CS188 Summer 2024 — Introduction to Artificial Intelligence — Midterm

- You have 110 minutes.
- The exam is closed book, no calculator, and closed notes, other than one double-sided cheat sheet 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)	

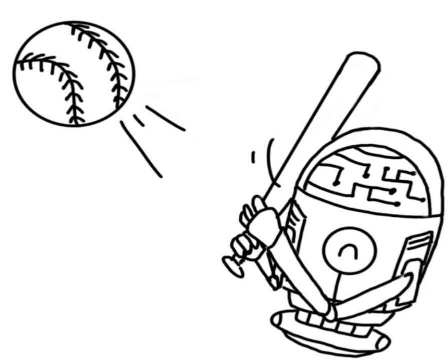
Honor code: "As a member of the UC Berkeley community, I act with honesty, integrity, and respect for others." By signing below, I affirm that all work on this exam is my own work, and honestly reflects my own understanding of the course material. I have not referenced any outside materials (other than 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	20
Q2.	Search: Connected Agents	16
Q3.	CSPs: Staff Dinner	17
Q4.	Games: Make Up Your Mind!	15
Q5.	Bayes Nets: Inverting Arrows	22
Q6.	HMM's: Bayesball	10
	Total	100

We hope you hit it out of the park!



Q1. [20 pts] Potpourri

(a) [3 pts] Select all true statements.

☐ Running A* search with the trivial heuristic on a state space graph will always yield the same path as running greedy search with the trivial heuristic on that state space graph.

☐ The trivial heuristic is consistent for a graph with negative edge weights.

☐ BFS graph search will never expand strictly more nodes than BFS tree search.

☐ DFS graph search on a finite search graph will never get stuck in an infinite search loop.

○ None of the above.

(b) [3 pts] In three sentences or fewer, explain why a heuristic being consistent implies that it is also admissible.

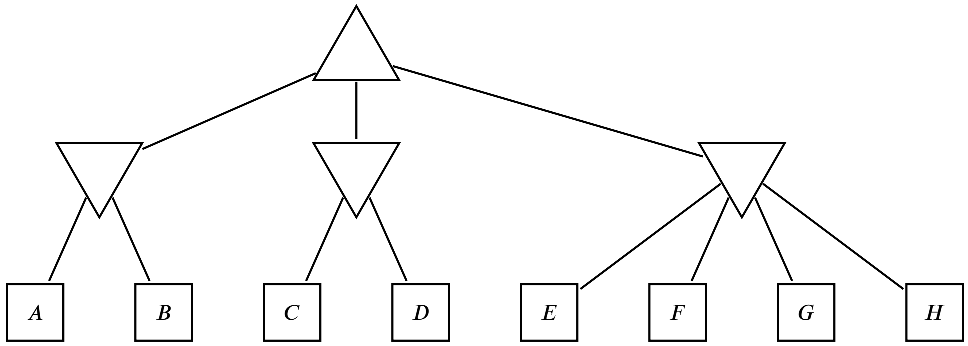
(c) [3 pts] Saner plans to run A* tree search on a graph, and is given a non-negative heuristic $h(x)$ that always underestimates the true cost to the goal. He then defines a new heuristic $h'(x) = c \cdot h(x)$, where c is a constant real value such that $c \neq 0$ and $c \neq 1$.

Compute some value for c that guarantees A* tree search with $h'(x)$ as its heuristic will achieve an optimal path. Note that there are multiple values of c that can work, but you only need to answer with one value.

(d) [2 pts] Rebecca can either earn 10 dollars, or she can enter a lottery where she can either win 100 dollars with probability 50%, or nothing with probability 50%. Select all the utility functions, $U(x)$, that Rebecca can use to be considered risk-averse.

☐ $U(x) = x$ ☐ $U(x) = 5$ ☐ $U(x) = \frac{x}{2}$ ☐ $U(x) = x^2$ ○ None

(e) [3 pts] In the following mini-max game tree, assuming we visit nodes from left to right, which nodes are GUARANTEED to never be pruned?



☐ A ☐ B ☐ C ☐ D ☐ E ☐ F ☐ G ☐ H

◯ None of the above.

We have a Bayes net with the binary variables A, B, D, C, E, and we want to try computing the query $P(D|B = -b, E = -e)$. Select all sampling algorithms that could have generated the three consecutive samples shown given our query.

Note that if a sample is rejected while it is being generated, the rest of the sample will say "rejected".

(f) [2 pts]

Sample 1:	+a	+b	−c	+d	−e
Sample 2:	−a	+b	−c	−d	−e
Sample 3:	−a	+b	−c	−d	−e

☐ Prior Sampling ☐ Likelihood Weighting ◯ None of them

☐ Rejection Sampling ☐ Gibbs Sampling

(g) [2 pts]

Sample 1:	+a	−b	−c	+d	−e
Sample 2:	−a	−b	−c	−d	−e
Sample 3:	−a	+b	rejected	rejected	rejected

☐ Prior Sampling ☐ Likelihood Weighting ◯ None of them

☐ Rejection Sampling ☐ Gibbs Sampling

(h) [2 pts]

Sample 1:	+a	−b	−c	+d	−e
Sample 2:	−a	−b	−c	−d	−e
Sample 3:	−a	−b	−c	−d	−e

☐ Prior Sampling ☐ Likelihood Weighting ◯ None of them

☐ Rejection Sampling ☐ Gibbs Sampling

Q2. [16 pts] Search: Connected Agents

Alice and Bob are both in an $M \times N$ grid, and they each hold on to opposite ends of an *elastic* string, as shown in the diagram below. After each step, Alice and Bob can **each choose to either stay in place, or move up, down, left, or right exactly one square**, while still holding on to the elastic string. The string will always be a straight line, and Alice and Bob **cannot** occupy the same grid at the same time.

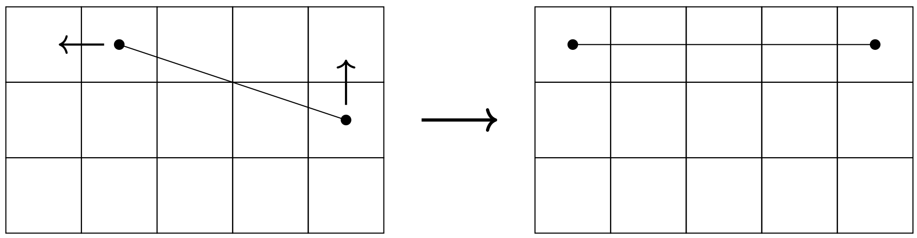


Figure 1: In an example 5x3 grid, assuming that $(1, 1)$ is located at the bottom left square, Alice is located at $(2, 3)$ and Bob is located at $(5, 2)$. After one timestep, Alice moves left while Bob moves up, as shown in the left grid. The resulting state is shown in the right grid.

Alice and Bob want to move around the grid in the fewest amount of turns possible until the string between them **is of length exactly** k, where k is some positive integer less than M and N, AND so that the string is **parallel with the x-axis (so horizontal)**. For example, if $k = 4$, then the diagram on the above right shows Alice and Bob in a goal state. They decide to model this as a search problem.

(a) [2 pts] The length of the string can also be considered as the __________ distance between Alice and Bob.

- ◯ Manhattan
- ◯ Euclidean
- ◯ Taxi-cab
- ◯ Weighted

(b) [2 pts] Compute the size of the state space in terms of M, N, and any constants if necessary.

(c) [2 pts] Compute the maximum branching factor for this problem.

Hint: one timestep results in Alice and Bob each making a move.

(d) [3 pts] Alice claims that there is at most one goal state, while Bob claims there are at most two goal states, because for any goal state, Alice and Bob's position can be flipped around to yield another possible goal state. Who is correct, and why?

Hint: Try running a short demo of the search problem and see what the goal state(s) look like.

◯ Alice is correct ◯ Bob is correct ◯ Neither are correct

(e) [3 pts] For this subpart only, assume that another person Danna also joins this $M \times N$ grid, and so there are now three people on this grid, with three elastic straight-line strings that connect every pair of two people (so three strings in total). The new goal test is now that the three string lengths are all exactly of length k, and none of the strings need to be horizontal.

Alice claims that since there are now three people, the state space must explicitly store the three (possibly different) string lengths, while Bob claims that the state space does NOT need to include this information. Who is correct, and why?

◯ Alice is correct ◯ Bob is correct ◯ Neither are correct

(f) [4 pts] Select all of the admissible heuristics. (x_a, y_a) and (x_b, y_b) are Alice's and Bob's current positions, respectively.

☐ $\frac{||x_b - x_a| - k|}{2}$, rounding down.

☐ Denote $MH((x_a, y_a), (x_b, y_b))$ as the Manhattan distance between Alice and Bob's positions. The heuristic is $|MH((x_a, y_a), (x_b, y_b)) - k|$.

☐ Denote $Eu((x_a, y_a), (x_b, y_b))$ as the Euclidean distance between Alice and Bob's positions. The heuristic is $|Eu((x_a, y_a), (x_b, y_b)) - k|$.

◯ None of the above.

Q3. [17 pts] CSPs: Staff Dinner

Sid is planning dinner events for the course staff of five CS courses this summer! Across the five courses, there are n staff members in total, $S_1, S_2, \ldots, S_n$, that will be going, and each staff member belongs to exactly one of the five CS classes. For a single dinner event, there are 3 possible restaurants to pick from, R_1, R_2, R_3, and 3 possible days for when a dinner will happen, which are Friday, Saturday, and Sunday. Note that all the staff members for a certain CS class will go to the same restaurant on the same day, and you may assume that n is much greater than 5.

Rather than each staff member having their own time/food constraints and preferences, each CS class will instead have their own collective constraints for their respective staff in order to save time. Here are the constraints for each of the five CS classes:

1. CS10's staff only want Friday or Saturday, and only want restaurant R_3.
2. CS61A's staff only want Friday, and are fine with any restaurant.
3. CS61BL's staff want their dinner event to be AFTER or on the SAME day as CS10's staff, and only want restaurant R_2.
4. CS70's staff only want Saturday, but are fine with any restaurant.
5. CS188's staff are fine with any of the three days, but want to go to the same restaurant as CS70's staff.

(a) [2 pts] Before enforcing any of the above constraints (unary and binary), what is the size of the domain for each variable S_i, where $1 \leq i \leq n$?

Sid decides to make this CSP easier to solve by cutting down on the number of variables needed to assign values to; to do this, instead of using n staff members, he uses the following five course numbers as variables, and will then assign the values from these courses to their corresponding staff members:

$$\{\text{CS10}, \text{CS61A}, \text{CS61BL}, \text{CS70}, \text{CS188}\}$$

(b) [4 pts] Assume that the answer to part (a) (domain size for each variable) is d. Fill in the two blanks in the statement below:

After Sid makes this change to the variables, the tightest upper bound it takes to solve this CSP goes from (i) time to (ii) time.

You may use n, d, and any necessary constants to fill in the blanks.

(i) = $O(\quad)$ (ii) = $O(\quad)$

(c) [2 pts] How many binary constraints are in this problem?

(d) [2 pts] The arc consistency algorithm first involves adding every single possible arc in our CSP into a queue. If Sid wanted to enforce arc consistency on this CSP, how many arcs would he start off with in the queue?

○ 5 ○ 10 ○ 20 ○ 40

(e) [3 pts] After enforcing unary constraints, compute the number of remaining values in the domain for each CS class.

CS10: ___ CS61A: ___ CS61BL: ___ CS70: ___ CS188: ___

(f) [4 pts] After enforcing unary constraints, we will solve this CSP using the minimum remaining values heuristic for the ordering of the variables. In the case of a draw using the MRV heuristic, CS10 ≻ CS61A ≻ CS61BL ≻ CS70 ≻ CS188. Also, in addition to the five lines of constraints at the beginning of the problem, an additional constraint is now being added for this CSP:

"A single restaurant and day combination **cannot** host more than one CS course."

For selecting which of the remaining values to assign our variables, $R_1 \succ R_2 \succ R_3$, and Friday ≻ Saturday ≻ Sunday. Furthermore, **apply forward checking** when assigning values.
Using these heuristics, find the value that is assigned to each of the five class variables.

CS10:

◯ (R_1, Friday) ◯ (R_2, Friday) ◯ (R_3, Friday)

◯ (R_1, Saturday) ◯ (R_2, Saturday) ◯ (R_3, Saturday)

◯ (R_1, Sunday) ◯ (R_2, Sunday) ◯ (R_3, Sunday)

CS61A:

◯ (R_1, Friday) ◯ (R_2, Friday) ◯ (R_3, Friday)

◯ (R_1, Saturday) ◯ (R_2, Saturday) ◯ (R_3, Saturday)

◯ (R_1, Sunday) ◯ (R_2, Sunday) ◯ (R_3, Sunday)

CS61BL:

◯ (R_1, Friday) ◯ (R_2, Friday) ◯ (R_3, Friday)

◯ (R_1, Saturday) ◯ (R_2, Saturday) ◯ (R_3, Saturday)

◯ (R_1, Sunday) ◯ (R_2, Sunday) ◯ (R_3, Sunday)

CS70:

◯ (R_1, Friday) ◯ (R_2, Friday) ◯ (R_3, Friday)

◯ (R_1, Saturday) ◯ (R_2, Saturday) ◯ (R_3, Saturday)

◯ (R_1, Sunday) ◯ (R_2, Sunday) ◯ (R_3, Sunday)

CS188:

◯ (R_1, Friday) ◯ (R_2, Friday) ◯ (R_3, Friday)

◯ (R_1, Saturday) ◯ (R_2, Saturday) ◯ (R_3, Saturday)

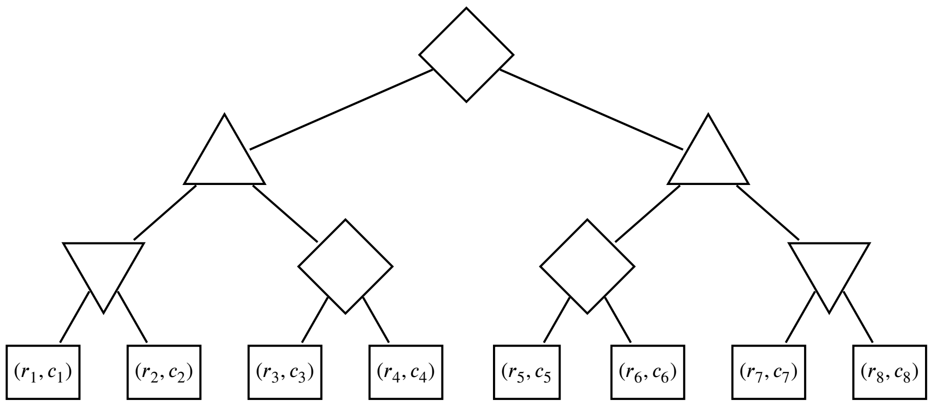
◯ (R_1, Sunday) ◯ (R_2, Sunday) ◯ (R_3, Sunday)

Q4. [15 pts] Games: Make Up Your Mind!

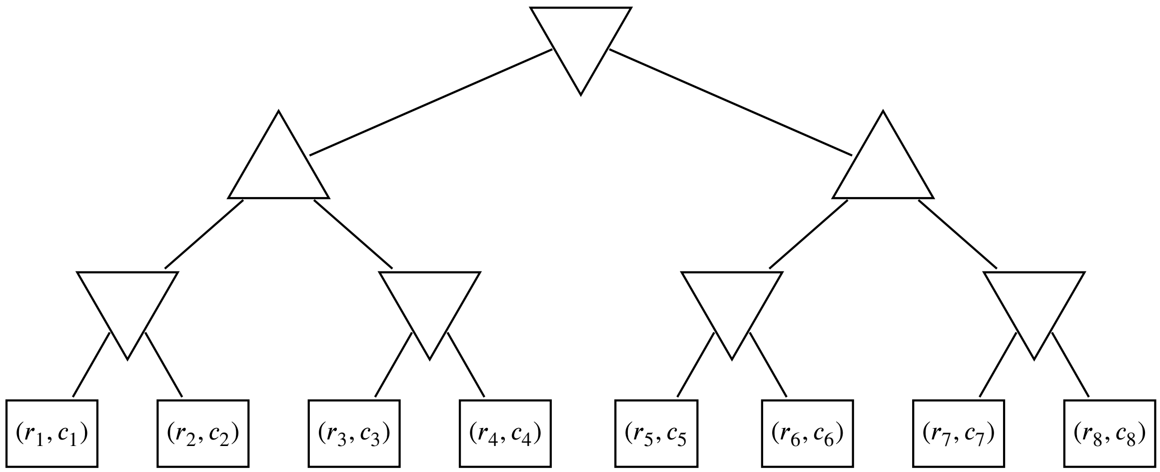
Rohan and Caiden are playing a game. In the corresponding gametree, for every terminal node (r, c), Rohan's utility will be represented as the first of the two leaf numbers, r, and Caiden's utility as the second, c.

Rohan will always choose the node containing the maximum utility for himself. Caiden, on the other hand, will either **choose the node containing the minimum utility for Rohan**, which is denoted by a minimizer node (a triangle pointing down), OR **choose the node according to max$(\mathbf{c} - \mathbf{r})$, where $(\mathbf{r}, \mathbf{c})$ is a leaf node value**, and this is denoted by a diamond node. For instance, if he had to choose between nodes $(2, 5)$ and $(4, 3)$ using the diamond node, he would choose $(2, 5)$ because $5 - 2 = 3$, which is greater than $3 - 4 = -1$. Ties for all action nodes are broken by choosing the LEFTMOST node.

Here is a sample gametree. The two maximizer nodes are controlled by Rohan, while the three diamond nodes and two minimizer nodes are controlled by Caiden. This means Caiden controls the first move, then Rohan, and then Caiden again.



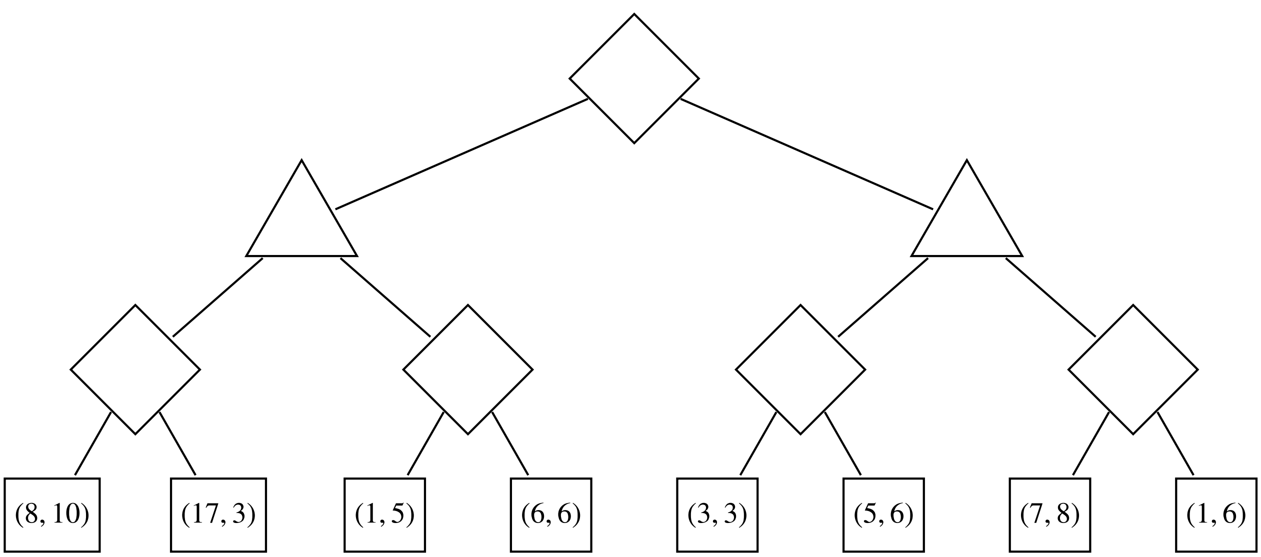
(a) [4 pts] For this subpart only, assume that every single one of Caiden's nodes in the above gametree is a minimizer node (meaning that Caiden is minimizing Rohan's utility, as explained above), as shown in the gametree below. Is pruning possible for this game tree?



○ Yes, pruning is possible.　　　○ No, pruning isn't possible.

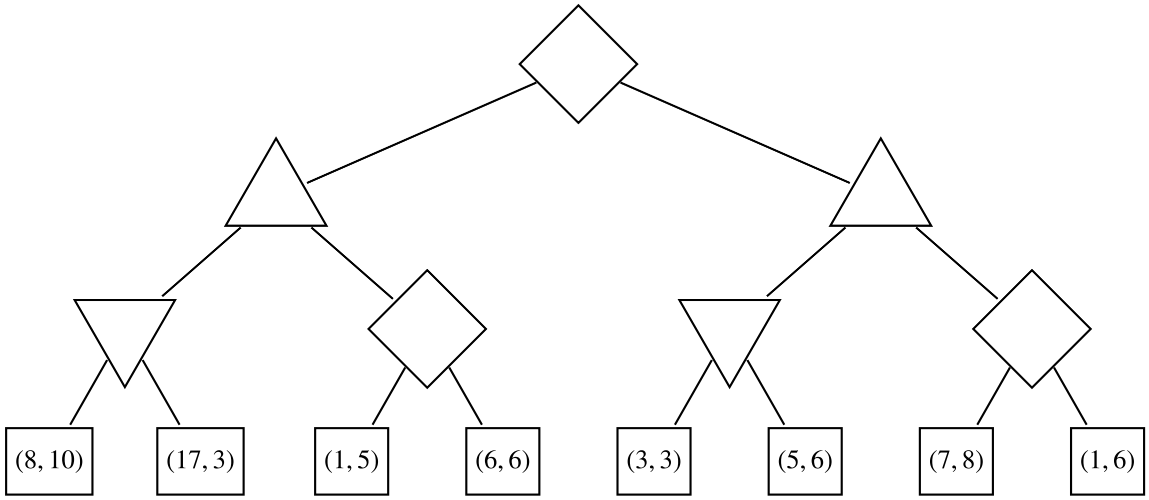
Explain why or why not in three sentences or fewer.

(b) [3 pts] For this subpart only, we now assume that every single one of Caiden's action nodes is a diamond node (as defined above in the problem statement), as shown in the gametree below. Given the leaf values below, fill in the values within the rest of the gametree (fill in each of the **seven** unfilled nodes in the gametree).



(c) [4 pts] We now keep the same leaf values from the previous part in their original positions. Note that Rohan is still always maximizing his own utility on his turns.

It turns out that if we form Caiden's actions like in the gametree below, the value $(8, 10)$ reaches the root node:

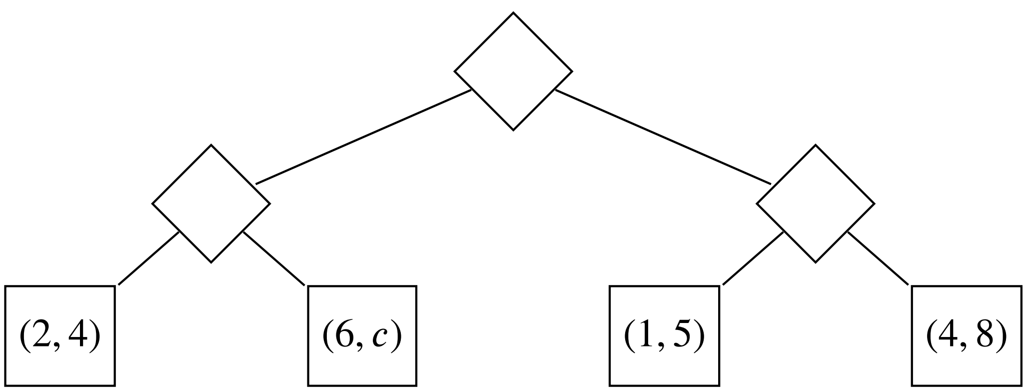


However, there exists *no* combination of diamond/minimizer nodes for Caiden's five actions in the gametree that allows the node $(6, 6)$ to reach the root.

For each of the four right-most leaf nodes, select the nodes for which there exists a combination of diamond and minimizer nodes for Caiden's five nodes that allows them to reach the root node. Note that this means you can change *any* of Caiden's five action nodes to be diamond nodes or minimizer nodes, but Rohan's two maximizer nodes cannot be changed.

☐ $(3, 3)$ ☐ $(5, 6)$ ☐ $(7, 8)$ ☐ $(1, 6)$ ○ None

(d) [4 pts] Given this new gametree where *every* node is a diamond node, compute lower and upper bounds for c such that node $(6, c)$ moves up exactly *once* (so it doesn't end up at the root, but rather in the middle depth of the game tree).



☐ $< c <$ ☐

Q5. [22 pts] Bayes Nets: Inverting Arrows

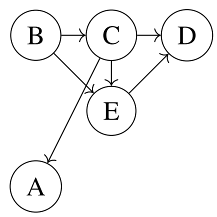
Consider a Bayes net with two nodes, A and B. We can either draw an arrow from A to B, or from B to A.

(a) [3 pts] Suppose we end up choosing to draw an arrow going from A to B.

Amad claims that given only the table $P(B|A)$, one can calculate the table $P(A|B)$. Is he correct?

- ◯ Amad is correct, because Bayes' theorem can calculate $P(A|B)$ given only $P(B|A)$.
- ◯ Amad is correct, because $P(A|B) = 1 - P(B|A)$.
- ◯ Amad is incorrect, because A might be independent of B, so $P(A|B) = P(A)$.
- ◯ Amad is incorrect, because in order to use Bayes theorem, more values are needed.

Now consider the following Bayes net, where every node is a binary variable **except for node** D, which is a ternary variable (can take on any of three values).



(b) [5 pts] Select all arrows that need to be flipped to form a conditional probability table (CPT) that holds the maximum number of entries possible (with these five nodes and six edges). What is the size of this maximal CPT?

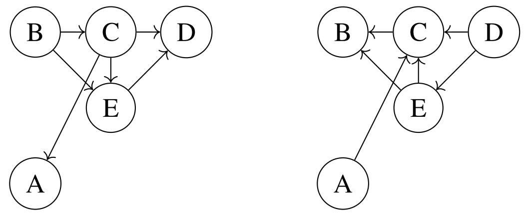
☐ AC ☐ BC ☐ BE ☐ CE ☐ CD ☐ ED

Size of maximal CPT: ☐

(c) [1 pt] Pratush claims that if he inverts any one of the six edges in the original Bayes net above, the resulting graph will still always be a valid Bayes net.

- ◯ He is correct.
- ◯ He is incorrect.

We now focus on the original Bayes net from before, as well as another Bayes net with the same nodes, but every arrow from the original Bayes net is inverted, as shown below.



(d) [6 pts] Our goal is to perform inference by enumeration on both Bayes nets to find corresponding values for $P(C)$.

Recall that in inference by enumeration, we collect all of our probability factors together, and then sum out all of the hidden variables relative to our query, where our query is $P(C)$.

If we were to use inference by enumeration in order to compute $P(C)$ for each Bayes net, would we be multiplying the same conditional probabilities together for both Bayes nets? Remember the original Bayes net is on the left.

○ Yes　　○ No

Fill in the blanks in the expression for inference by enumeration for the **original Bayes net**.

$$\sum_{a\in A}\sum_{b\in B}\sum_{d\in D}\sum_{e\in E} \boxed{}$$

(e) [4 pts] Select the Bayes net(s) that guarantees the following statements to be true.

$D \perp\!\!\!\perp A$

☐ The original Bayes net.　☐ The inverted Bayes net.　○ Neither Bayes net.

$D \perp\!\!\!\perp B|E$

☐ The original Bayes net.　☐ The inverted Bayes net.　○ Neither Bayes net.

(f) [3 pts] Suppose we use variable elimination on both Bayes nets to find $P(C)$ for both Bayes nets instead. Select all of the following true statements. Assume that "size" means the number of rows in the corresponding probability table.

☐ Eliminating E first from the original Bayes net will end up creating a factor with some size s_1. Eliminating E first from the inverted Bayes net will ALSO end up creating a factor with the same size s_1.

☐ Eliminating D first from the original Bayes net will end up creating a factor with some size s_2. Eliminating D first from the inverted Bayes net will ALSO end up creating a factor with the same size s_2.

☐ To compute $P(C)$ for both Bayes nets, variable elimination will always be at least as efficient as inference by enumeration.

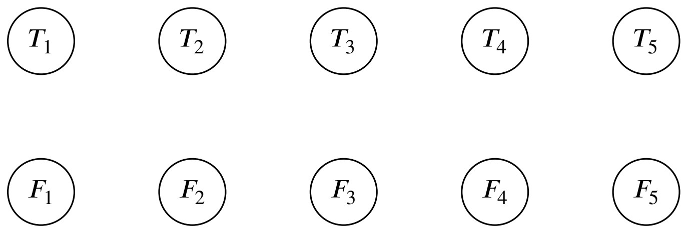
○ None of the above.

Q6. [10 pts] HMM's: Bayesball

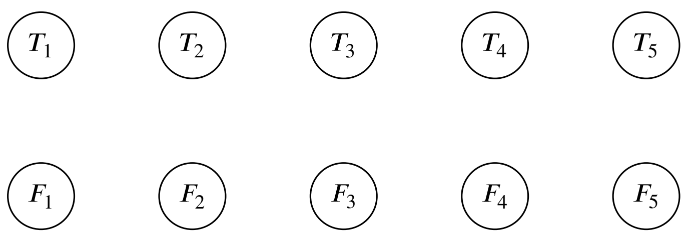
Casey the pitcher is playing baseball, with many of his fans watching him.

After Casey throws the ball, the fans will either boo or cheer, depending on his throw. The sound of the fans also ends up affecting Casey's next **two** throws. In addition to this, every throw by Casey is affected by his throw right before.

(a) [3 pts] Given the nodes $T_1, T_2, \dots T_5$ (which represent Casey's first throw, second throw, all the way to the fifth throw), and the nodes $F_1, F_2, \dots F_5$ (the fans' sound after the first throw, after the second throw, etc.), as shown in the diagram below, draw arrows between these nodes as indicated by the problem.



(b) [3 pts] Casey throws five pitches again (still labeled T_1 to T_5), with the fans still reacting to his throws (again labeled F_1 to F_5). However, after the second throw, Casey realizes that the fans' sound is causing him to lose focus and throw worse. He therefore decides to wear some headphones to block out the noise right after his second throw. This means he is not affected by the fans during his third throw and onwards, but his current throw is still always affected by his throw right before. Draw arrows in the diagram below reflecting this updated situation.



(c) [4 pts] For the above sub-part's Bayes net (where Casey wears headphones right after the second throw), it turns out we can rewrite $P(F_1, T_1, T_2, T_3, T_4, T_5)$ as a product of smaller probabilities. Fill in the blanks below that achieve this.

$$P(F_1, T_1, T_2, \dots, T_5) = P(T_1) \cdot \textbf{(i)} \cdot \textbf{(ii)} \cdot \textbf{(iii)} P\left(\textbf{(iv)} | \textbf{(v)}\right)$$

(i) ○ 1 ○ $P(F_1|T_1)$ ○ 5 ○ $P(T_5)$

(ii) ○ $P(T_2)$ ○ 5 ○ $P(F_1|T_1)$ ○ $P(T_2|T_1, F_1)$

(iii) ○ $\sum_{i=1}^{5}$ ○ $\sum_{i=3}^{5}$ ○ $\prod_{i=1}^{5}$ ○ $\prod_{i=3}^{5}$

(iv) ○ T_i ○ F_i ○ T_{i-1} ○ T_1

(v) ○ T_{i-1} ○ T_i ○ F_i ○ F_{i-1}