CS 188
Fall 2011

Introduction to Artificial Intelligence

Final Exam

INSTRUCTIONS

- You have 3 hours.
- The exam is closed book, closed notes except two pages of crib sheets.
- Please use non-programmable calculators only.
- Mark your answers ON THE EXAM ITSELF. If you are not sure of your answer you may wish to provide a *brief* explanation. All short answer sections can be successfully answered in a few sentences at most.

Last Name	
First Name	
SID	
Login	
All the work on this exam is my own. **(please sign)**	

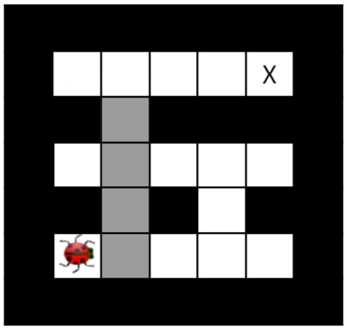
For staff use only

Q. 1	Q. 2	Q. 3	Q. 4	Q. 5	Q. 6	Q. 7	Q. 8	Total
/24	/12	/12	/8	/8	/14	/10	/12	/100

THIS PAGE INTENTIONALLY LEFT BLANK

1. **(24 points) Question Menagerie**

Parts (a) and (b) The hive of insects needs your help again. As before, you control an insect in a rectangular maze-like environment with dimensions $M \times N$, as shown to the right. At each time step, the insect can move into a free adjacent square or stay in its current location. All actions have cost 1.



In this particular case, the insect must pass through a series of partially flooded tunnels. Flooded squares are lightly shaded in the example map shown. The insect can hold its breath for A time steps in a row. Moving into a flooded square requires your insect to expend 1 unit of air, while moving into a free square refills its air supply.

(a) **(2 pt)** Give a minimal state space for this problem (i.e. do not include extra information). You should answer for a general instance of the problem, not the specific map shown.

(b) **(1 pt)** Give the size of your state space.

Parts (c), (d), and (e) Consider a search problem where all edges have cost 1 and the optimal solution has cost C. Let h be a heuristic which is $\max\{h^* - k, 0\}$, where h^* is the actual cost to the closest goal and k is a nonnegative constant.

(c) **(2 pt)** Circle all of the following that are true (if any).

(i) h is admissible.

(ii) h is consistent.

(iii) A* tree search (no closed list) with h will be optimal.

(iv) A* graph search (with closed list) with h will be optimal.

(d) **(1 pt)** Which of the following is the most reasonable description of how much more work will be done (= how many more nodes will be expanded) with heuristic h compared to h^*, as a function of k?

(i) Constant in k

(ii) Linear in k

(iii) Exponential in k

(iv) Unbounded

Now consider the same search problem, but with a heuristic h' which is 0 at all states that lie along an optimal path to a goal and h^* elsewhere.

(e) **(2 pt)** Circle all of the following that are true (if any).

(i) h' is admissible.

(ii) h' is consistent.

(iii) A* tree search (no closed list) with h' will be optimal.

(iv) A* graph search (with closed list) with h' will be optimal.

(f) **(2 pt)** You are running minimax in a large game tree. However, you know that the minimax value is between $x - \epsilon$ and $x + \epsilon$, where x is a real number and ϵ is a small real number. You know nothing about the minimax values at the other nodes. Describe briefly but precisely how to modify the alpha-beta pruning algorithm to take advantage of this information.

(g) **(2 pt)** An agent prefers chocolate ice cream over strawberry and prefers strawberry over vanilla. Moreover, the agent is indifferent between deterministically getting strawberry and a lottery with a 90% chance of chocolate and a 10% chance of vanilla. Which of the following utilities can capture the agent's preferences?

(i) Chocolate 2, Strawberry 1, Vanilla 0

(ii) Chocolate 10, Strawberry 9, Vanilla 0

(iii) Chocolate 21, Strawberry 19, Vanilla 1

(iv) No utility function can capture these preferences.

(h) (2 pt) Fill in the joint probability table so that the binary variables X and Y are independent.

X	Y	$P(X,Y)$
$+$	$+$	3/5
$+$	$-$	1/5
$-$	$+$	
$-$	$-$	

(i) (1 pt) Suppose you are sampling from a conditional distribution $P(B|A = +a)$ in a Bayes' net. Give the fraction of samples that will be *accepted* in rejection sampling.

(j) (2 pt) One could simplify the particle filtering algorithm by getting rid of the resampling phase and instead keeping weighted particles at all times, with the weight of a particle being the product of all observation probabilities $P(e_i|X_i)$ up to and including the current timestep. Circle all of the following that are true (if any).

(i) This will always work as well as standard particle filtering.

(ii) This will generally work less well than standard particle filtering because all the particles will cluster in the most likely part of the state space.

(iii) This will generally work less well than standard particle filtering because most particles will end up in low-likelihood parts of the state space.

(iv) This will generally work less well than standard particle filtering because the number of particles you have will decrease over time.

(k) (2 pt) Circle all settings in which particle filtering is preferable to exact HMM inference (if any).

(i) The state space is very small.

(ii) The state space is very large.

(iii) Speed is more important than accuracy.

(iv) Accuracy is more important than speed.

(l) (2 pt) Circle all of the following statements that are true (if any) about the perceptron and MIRA (with capacity $C = \infty$).

(i) Immediately after updating on a missed example, perceptron will classify that example correctly.

(ii) Immediately after updating on a missed example, MIRA will classify that example correctly.

(iii) On a missed example, from the same starting weight vector, the perceptron might make an update with a larger step size than MIRA.

(iv) On a missed example, from the same starting weight vector, MIRA might make an update with a larger step size than the perceptron.

(v) Immediately after updating on a missed example, perceptron will have a lower training error rate.

(vi) Immediately after updating on a missed example, MIRA will have a lower training error rate.

Parts (m) and (n) You have classification data with classes $Y \in \{+1, -1\}$ and features $F_i \in \{+1, -1\}$ for $i \in \{1, \ldots, K\}$. In an attempt to turbocharge your classifier, you duplicate each feature, so now each example has $2K$ features, with $F_{K+i} = F_i$ for $i \in \{1, \ldots, K\}$. The following questions compare the *original* feature set with the *doubled* one. You may assume that in the case of ties, class +1 is always chosen. Assume that there are equal numbers of training examples in each class.

(m) (2 pt) For a Naïve Bayes classifier, circle all of the following that apply.

(i) The test accuracy could be higher with the original features.

(ii) The test accuracy could be higher with the doubled features.

(iii) The test accuracy will be the same with either feature set.

(iv) On a given training instance, the conditional probability $P(Y|F_1, \ldots)$ on a training instance could be more extreme (i.e. closer to 0 or 1) with the original features.

(v) On a given training instance, the conditional probability $P(Y|F_1, \ldots)$ on a training instance could be more extreme (i.e. closer to 0 or 1) with the doubled features.

(vi) On a given training instance, the conditional probability $P(Y|F_1, \ldots)$ on a training instance will be the same with either feature set.

(n) (1 pt) For a perceptron classifier, circle all of the following that apply.

(i) The test accuracy could be higher with the original features.

(ii) The test accuracy could be higher with the doubled features.

(iii) The test accuracy will be the same with either feature set.

2. (12 points) Consistency and Harmony

(a) (2 pt) Consider a simple two-variable CSP, with variables A and B, both having domains $\{1, 2, 3, 4\}$. There is only one constraint, which is that A and B sum to 6. In the table below, circle the the values that remain in the domains after enforcing arc consistency.

Variable	Values remaining in domain
A	{ 1 , 2 , 3 , 4 }
B	{ 1 , 2 , 3 , 4 }

Now consider a general CSP C, with variables X_i having domains D_i. Assume that all constraints are between pairs of variables. C is not necessarily arc consistent, so we enforce arc consistency in the standard way, i.e. by running AC-3 (the arc consistency algorithm from class). For each variable X_i, let D_i^{AC} be its resulting arc consistent domain.

(b) (2 pt) Circle all of the following statements that are *guaranteed* to be true about the domains D_i^{AC} compared to D_i.

(i) For all i, $D_i^{AC} \subseteq D_i$.

(ii) If C is initially arc consistent, then for all i, $D_i^{AC} = D_i$.

(iii) If C is not initially arc consistent, then for at least one i, $D_i^{AC} \neq D_i$.

(c) (2 pt) Let n be the number of solutions to C and let n^{AC} be the number of solutions remaining after arc consistency has been enforced. Circle all of the following statements that *could* be true, if any.

(i) $n^{AC} = 0$ but no D_i^{AC} are empty.

(ii) $n^{AC} \geq 1$ but some D_i^{AC} is empty.

(iii) $n^{AC} \geq 1$ but a backtracking solver could still backtrack.

(iv) $n^{AC} > 1$

(v) $n^{AC} < n$

(vi) $n^{AC} = n$

(vii) $n^{AC} > n$

Consider an alternate, more aggressive property that can hold for an arc $X \to Y$ of a CSP. We say that $X \to Y$ is *arc harmonic* if **for all** $x \in X$, **for all** $y \in Y$, (x, y) satisfies any constraint that exists between X and Y. Just as with arc consistency, we can make a single arc harmonic by deleting domain values. Also as with arc consistency, we can enforce arc harmony over an entire CSP by processing one non-harmonic arc at a time, making it harmonic.

(d) (2 pt) Consider again the simple two-variable CSP, with variables A and B, both having domains $\{1, 2, 3, 4\}$ with the constraint that A and B should sum to 6. Specify reduced (but non-empty) domains for which the CSP is arc harmonic by circling remaining values in the table below. Note that, unlike with arc consistency, there will in general be multiple ways to make arcs harmonic, but you only need to specify one possibility here.

Variable	Values remaining in domain
A	{ 1 , 2 , 3 , 4 }
B	{ 1 , 2 , 3 , 4 }

After we have enforced arc harmony in the general CSP C, let D_i^{AH} be the reduced domain of the variable X_i, and let n^{AH} be the number of possible solutions remaining for this reduced CSP.

(e) (2 pt) Circle all of the following statements that *could* be true, if any.

(i) $n^{AH} = 0$ but no D_i^{AH} are empty.

(ii) $n^{AH} \geq 1$ but some D_i^{AH} is empty.

(iii) $n^{AH} \geq 1$ but a backtracking solver could still backtrack in this reduced CSP.

(iv) $n^{AH} > 1$

(v) $n^{AH} < n$

(vi) $n^{AH} = n$

(vii) $n^{AH} > n$

Define S_i, the *solution domain* of a variable X_i, to be the set of values that X_i takes in some solution. That is, $x \in S_i$ if and only if for some solution to the CSP, $X_i = x$.

(f) (2 pt) Circle all of the following statements that are *guaranteed* to be true, if any.

(i) $S_i \subseteq D_i$

(ii) $D_i \subseteq S_i$

(iii) $S_i \subseteq D_i^{AC}$

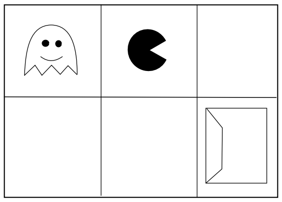
(iv) $D_i^{AC} \subseteq S_i$

(v) $S_i \subseteq D_i^{AH}$

(vi) $D_i^{AH} \subseteq S_i$

3. (12 points) Pac-Infiltration

A special operations Pacman must reach the exit (located on the bottom right square) in a ghost-filled maze, as shown to the right. Every time step, each agent (i.e., Pacman and each ghost) can move to any adjacent non-wall square or stay in place. Pacman moves first and the ghosts move collectively and simultaneously in response (so, consider the ghosts to be one agent). Pacman's goal is to exit as soon as possible. If Pacman and a ghost ever occupy the same square, either through Pacman or ghost motion, the game ends, and Pacman receives utility −1. If Pacman moves into the exit square, he receives utility +1 and the game ends. The ghosts are adversarial and seek to minimize Pacman's utility.



For parts (a) and (b), consider the 2x3 board with one ghost shown above. In this example, it is Pacman's move.

(a) (2 pt) What is the minimax value of this state?

(b) (2 pt) If we perform a depth-limited minimax search from this state, searching to depth $d = 1, 2, 3, 4$ and using zero as the evaluation function for non-terminal states, what minimax values will be computed? Reminder: each layer of depth includes one move from each agent.

d	Minimax value
1	
2	
3	
4	

For parts (c) and (d), consider a grid of size $N \times M$ containing K ghosts. Assume we perform a depth-limited minimax search, searching to depth d.

(c) (2 pt) How many states will be expanded (asymptotically) in terms of d using standard minimax? Your answer should use all the relevant variables needed and should not be specific to any particular map or configuration.

(d) (1 pt) For small boards, why is standard recursive minimax a poor choice for this problem? Your answer should be no more than one sentence!

Let the current state be $s \in S$, specifying the positions of Pacman and the ghosts. Let the actions available to Pacman be $a \in A$, and the (collective) action available to the ghosts be $b \in B$ (i.e., each single action b moves each ghost once). Let $S_A(a, s)$ denote the result of applying action a to state s. Similarly, let $S_B(b, s)$ be the result of applying action b. Let $M(s)$ be the minimax value of a state s where Pacman is next to move.

(e) (2 pt) Write a Bellman-style equation that defines the minimax value $M(s)$ of a non-terminal state s in terms of other state values $M(s')$. Your equation should be general, not specific to the single state above. Do not worry about special casing for terminal states.

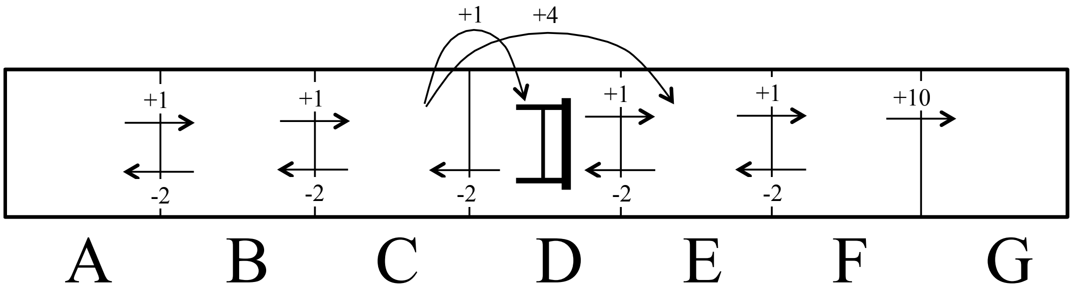
(f) (1 pt) If Pacman used your Bellman equation for an analogue of value iteration, what would the asymptotic time complexity of each round of value iteration be, in terms of the sizes of the sets defined above? (E.g., $|B||A|^2$).

The tables turn! Pacman is hiding while he waits for extraction. He cannot move, but the ghosts do not know where he is unless they collide with him. An optimal ghost mastermind is planning a search pattern for K ghosts that minimizes the worst-case time to find Pacman (i.e. guarantees that Pacman will be found in t steps for the smallest t possible).

(g) (2 pt) Specify a new state representation that the ghosts can use to plan their ghostly sweep. You should describe the elements of your state representation. As always, be compact in both state space size and answer size!

Consider an MDP modeling a hurdle race track, shown below. There is a single hurdle in square D. The terminal state is G. The agent can run *left* or *right*. If the agent is in square C, it cannot run *right*. Instead, it can *jump*, which either results in a fall to the hurdle square D, or a successful hurdle jump to square E. Rewards are shown below. Assume a discount of $\gamma = 1$.

Rewards:



Actions:

- *right*: Deterministically move to the right.
- *left*: Deterministically move to the left.
- *jump*: Stochastically jump to the right. This action is available for square C only.
  $T(\text{C}, \textit{jump}, \text{E}) = 0.5$ (jump succeeds)
  $T(\text{C}, \textit{jump}, \text{D}) = 0.5$ (jump fails)

Note: The agent cannot use *right* from square C.

(a) (2 pt) For the policy π of always moving forward (i.e., using actions *right* or *jump*), compute $V^\pi(\text{C})$.

(b) (3 pt) Perform two iterations of value iteration and compute the following. Iteration 0 corresponds to the initialization of all values to 0.

$V_2(\text{B})$	
$Q_2(\text{B}, \textit{right})$	
$Q_2(\text{B}, \textit{left})$	

(c) (3 pt) Fill in the blank cells of the table below with the Q-values that result from applying the Q-learning update for the 4 transitions specified by the episode below. You may leave Q-values that are unaffected by the current update blank. Use a learning rate α of 0.5. Assume all Q-values are initialized to 0.

Episode

s	a	r	s	a	r	s	a	r	s	a	r	s
C	*jump*	+4	E	*right*	+1	F	*left*	-2	E	*right*	+1	F

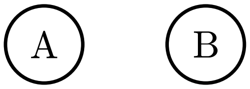
	Q(C, *left*)	Q(C, *jump*)	Q(E, *left*)	Q(E, *right*)	Q(F, *left*)	Q(F, *right*)
Initial	0	0	0	0	0	0
Transition 1						
Transition 2						
Transition 3						
Transition 4						

4. (8 points) A Game of d-Separation

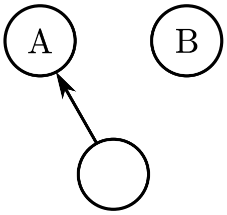
Consider the following two player game, which starts with a set of unconnected nodes that represent a Bayes net. Two nodes are labeled "A" and "B", and some are shaded. Play proceeds according to the following rules, with players taking alternating turns.

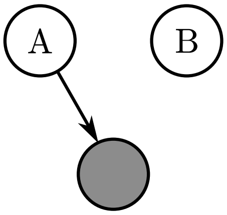
- On each turn, a player chooses any two nodes and draws a directed edge between them.
- No player may cause the graph to become cyclic or draw an edge that has already been drawn.
- Players attempt to keep A and B conditionally independent (i.e., d-separated) in the Bayes net given the shaded nodes.
- The first player to break the conditional independence between A and B loses.

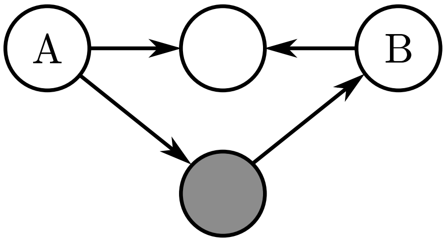
As an example, the game below starts with only two nodes, neither shaded. The first player has two options: draw an edge from A to B, or draw an edge from B to A. Either one breaks the independence of A and B, and so causes the first player to lose immediately.



(a) (3 pt) For each of the following three positions, determine if the next player has a move which does not lose immediately. If there is at least one such a move, draw one. If every move loses immediately, draw an "X" over the entire position.



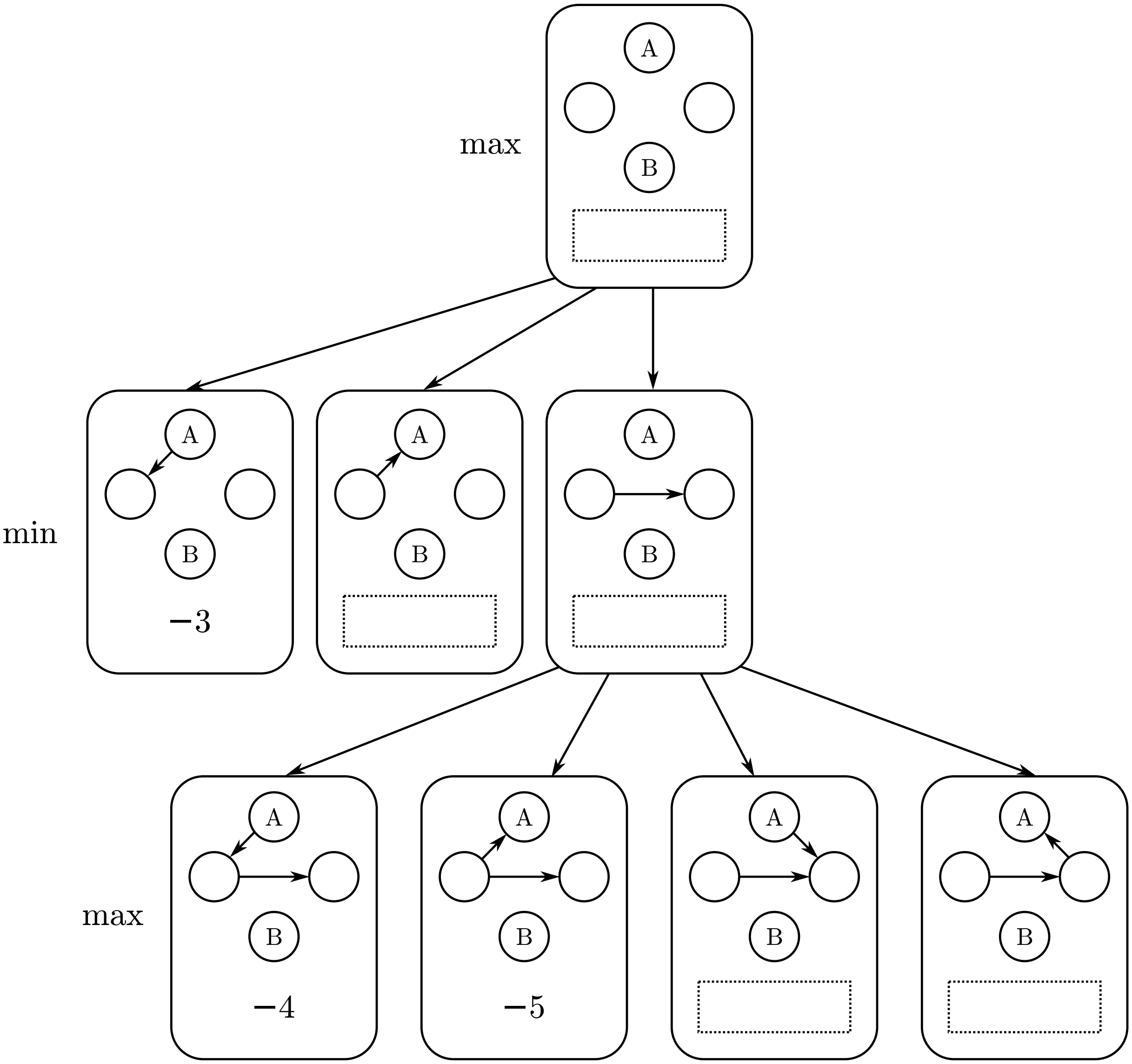




As an evaluation function, take the negative of the number of valid moves (i.e., no duplicates or cycles) that break d-separation. A partial game tree is shown below.

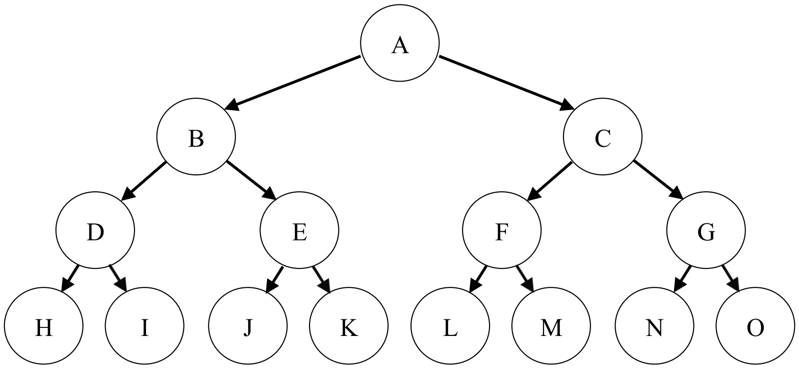
(b) **(3 pt)** Using minimax with this evaluation function, write the value of every node in the dotted boxes provided. Three values have been filled in for you.

(c) **(2 pt)** Draw an "X" over any nodes that alpha-beta search would prune when evaluating the root node, assuming it evaluates nodes from left to right.



5. (14 points) Occupy Cal

You are at Occupy Cal, and the leaders of the protest are deciding whether or not to march on California Hall. The decision is made centrally and communicated to the occupiers via the "human microphone"; that is, those who hear the information repeat it so that it propagates outward from the center. This scenario is modeled by the following Bayes net:



A	*P(A)*
+*m*	0.5
-*m*	0.5

π(X)	*X*	*P(X|π(X))*
+*m*	+*m*	0.9
+*m*	-*m*	0.1
-*m*	+*m*	0.1
-*m*	-*m*	0.9

Each random variable represents whether a given group of protestors hears instructions to march ($+m$) or not ($-m$). The decision is made at A, and both outcomes are equally likely. The protestors at each node relay what they hear to their two child nodes, but due to the noise, there is some chance that the information will be misheard. Each node except A takes the same value as its parent with probability 0.9, and the opposite value with probability 0.1, as in the conditional probability tables shown.

(a) (2 pt) Compute the probability that node A sent the order to march ($A = +m$) given that both B and C receive the order to march ($B = +m$, $C = +m$).

(b) (2 pt) Compute the probability that D receives the order $+m$ given that A sent the order $+m$.

You are at node D, and you know what orders have been heard at node D. Given your orders, you may either decide to march (*march*) or stay put (*stay*). (Note that these actions are distinct from the orders $+m$ or $-m$ that you hear and pass on. The variables in the Bayes net and their conditional distributions still behave exactly as above.) If you decide to take the action corresponding to the decision that was actually made at A (not necessarily corresponding to your orders!), you receive a reward of $+1$, but if you take the opposite action, you receive a reward of -1.

(c) (2 pt) Given that you have received the order $+m$, what is the expected utility of your optimal action? (Hint: your answer to part (b) may come in handy.)

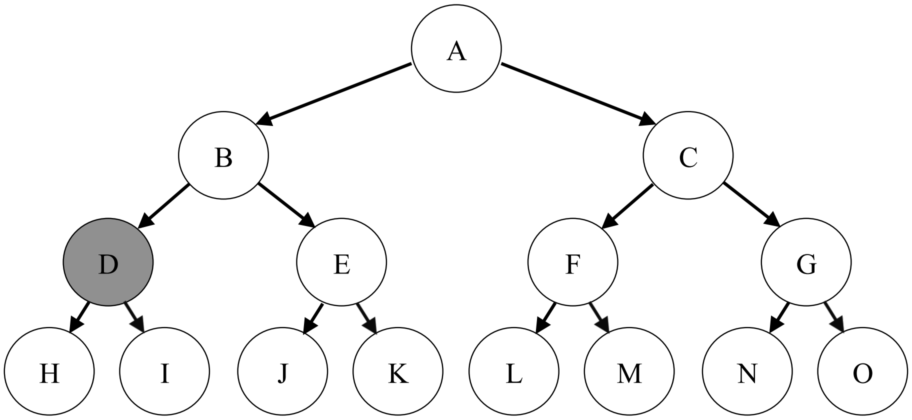
Now suppose that you can have your friends text you what orders they have received. (Hint: for the following two parts, you should not need to do much computation due to symmetry properties and intuition.)

(d) (2 pt) Compute the VPI of A given that $D = +m$.

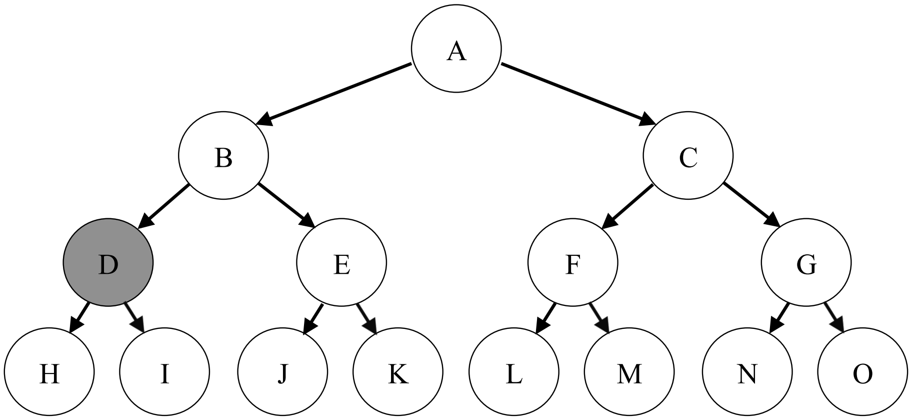
(e) (2 pt) Compute the VPI of F given that $D = +m$.

For the following parts, you should circle nodes in the accompanying diagrams that have the given properties. Use the quantities you have already computed and intuition to answer the following question parts; you should not need to do any computation.

(f) (2 pt) Circle the nodes for which knowing the value of that node changes your belief about the decision made at A given evidence at D (i.e. nodes X such that $P(A|X, D) \neq P(A|D)$).



(g) (2 pt) Circle the nodes which have nonzero VPI given evidence at D.



6. (10 points) HMMs and Particle Filtering

Consider a Markov Model with a binary state X (i.e., X_t is either 0 or 1). The transition probabilities are given as follows:

X_t	X_{t+1}	$P(X_{t+1} \mid X_t)$
0	0	0.9
0	1	0.1
1	0	0.5
1	1	0.5

(a) (2 pt) The prior belief distribution over the initial state X_0 is uniform, i.e., $P(X_0 = 0) = P(X_0 = 1) = 0.5$. After one timestep, what is the new belief distribution, $P(X_1)$?

X_1	$P(X_1)$
0	
1	

Now, we incorporate sensor readings. The sensor model is parameterized by a number $\beta \in [0, 1]$:

X_t	E_t	$P(E_t \mid X_t)$
0	0	β
0	1	$(1 - \beta)$
1	0	$(1 - \beta)$
1	1	β

(b) (2 pt) At $t = 1$, we get the first sensor reading, $E_1 = 0$. Use your answer from part (a) to compute $P(X_1 = 0 \mid E_1 = 0)$. Leave your answer in terms of β.

(c) (2 pt) For what range of values of β will a sensor reading $E_1 = 0$ increase our belief that $X_1 = 0$? That is, what is the range of β for which $P(X_1 = 0 \mid E_1 = 0) > P(X_1 = 0)$?

(d) (2 pt) Unfortunately, the sensor breaks after just one reading, and we receive no further sensor information. Compute $P(X_\infty \mid E_1 = 0)$, the stationary distribution *very many* timesteps from now.

X_∞	$P(X_\infty \mid E_1 = 0)$
0	
1	

(e) (2 pt) How would your answer to part (d) change if we never received the sensor reading E_1, i.e. what is $P(X_\infty)$ given no sensor information?

X_∞	$P(X_\infty)$
0	
1	

7. (12 points) ML

You are a Hollywood producer. You have a script in your hand and you want to make a movie. Before starting, however, you want to predict if the film you want to make will rake in huge profits, or utterly fail at the box office. You hire two critics A and B to read the script and rate it on a scale of 1 to 5 (assume only integer scores). Each critic reads it independently and announces their verdict. Of course, the critics might be biased and/or not perfect. For instance, for the past five movies you made, these are the critics' scores and the performance of the movie:

Movie Name	A	B	Profit?
Pellet Power	1	1	No
Ghosts!	3	2	Yes
Pac is Bac	4	5	No
Not a Pizza	3	4	Yes
Endless Maze	2	3	Yes

Table 1: Training Data

You decide to use machine learning to learn a classifier to predict if the script will lead to profits or not.

You first decide to use a perceptron to classify your data. This problem will use the multi-class formulation even though there are only two classes. Suppose you directly use the scores given above as features, together with a bias feature. That is $f_0 = 1$, $f_1 =$ score given by A and $f_2 =$ score given by B.

(a) (2 pt) You want to train the perceptron on the training data in Table 1. The initial weights are given below:

Profit	Weights
Yes	[-1, 0, 0]
No	[1, 0, 0]

(i) Which is the first training instance at which you update your weights?

(ii) In the table below, write the updated weights after the first update.

Profit	Weights
Yes	
No	

(b) (2 pt) More generally, irrespective of the training data, you want to know if your features are powerful enough to allow you to handle a range of scenarios. Some scenarios are given below. Circle those scenarios for which a perceptron using the features above can indeed perfectly classify the data.

(i) Your reviewers are awesome: if the total of their scores is more than 8, then the movie will definitely be a success and otherwise it will fail.

(ii) Your reviewers are art critics. Your movie will succeed if and only if each reviewer gives either a score of 2 or a score of 3.

(iii) Your reviewers have weird but different tastes. Your movie will succeed if and only if both reviewers agree.

You decide to use a different set of features. Consider the following feature space:

$$\begin{aligned}
f_0 &= 1 \text{ (The bias feature)}\\
f_{1A} &= 1 \text{ if score given by A is 1, 0 otherwise}\\
f_{1B} &= 1 \text{ if score given by B is 1, 0 otherwise}\\
f_{2A} &= 1 \text{ if score given by A is 2, 0 otherwise}\\
f_{2B} &= 1 \text{ if score given by B is 2, 0 otherwise}\\
&\ldots\\
f_{5B} &= 1 \text{ if score given by B is 5, 0 otherwise}
\end{aligned}$$

(c) (2 pt) Consider again the three scenarios in part (b). Using a perceptron with the new features, which of the three scenarios can be perfectly classified? Circle your answer(s) below:

(i) Your reviewers are awesome: if the total of their scores is more than 8, then the movie will definitely be a success, and otherwise it will fail.

(ii) Your reviewers are art critics. Your movie will succeed if and only if each reviewer gives either a score of 2 or a score of 3.

(iii) Your reviewers have weird but different tastes. Your movie will succeed if and only if both reviewers agree.

You have just heard of naive Bayes and you want to use a naive Bayes classifier. You use the scores given by the reviewers as the features of the naive Bayes classifier, i.e., the random variables in your naive Bayes model are A and B, each with a domain of $\{1, 2, \ldots, 5\}$, and *Profit* with a domain of *Yes* and *No*.

(d) (2 pt) Draw the Bayes' net corresponding to the naive Bayes model.

(e) (2 pt) List the types of the conditional probability tables you need to estimate along with their sizes (e.g., $P(X \mid Y)$ has 24 entries).

Probability	Size

(f) (2 pt) Your nephew is taking the CS188 class at Berkeley. He claims that the naive Bayes classifier you just built is actually a linear classifier in the feature space used for part (c). In other words, the decision boundary of the naive Bayes classifier is a hyperplane in this feature space. For the positive class, what is the weight of the feature f_{3B} in terms of the parameters of the naive Bayes model? You can answer in symbols, but be precise. (Hint: Consider the log of the probability.)