilibrium will show the most likely state.

Note: the points per option were 0.5, 0.5, 1, and 1.

In the rest of the question, Blinky wants to estimate $T(s_{t+1}|s_t)$ by playing the game.

Blinky plays the game for a long time, choosing wheels uniformly at random at every time step, and records many samples in the following format: $(A_t, R_t, A_{t+1}, R_{t+1})$. For example, $(X, 0, Y, 1)$ represents Blinky choosing X and receiving a reward of 0, and then choosing Y and receiving a reward of 1 at the next time step.

Blinky writes the following expression:

$$T_{\text{guess}}(S_{t+1} = Y | S_t = X) = \frac{\text{Count}\,((X,1,Y,1))}{\text{Count}\,((X,1,Y,1)) + \text{Count}\,((X,1,Y,0))}$$

where Count$(\cdot)$ denotes the number of matching samples.

(e) [1 pt] Will this expression approach the true value of $T(S_{t+1} = Y | S_t = X)$ as the number of samples approaches infinity? (Assume all transition probabilities are greater than zero.)

[✓ Yes] ○ No

Yes, this is an unbiased estimation. This will equal the true transition function in expectation.

(f) [1 pt] Does the expression below always evaluate to 1? (Note: This subpart uses T instead of T_{guess}.)

$$\sum_{i \in \{X,Y,Z\}} T(S_{t+1} = i \mid S_t = X)$$

[✓ Yes] ○ No

True. This is by definition of transition functions. By iterating over all possible future states i, the combined probabilities must add up to 1.

(g) [1 pt] Does the expression below always evaluate to 1? (Note: This subpart uses T_{guess} instead of T.)

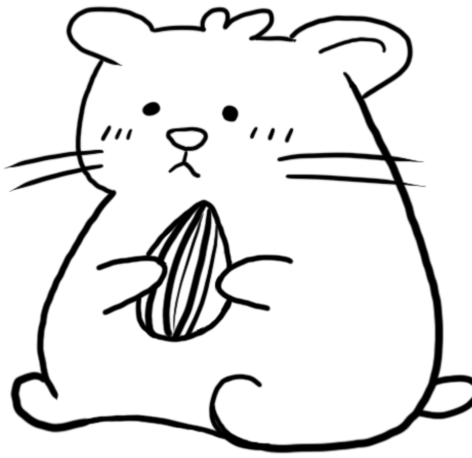
$$\sum_{i \in \{X,Y,Z\}} T_{\text{guess}}(S_{t+1} = i \mid S_t = X)$$

○ Yes [✓ No]

No. Because of variance, it is possible this does not add up to 1. Alternatively, since each estimate for future states are based in independent sets of samples, there is no guarantee that they will together add up to 1.

For example, imagine if by chance, every time we pulled L_1 then L_2, we got reward 1. However, we also tried L_1 followed by L_3, and sometimes got reward. $T_{guess}(s_{t+1} = L_1) = 1$, and so $T_{guess}(s_{t+1}|L_1)$ will not add up to one.

If you did want to use these samples to form a real transition function, you would need to normalize!

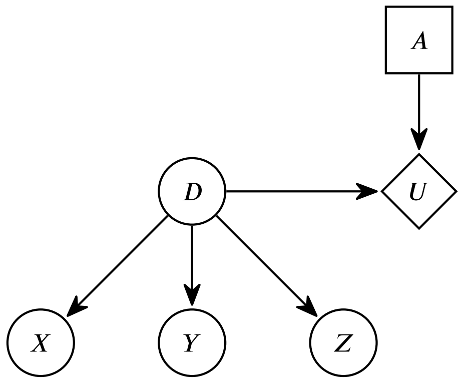


Q5. [16 pts] VPI: Cost of Extra Samples

Consider the following game. There are two doors, Left and Right. One of the doors contains a prize, and you want to choose the door with the prize.

Consider the following decision network:

- D: Random variable indicating which door (Left or Right) has the prize. The prize is equally likely to be behind each door.
- A: Action indicating which door (Left or Right) the player chooses to open.
- X, Y, Z: Three independent, imperfect sensors indicating which door the prize is behind. Each sensor is correct 70% of the time, and incorrect 30% of the time.
- U: The player's utility is 100 if they choose the door with the prize, and 0 if they choose the door without the prize.



D	A	U
Left	Left	100
Left	Right	0
Right	Left	0
Right	Right	100

D	$P(D)$
Left	0.5
Right	0.5

X	D	$P(X\|D)$
Left	Left	0.7
Left	Right	0.3
Right	Left	0.3
Right	Right	0.7

Similar tables for $P(Y|D)$ and $P(Z|D)$ omitted.

(a) [2 pts] From the above tables, we can derive $P(D|X)$, shown below.

D	X	$P(D\|X)$
Left	Left	0.7
Left	Right	0.3
Right	Left	0.3
Right	Right	0.7

Select all true statements.

■ This table can be derived using Bayes' rule.
■ This table's values would be different if the prize was not equally likely to be behind either of the two doors.
■ This table's values would be different if the sensor readings were accurate 80% of the time instead.
☐ This table's values would be different if the utility for guessing correctly was different.
○ None of the above.

We can use Bayes' rule to derive this table:

$$P(D|X) = \frac{P(X|D)P(D)}{P(X)} = \frac{P(X|D)P(D)}{\sum_d P(X|d)P(d)} = \frac{P(X|D)P(D)}{P(X|D=\text{Left})P(D=\text{Left}) + P(X|D=\text{Right})P(D=\text{Right})}$$

Note that in the denominator, $P(D = \text{Left}) = P(D = \text{Right}) = 0.5$, and in the numerator, $P(D) = 0.5$, so we can factor out 0.5 to simplify this to:

$$P(D|X) = \frac{P(X|D)}{P(X|D=\text{Left}) + P(X|D=\text{Right})}$$

Note that this trick was only possible because the prize was equally likely to be between two doors, i.e. $P(D) = 0.5$ for both values of D.

At this point, we can start plugging in values for X and D.

Note that if you plug in different values of $P(X|D)$ (i.e. sensor readings are accurate 80% of the time instead), then this expression would output different values.

The Utility has no affect on the probability table.

An intuitive interpretation of this table would be: Given a sensor reading, the prize location matches the sensor reading (i.e. the reading is correct) 70% of the time.

For the rest of the question, suppose that we learn that sensor X has value Left.

(b) [2 pts] What is EU(Right$|X$ = Left), the expected utility of choosing action Right, given this sensor reading?

$\boxed{30}$

$$\begin{aligned}\text{EU}(\text{Right}|X=\text{Left}) &= P(D=\text{Left}|X=\text{Left})\cdot U(D=\text{Left},A=\text{Right})\\ &\quad + P(D=\text{Right}|X=\text{Left})\cdot U(D=\text{Right},A=\text{Right})\\ &= (0.7\cdot 0)+(0.3\cdot 100)\\ &= 30\end{aligned}$$

Intuitively, the sensor is right 30% of the time, so expected reward is 30.

(c) [2 pts] What is MEU(X = Left), the maximum expected utility, given this sensor reading?

$\boxed{70}$

$$\begin{aligned}\text{EU}(\text{Left}|X=\text{Left}) &= P(D=\text{Left}|X=\text{Left})\cdot U(D=\text{Left},A=\text{Left})\\ &\quad + P(D=\text{Right}|X=\text{Left})\cdot U(D=\text{Right},A=\text{Left})\\ &= (0.7\cdot 100)+(0.3\cdot 0)\\ &= 70\end{aligned}$$

Intuitively, the sensor is right 70% of the time, so expected reward is 70.

Choosing action Right would give expected utility of 30, which is worse.

Intuitively, given the sensor reading of Left, you should follow the sensor reading and choose Left. Since the sensor reading will be correct 70% of the time, you will gain the 100 utility 70% of the time, for a maximum expected utility of 70.

From the tables in the decision network, we can derive $P(D|X=\text{Left},Y,Z)$, shown below.

	D	Y	Z	$P(D\|X=\text{Left},Y,Z)$
(i)	Left	Left	Left	0.927
(ii)	Right	Left	Left	0.073
(iii)	Left	Left	Right	0.7
(iv)	Right	Left	Right	0.3
(v)	Left	Right	Left	0.7
(vi)	Right	Right	Left	0.3
(vii)	Left	Right	Right	0.3
(viii)	Right	Right	Right	0.7

(d) [3 pts] What is VPI($Y,Z|X$ = Left), the value of learning the other two sensor readings, rounded to the nearest integer?

Hint: Our solution uses the probabilities in rows (i), (iii), (v), and (viii) in the table above, along with values from the utilities table.

$\boxed{\text{5 or 6}}$

For each instantiation of evidence, we can get the MEU by multiplying $P(D|X,Y,Z)$ from the table (the probability the prize is behind the door we picked), by 100 (the utility of picking the prize). This gives us the four values of:

92.7 (pick Left when all three sensors say Left, correct 0.927 of the time)

70 (pick Left when X and Y say Left, but Z says Right, correct 0.7 of the time)

70 (pick Left when X and Z say Left, but Y says Right, correct 0.7 of the time)

70 (pick Right when X says Left, but Y and Z say Right, correct 0.7 of the time)

We can average this MEU values (as mentioned in an earlier subpart). The four numbers sum to 302.7. Divide by 4 to get roughly 75.

This tells us that the MEU, given all three sensors, is roughly 75.

From the earlier subparts, we found that the MEU, given only a single sensor, is 70.

The gain in MEU from learning two extra sensor readings is approximately 5, exact is 5.675

(e) [1 pt] The structure of D, X, Y, and Z in the decision network looks similar to a Naive Bayes structure. Which of the following best explains why?

- ○ The sensor readings are independent.
- [✓ The sensor readings are conditionally independent, given the prize location.]
- ○ The sensor readings are the training data, and the prize location is the test data.
- ○ The sensor readings are the test data, and the prize location is the training data.

The first option is a false statement.

The bottom two options are false because there's no notion of training data and test data in a decision network.

(f) [2 pts] Suppose you and your friend each play the game once. You play the game with access to one sensor reading, and your friend plays the game with access to all three sensor readings. Is it possible that you receive more utility than your friend?

- [✓ Yes, because VPI is an expectation and does not guarantee anything about a particular game.]
- ○ Yes, because the two extra sensor readings never result in additional utility.
- ○ No, because VPI is always non-negative.
- ○ No, because the two extra sensor readings mean that your friend has strictly more information than you.

Yes, because the outcomes (0 or 100) are non-deterministic. You could get utility 100 with one sample, and your friend could get utility 0 with three samples.

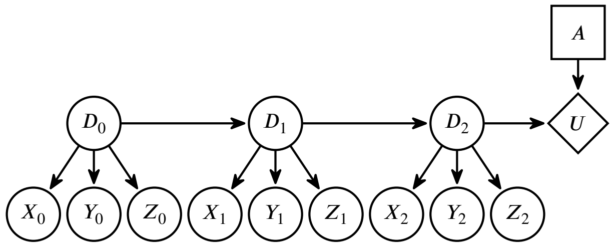
Now, consider a different game setting: the prize is revealed to be a hamster! There are still two doors, but the hamster can move between doors on each time step. You can use the sensor to try and locate the hamster for a few time steps, before making a decision.

At time step $t = 0$, the hamster is hiding behind one of the doors. At each time step, the hamster might move behind the other door, or stay behind the same door. At time step $t = 2$, you want to guess where the hamster is, and you win a prize for guessing correctly.

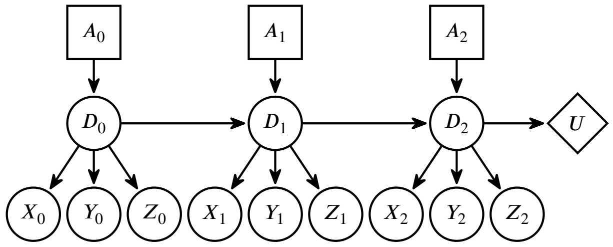
As before, there are three independent, imperfect sensors showing which door has the prize. If you have access to a sensor, you can read its value once per time step ($t = 0$, $t = 1$, and $t = 2$).

X_t represents the value of the sensor X at time t (and similar for Y_t, Z_t).

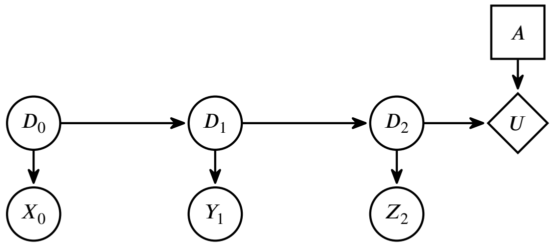
(g) [2 pts] Which of the following decision network structures correctly models the game described?



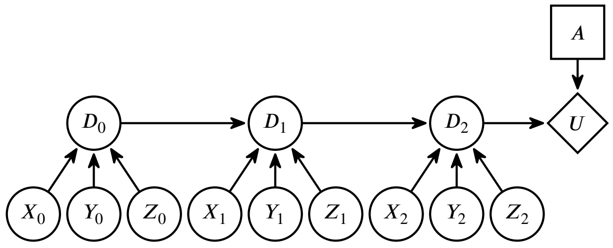
[✓ Network (i)]



Network (ii)



Network (iii)



Network (iv)

Network (ii) is incorrect because we don't have multiple choices of actions. We just pick the door at the end of the game.

Network (iii) is incorrect because it does not model each of the three sensors producing three readings (one per time step).

Network (iv) is incorrect because it breaks the assumption that the sensors are independent, given the prize location.

(h) [2 pts] Write a VPI expression for the value of having access to two additional sensors Y and Z during this game, given that you already have access to sensor X.

VPI (	)

$VPI(Y_0, Z_0, Y_1, Z_1, Y_2, Z_2 | X_0, X_1, X_2)$

Q6. [11 pts] RL: Optimistic Q-Learning

Consider a Markov Decision Process where the reward function is bounded between $-R$ and R, inclusive. Pacman suggests we use optimistic initialization to perform Q-learning on our MDP.

Recall that the online Q-learning algorithm uses the following update rule: $Q(s, a) \longleftarrow (1-\alpha)Q(s, a)+\alpha \left(r + \gamma \cdot \max_{a'} Q(s', a')\right)$.

(a) [2 pts] What is the highest Q-value that can possibly be achieved in the MDP? *Hint:* $\sum_{i=0}^{\infty} c^i = \frac{1}{1-c}$, for $|c| < 1$.

$$\boxed{\frac{R}{1-\gamma}}$$

(b) [1 pt] Sometimes MDPs can be sparse, which means that nonzero rewards are given very rarely.

For this subpart only, suppose that reward in our MDP is given only when transitioning to terminal states.

What is the highest Q-value that can possibly be achieved in this MDP?

$$\boxed{R}$$

(c) [2 pts] Select all true statements about comparing online Q-Learning with different forms of Q-value initialization.

- [✓] Optimistic initialization could be implemented by setting $Q(s, a)$ to the maximum possible Q-value in the MDP for every state-action pair (s, a).
- ☐ Q-learning with optimistic initialization is guaranteed to converge to the optimal policy with a finite number of samples.
- [✓] Q-values are more likely to be updated downward with optimistic initialization, than with zero initialization.
- ☐ Zero initialization is likely to perform better than optimistic initialization when the rewards are sparse.
- ○ None of the above.

(d) [2 pts] Suppose we use online Q-learning with optimistic initialization and generate samples in an MDP with a purely greedy policy (i.e. the agent always selects actions with the highest Q-values and breaks ties randomly). You can assume an initial learning rate $\alpha < 1$ that decays exponentially over time.

Is the agent guaranteed to find to the optimal policy for the MDP after it has seen an infinite number of transitions?

- ○ Yes, because when the policy selects suboptimal actions, their Q-values will eventually be decreased.
- ○ Yes, because the policy by definition selects the action with the highest Q-value.
- [✓] No, because rare transitions may cause the agent to underestimate the Q-value of the *optimal* action.
- ○ No, because new samples will always cause the policy to change.

(e) [2 pts] Which of the following are forms of optimism?

- [✓] Admissible heuristics
- ☐ Alpha-beta pruning
- [✓] UCB1
- ☐ Gibbs sampling
- ○ None of the above.

(f) [2 pts] This subpart is about exploration and exploitation. Suppose we have taken some actions and collected a finite number of samples in an arbitrary MDP.

In which of the following scenarios is it possible for an agent following a purely greedy policy to ignore explored state-action pairs (s, a)?

- ☐ When Q-values are initialized to zero and rewards are always positive.
- [✓] When Q-values are initialized to zero and rewards are always negative.
- [✓] When Q-values are initialized optimistically and rewards are always positive.
- [✓] When Q-values are initialized optimistically and rewards are always negative.
- ○ None of the above.

Q7. [17 pts] ML: Spam Filter

Pacman has hired you to work on his PacMail email service. You have been given the task of designing a spam detector.

You are given a dataset of emails X, each with labels Y of "spam" or "ham." Here are some examples from the dataset.

Spam Email Ex. 1: "WINNER!! As a valued network customer you have been selected to receive a $900 prize reward!!!"

Spam Email Ex. 2: "We are trying to contact you. Last weekend's draw shows that you won a £1000 prize GUARANTEED!!!"

Ham Email Ex. 1: "Hey! Did you want to grab coffee before the team meeting on Friday?"

Ham Email Ex. 2: "Thank you for attending the talk this morning. I've attached the presentation for you to share with your team. Please let me know if you have any questions."

Your job is to classify the emails in a second dataset, the test dataset, which do not have labels.

(a) [3 pts] Considering only the examples given, which of the following features, in isolation, would be sufficient to classify the examples correctly using a linear classifier?

- ☐ The number of words in the email.
- [✓] The number of times the exclamation point ("!") appears in the email.
- [✓] The number of times "prize" appears in the email.
- [✓] The number of capital letters in the email.
- ○ None of the above.

(b) [2 pts] Select all true statements about using naive Bayes to solve this problem.

- [✓] We assume that each feature is conditionally independent of the other features, given the label.
- [✓] Given that the prior (class) probabilities are the same, the probability of classifying an email as "spam", given the contents of the email, is proportional to the probability of the contents, given that the email is labeled "spam".
- ☐ Including more features in the model will always increase the test accuracy.
- [✓] Naive Bayes uses the maximum likelihood estimate to compute probabilities in the Bayes net.
- ○ None of the above.

(c) [3 pts] You want to determine whether naive Bayes or logistic regression is better for your problem. Select all true statements about these two methods.

- [✓] Logistic regression requires fewer learnable parameters than naive Bayes, assuming the same features.
- ☐ Both logistic regression and naive Bayes use the same independence assumption.
- [✓] Both logistic regression and naive Bayes can be used for multi-class classification.
- [✓] Logistic regression models the conditional class distribution $P(Y|W)$ directly, whereas naive Bayes models the joint distribution $P(Y,W)$.
- ○ None of the above.

You decide to use binary bag-of-words (see definition below) to extract a feature vector from each email in the dataset.

Binary bag-of-words: given a vocabulary of N words, bag-of-words represents a string as an N–element vector, where the value at index i is 1 if word i appears in the string, and 0 otherwise.

The next 3 subparts (d)-(f) are connected. After training a naive Bayes model with **binary bag-of-words** features, you compute the following probability tables for $P(W = w_i|Y = y)$, where w_i is the ith word in your vocabulary. Assume that there are no other words in your model's vocabulary.

	"hey"	"valued"	"team"	"share"
Spam	0.25	0.6	0.3	0.8
Ham	0.4	0.1	0.5	0.3

Now you are tasked with classifying this new email:

"Hey, what time is our team meeting? Can't wait to share with the team!!!"

(d) [1 pt] Fill in the following table with integers corresponding to the bag-of-words vector $\mathbf{w}$ for the new email. Ignore punctuation and capitalization.

	"hey"	"valued"	"team"	"share"
w	1	0	1	1

As we are using binary bag-of-words, "team" is only recorded once.

(e) [3 pts] Compute the probability distribution $P(Y, W = \mathbf{w})$ for this email. You may assume the prior probabilities of each class are equal, i.e. $P(Y = \text{spam}) = P(Y = \text{ham}) = 0.5$. You may ignore words in the sentence that are not present in our model's vocabulary. Write your answer as a single decimal value, rounded to 3 decimal places.

$P(Y = \text{spam}, W = \mathbf{w}) = \boxed{0.012}$ $P(Y = \text{ham}, W = \mathbf{w}) = \boxed{0.027}$

$$P(Y = \text{spam}, W = \mathbf{w}) = P(Y = \text{spam}) \prod_i P(W = w_i|Y = \text{spam})$$
$$= 0.5 \cdot 0.25 \cdot 0.4 \cdot 0.3 \cdot 0.8$$
$$= 0.012$$

$$P(Y = \text{ham}, W = \mathbf{w}) = P(Y = \text{ham}) \prod_i P(W = w_i|Y = \text{ham})$$
$$= 0.5 \cdot 0.4 \cdot 0.9 \cdot 0.5 \cdot 0.3$$
$$= 0.027$$

(f) [2 pts] To get the conditional class distribution from the joint probabilities in part (e), we normalize $P(Y, W = \mathbf{w})$ by dividing it by some Z, such that $P(Y|W = \mathbf{w}) = \frac{1}{Z}P(Y, W = \mathbf{w})$. Write an expression for Z using any of the following terms: $P(Y = \text{ham}, W = \mathbf{w})$, $P(Y = \text{spam}, W = \mathbf{w})$, $P(Y = \text{ham})$, $P(Y = \text{spam})$, and the integer 1.

$Z = \boxed{P(Y = \text{spam}, W = \mathbf{w}) + P(Y = \text{ham}, W = \mathbf{w})}$

To normalize the joint probability/factor, we sum up all of the joint probabilities over Y.

After training the binary bag-of-words model, you find that the test accuracy is still low.

(g) [1 pt] Instead of treating each word as a feature, you decide to use n-grams of words instead. You then split your labeled dataset into a large training set and a small validation set.

Which of the following is the best way of identifying the optimal value of n for your n-gram model?

- [✓ Train the model on the training data using different values of n; select the n with the highest validation accuracy.]
- ○ Train the model on the training data using different values of n; select the n with the highest training accuracy.
- ○ Select the n that maximizes the number of sequences of n repeated words in the training data.
- ○ Select n to be the average number of characters per word divided by 2.

(h) [2 pts] You apply Laplace smoothing on the bag-of-words data. Select all true statements.

- ☐ Laplace smoothing for bag-of-words always leads to overfitting.
- ☐ Laplace smoothing is applied by subtracting a constant positive value from each word count.
- ☐ Laplace smoothing eliminates the need for a validation set.
- ☐ Laplace smoothing is only useful for large-vocabulary training datasets.
- [✓ None of the above.]