CS 188 Fall 2011 — Introduction to Artificial Intelligence — Midterm Exam

INSTRUCTIONS

- You have 3 hours.
- The exam is closed book, closed notes except a one-page crib sheet.
- 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)**	

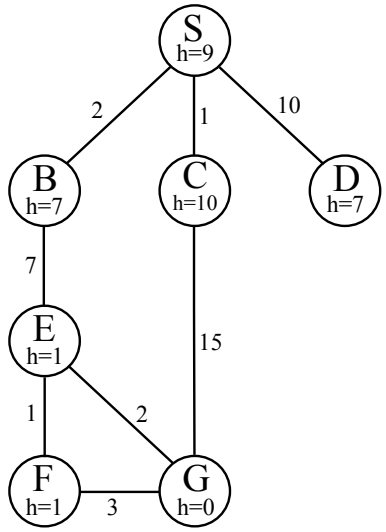
For staff use only

Q. 1	Q. 2	Q. 3	Q. 4	Q. 5	Q. 6	Q. 7	Q. 8	Total
/12	/12	/13	/7	/7	/11	/12	/6	/80

THIS PAGE INTENTIONALLY LEFT BLANK

1. (12 points) Search

Consider the search graph shown below. S is the start state and G is the goal state. All edges are bidirectional.



For each of the following search strategies, give the path that would be returned, or write *none* if no path will be returned. If there are any ties, assume alphabetical tiebreaking (i.e., nodes for states earlier in the alphabet are expanded first in the case of ties).

(a) (1 pt) Depth-first graph search
S-B-E-F-G

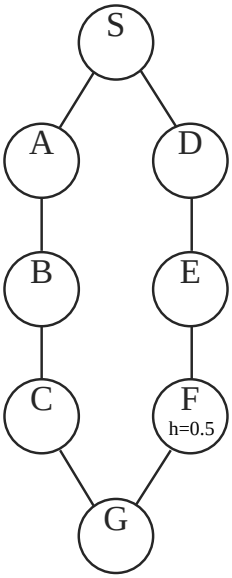
(b) (1 pt) Breadth-first graph search
S-C-G

(c) (1 pt) Uniform cost graph search
S-B-E-G

(d) (1 pt) Greedy graph search
S-B-E-G

(e) (2 pt) A* graph search
S-B-E-G

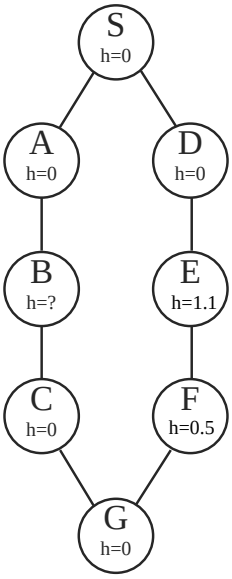
For the following question parts, all edges in the graphs discussed have cost 1.



(f) (3 pt) Suppose that you are designing a heuristic h for the graph on the right. You are told that $h(F) = 0.5$, but given no other information. What ranges of values are possible for $h(D)$ if the following conditions must hold? Your answer should be a range, e.g. $2 \leq h(D) < 10$. You may assume that h is nonnegative.

i. h must be admissible
$0 \leq h(D) \leq 3$
The path to goal from D is 3.

ii. h must be admissible and consistent
$0 \leq h(D) \leq 2.5$
In order for $h(E)$ to be consistent, it must hold that $h(E) - h(F) \leq 1$, since the path from E to F is of cost 1. Similarly, it must hold that $h(D) - h(F) = h(D) - 0.5 \leq 2$, or $h(D) \leq 2.5$.



(g) (3 pt) Now suppose that $h(\text{F}) = 0.5$, $h(\text{E}) = 1.1$, and all other heuristic values except $h(\text{B})$ are fixed to zero (as shown on the right). For each of the following parts, indicate the range of values for $h(\text{B})$ that yield an admissible heuristic AND result in the given expansion ordering when using A* graph search. If the given ordering is impossible with an admissible heuristic, write *none*. Break ties alphabetically. Again, you may assume that h is nonnegative.

i. B expanded before E expanded before F
$0.0 \leq h(\text{B}) \leq 1.1$

ii. E expanded before B expanded before F
$1.1 < h(\text{B}) \leq 1.5$

2. (12 points) Formulation: Holiday Shopping

You are programming a holiday shopping robot that will drive from store to store in order to buy all the gifts on your shopping list. You have a set of N gifts $G = \{g_1, g_2, \ldots g_N\}$ that must be purchased. There are M stores, $S = \{s_1, s_2, \ldots s_M\}$ each of which stocks a known inventory of items: we write $g_k \in s_i$ if store s_i stocks gift g_k. Shops may cover more than one gift on your list and will never be out of the items they stock. Your home is the store s_1, which stocks no items.

The actions you will consider are travel-and-buy actions in which the robot travels from its current location s_i to another store s_j in the fastest possible way and buys whatever items remaining on the shopping list that are sold at s_j. The time to travel-and-buy from s_i to s_j is $t(s_i, s_j)$. You may assume all travel-and-buy actions represent shortest paths, so there is no faster way to get between s_i and s_j via some other store. The robot begins at your home with no gifts purchased. You want it to buy all the items in as short a time as possible and return home.

For this planning problem, you use a state space where each state is a pair (s, u) where s is the current location and u is the set of unpurchased gifts on your list (so $g \in u$ indicates that gift g has not yet been purchased).

(a) (1 pt) How large is the state space in terms of the quantities defined above?

$M \times 2^N$. You are in one of M places (simple index from 1 to M), and have not purchased some subset of N items (binary vector of size N).

(b) (4 pt) For each of the following heuristics, which apply to states (s, u), circle whether it is admissible, consistent, neither, or both. Assume that the minimum of an empty set is zero.

([neither] / admissible / consistent / both) The shortest time from the current location to any other store: $\min_{s' \neq s} t(s, s')$

(neither / admissible / consistent / [both]) The time to get home from the current location: $t(s, s_1)$

(neither / admissible / consistent / [both]) The shortest time to get to any store selling any unpurchased gift: $\min_{g \in u}(\min_{s': g \in s'} t(s, s'))$

(neither / [admissible] / consistent / both) The shortest time to get home from any store selling any unpurchased gift: $\min_{g \in u}(\min_{s': g \in s'} t(s', s_1))$

([neither] / admissible / consistent / both) The total time to get each unpurchased gift individually: $\sum_{g \in u}(\min_{s': g \in s'} t(s, s'))$

([neither] / admissible / consistent / both) The number of unpurchased gifts times the shortest store-to-store time: $|u|(\min_{s_i, s_j \neq s_i} t(s_i, s_j))$

Remember, a consistent heuristic doesn't decrease from state to state by more than it actually costs to get from state to state. And of course, a heuristic is admissible if it is consistent. If you're confused, remember: the problem defines the minimum of an empty set as 0.

i. This heuristic does not return 0 in the goal state $(s_1, \emptyset)$, since it gives the minimum distance to any store *other than the current one*.

ii. We'll always need to get home from any state; the distance to home from home is 0; and this heuristic does not decrease by more than it costs to get from state to state.

iii. We'll always need to get that last unpurchased item, and taking the min distance store guarantees that we underestimate how much distance we actually have to travel. It is consistent because the heuristic never diminishes by more than what is travelled.

iv. We'll always need to get home from getting the last unpurchased item, and taking the min underestimates the actual requirement. What makes this heuristic inconsistent is that when we visit the last store to pick up the last unfinished item, the value of the heuristic goes to 0. Let's say the graph looks like this: s_3 `--1-->` s_2 `----5---->` s_1, with s_2 containing the last item. From s_3, the heuristic is 5, but from s_2, the heuristic is now 0, meaning that traveling from s_3 to s_2 decreases the heuristic by 5 but the actual cost is only 1.

v. This can overestimate the actual amount of work required.

vi. Same.

You have waited until very late to do your shopping, so you decide to send an swarm of R robot minions to shop in parallel. Each robot moves at the same speed, so the same store-to-store times apply. The problem is now to have all robots start at home, end at home, and for each item to have been bought by at least one robot (you don't have to worry about whether duplicates get bought). Hint: consider that robots may not all arrive at stores in sync.

(c) (4 pt) Give a minimal state space for this search problem (be formal and precise!)

We need the location of each robot at each time. At a given time, a robot can either be at one of M stores, or in any of $(T-1)M$ transition locations, where T is the maximum travel distance between two stores. Thus, the location of each robot takes $(MT)^R$. We also need the set of items purchased (2^N). Therefore, the size of each state is: $(MT)^R \times 2^N$.

One final task remains: you still must find your younger brother a stuffed Woozle, the hot new children's toy. Unfortunately, no store is guaranteed to stock one. Instead, each store s_i has an initial probability p_i of still having a Woozle available. Moreover, that probability drops exponentially as other buyers scoop them up, so after t time has passed, s_i's probability has dropped to $\beta^t p_i$. You cannot simply try a store repeatedly; once it is out of stock, that store will stay out of stock. Worse, you only have a single robot that can handle this kind of uncertainty! Phrase the problem as a single-agent MDP for planning a search policy for just this one gift (no shopping lists). You receive a single reward of +1 upon successfully buying a Woozle, at which point the MDP ends (don't worry about getting home); all other rewards are zeros. You may assume a discount of 1.

(d) (3 pt) Give a minimal state space for this MDP (be formal and precise!)

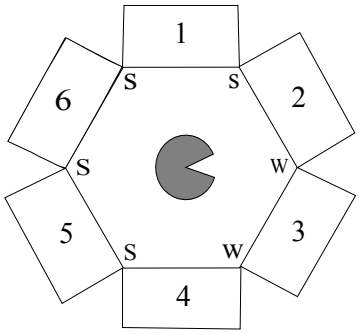
Which stores have been checked: 2^M
Whether Woozle has been bought: 2
Current time: T.
We may also want to keep track of the current location (M), but since there is no reward for traveling, we don't have to model that aspect of the problem.

3. (13 points) CSPs: Trapped Pacman

Pacman is trapped! He is surrounded by mysterious corridors, each of which leads to either a pit (P), a ghost (G), or an exit (E). In order to escape, he needs to figure out which corridors, if any, lead to an exit and freedom, rather than the certain doom of a pit or a ghost.

The one sign of what lies behind the corridors is the wind: a pit produces a strong breeze (S) and an exit produces a weak breeze (W), while a ghost doesn't produce any breeze at all. Unfortuantely, Pacman cannot measure the the strength of the breeze at a specific corridor. Instead, he can stand *between* two adjacent corridors and feel the max of the two breezes. For example, if he stands between a pit and an exit he will sense a strong (S) breeze, while if he stands between an exit and a ghost, he will sense a weak (W) breeze. The measurements for all intersections are shown in the figure below.

Also, while the total number of exits might be zero, one, or more, Pacman knows that two neighboring squares will *not* both be exits.



Pacman models this problem using variables X_i for each corridor i and domains P, G, and E.

(a) (3 pt) State the binary and/or unary constraints for this CSP (either implicitly or explicitly).

From the breezes, we get the following constraints:

Binary		Unary
$X_1 = P$ or $X_2 = P$,	$X_2 = E$ or $X_3 = E$,	$X_2 \neq P$
$X_3 = E$ or $X_4 = E$,	$X_4 = P$ or $X_5 = P$,	$X_3 \neq P$
$X_5 = P$ or $X_6 = P$,	$X_1 = P$ or $X_6 = P$,	$X_4 \neq P$

And there is another binary constraint: If adjacent(i, j), then $\neg(X_i = E \wedge X_j = E)$.

(b) (4 pt) Cross out the values from the domains of the variables that will be deleted in enforcing arc consistency.

X_1	P		
X_2		G	E
X_3		G	E
X_4		G	E
X_5	P		
X_6	P	G	E

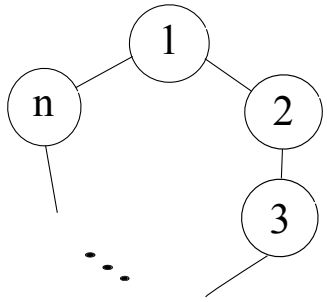
(c) **(1 pt)** According to MRV, which variable or variables could the solver assign first?

X_1 or X_5 (tie breaking)

(d) **(1 pt)** Assume that Pacman knows that $X_6 = G$. List all the solutions of this CSP or write *none* if no solutions exist.

(P,E,G,E,P,G)
(P,G,E,G,P,G)
Don't forget that exits cannot be adjacent to each other, and that it takes at least one exit to generate a weak breeze.

The CSP described above has a circular structure with 6 variables. Now consider a CSP forming a circular structure that has n variables ($n > 2$), as shown below. Also assume that the domain of each variable has cardinality d.



(e) **(2 pt)** Explain precisely how to solve this general class of circle-structured CSPs efficiently (i.e. in time linear in the number of variables), using methods covered in class. Your answer should be at most two sentences.

We fix X_j for some j and assign it a value from its domain (i.e. use cutset conditioning on one variable). The rest of the CSP now forms a tree structure, which can be efficiently solved without backtracking by one backward arc-enforcing and one forward value-setting pass. We try all possible values for our selected variable X_j until we find a solution.

(f) **(2 pt)** If standard backtracking search were run on a circle-structured graph, enforcing arc consistency at every step, what, if anything, can be said about the worst-case backtracking behavior (e.g. number of times the search could backtrack)?

A tree structured CSP can be solved without any backtracking. Thus, the above circle-structured CSP can be solved after backtracking at most d times, since we might have to try up to d values for X_j before finding a solution.

4. (7 points) Games: Multiple Choice

In the following problems please choose **all** the answers that apply. You may circle more than one answer. You may also circle no answers, if none apply.

(a) (2 pt) In the context of adversarial search, α-β pruning

(i) can reduce computation time by pruning portions of the game tree

(ii) is generally faster than minimax, but loses the guarantee of optimality

(iii) always returns the same value as minimax for the root of the tree

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

(v) always returns the same value as minimax for all nodes of the tree

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

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

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

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

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

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

(i) v_∞ may be greater than or equal to v_{10}.

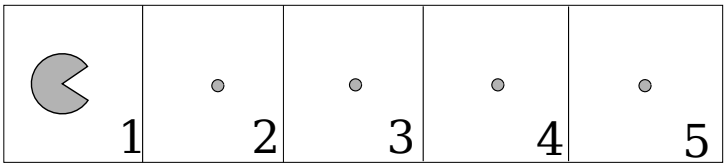
(ii) v_∞ may be less than or equal to v_{10}.

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

(iv) Against a perfect opponent, the actual outcome from following π_{10} may be less than v_∞.

This assumes that the perfect opponent is playing with infinite depth lookahead.

5. (7 points) MDPs: Bonus level!



Pacman is in a bonus level! With no ghosts around, he can eat as many dots as he wants. He is in the 5×1 grid shown. The cells are numbered from left to right as $1, \ldots, 5$. In cells 1 through 4, the actions available to him are to move *Right* (R) or to *Fly* (F) out of the bonus level. The action *Right* deterministically lands Pacman in the cell to the right (and he eats the dot there), while the *Fly* action deterministically lands him in a terminal state and ends the game. From cell 5, *Fly* is the only action. Eating a dot gives a reward of 10, while flying out gives a reward of 20. Pacman starts in the leftmost cell (cell 1).

We write this as an MDP where the state is the cell that Pacman is in. The discount is γ.

Consider the following 3 policies:

$$\pi_0(s) = F \text{ for all } s$$
$$\pi_1(s) = R \text{ if } s \leq 3, \ F \text{ otherwise}$$
$$\pi_2(s) = R \text{ if } s \leq 4, \ F \text{ otherwise}$$

(a) (4 pt) Assume $\gamma = 1.0$. What is:

i. $V^{\pi_0}(1)$?
20

ii. $V^{\pi_1}(1)$?
50

iii. $V^{\pi_2}(1)$?
60

iv. $V^*(1)$?
60

(b) (3 pt) Now consider an arbitrary value for γ.

i. Does there exist a value for γ such that π_0 is strictly better than both π_1 and π_2? If yes, give a value for γ. If no, write *none*.
Yes. $0 \leq \gamma < \frac{1}{2}$ How to get this answer. Assuming we start at state 1:

$$\begin{aligned} V^{\pi_0}(1) &> V^{\pi_2}(1) \\ 20 &> 10 + 10(\gamma + \gamma^2 + \gamma^3 + 2\gamma^4) \\ 1 &> \gamma + \gamma^2 + \gamma^3 + 2\gamma^4 \end{aligned}$$

It should be clear that this implies $\gamma < 0.5$, and of course gamma is always bounded by $[0, 1]$.

ii. Does there exist a value for γ such that π_1 is strictly better than both π_0 and π_2? If yes, give a value for γ. If no, write *none*.
None

iii. Does there exist a value for γ such that π_2 is strictly better than both π_0 and π_1? If yes, give a value for γ. If no, write *none*.
Yes. $\frac{1}{2} < \gamma \leq 1$. From before, we know that to beat π_0, we must have $\gamma > \frac{1}{2}$. Writing out $V^{\pi_1}(1) < V^{\pi_2}(1)$ will give the same inequality.

6. (11 points) MDPs and RL: Return to Blackjack

Armed with the power of Q-learning, you return to the CS188 casino! In this question, you will play a simplified version of blackjack where the deck is infinite and the dealer always has a fixed count of 15. The deck contains cards 2 through 10, J, Q, K, and A, each of which is equally likely to appear when a card is drawn. Each number card is worth the number of points shown on it, the cards J, Q, and K are worth 10 points, and A is worth 11. At each turn, you may either *hit* or *stay*. If you choose to hit, you receive no immediate reward and are dealt an additional card. If you stay, you receive a reward of 0 if your current point total is exactly 15, +10 if it is higher than 15 but not higher than 21, and −10 otherwise (i.e. lower than 15 or larger than 21). After taking the *stay* action, the game enters a terminal state *end* and ends. A total of 22 or higher is refered to as a *bust*; from a bust, you can only choose the action *stay*.

As your state space you take the set $\{0, 2, \ldots, 21, \text{bust}, \text{end}\}$ indicating point totals, "bust" if your point total exceeds 21, and "end" for the end of the game.

(a) (3 pt) Suppose you have performed k iterations of value iteration. Compute $V_{k+1}(12)$ given the partial table below for $V_k(s)$. Give your answer in terms of the discount γ as a variable. Note: do not worry about whether the listed V_k values could actually result from this MDP!

s	$V_k(s)$
13	2
14	10
15	10
16	10
17	10
18	10
19	10
20	10
21	10
bust	−10
end	0

$$V_{k+1}(12) = \frac{1}{13}(8 \cdot 10\gamma + 5 \cdot (-10)\gamma) = \frac{30}{13}\gamma$$

There are 8 cards (2 through 9) that will take us to a state with return 10, and 5 cards (10 through Ace) that will take us to the *bust* state.

You suspect that the cards do not actually appear with equal probability and decide to use Q-learning instead of value iteration.

(b) (4 pt) Given the partial table of initial Q-values below, fill in the partial table of Q-values on the right after the following episode occurred. Assume a learning rate of 0.5 and a discount factor of 1. The initial portion of the episode has been omitted. Leave blank any values which Q-learning does not update.

Initial values

s	a	$Q(s,a)$
19	hit	−2
19	stay	5
20	hit	−4
20	stay	7
21	hit	−6
21	stay	8
bust	stay	−8

Episode

s	a	r	s	a	r	s	a	r
19	hit	0	21	hit	0	bust	stay	−10

Updated values

s	a	$Q(s,a)$
19	hit	3
19	stay	
20	hit	
20	stay	
21	hit	-7
21	stay	
bust	stay	-9

How are the values updated? Here's a sample one: $Q(19, hit) = 0.5 * -2 + 0.5 * (0 + 1 * \max(-6, 8)) = 3$

Unhappy with your experience with basic Q-learning, you decide to featurize your Q-values, representing them in the form $\sum_i w_i f_i(s,a)$ for some feature functions $f_i(s,a)$.

First, consider the two feature functions

$$f_1(s,a) = \begin{cases} 0, \text{ if } a = \text{stay}; \\ +1, \text{ if } a = \text{hit and } s \geq 15; \\ -1, \text{ if } a = \text{hit and } s < 15; \end{cases} \quad \text{and} \quad f_2(s,a) = \begin{cases} 0, \text{ if } a = \text{stay}; \\ +1, \text{ if } a = \text{hit and } s \geq 18; \\ -1, \text{ if } a = \text{hit and } s < 18. \end{cases}$$

(c) (3 pt) Circle all of the following partial policy tables for which it is possible to represent Q-values in the form $w_1 f_1(s,a) + w_2 f_2(s,a)$ that imply that policy unambiguously (i.e., without having to break ties).

(i)

s	$\pi(s)$
14	hit
15	hit
16	hit
17	hit
18	hit
19	hit

(ii)

s	$\pi(s)$
14	stay
15	hit
16	hit
17	hit
18	stay
19	stay

(iii)

s	$\pi(s)$
14	hit
15	hit
16	hit
17	hit
18	stay
19	stay

(iv)

s	$\pi(s)$
14	hit
15	hit
16	hit
17	hit
18	hit
19	stay

(v)

s	$\pi(s)$
14	hit
15	hit
16	hit
17	stay
18	hit
19	stay

You find these features limiting, so you want a *single* set of features that can represent *any* arbitrary policy for this game via Q-values in the form $\sum_{i=1}^{n} w_i f_i(s,a)$. Remember that policies are defined over all states with choices: $\{0, 2, \ldots, 21\}$, not just the states listed in the previous part.

Write out the Q-values with respect to w_1 and w_2.

Q(s,a)	hit	stay
14	-w1-w2	0
15	w1-w2	0
16	w1-w2	0
17	w1-w2	0
18	w1+w2	0
19	w1+w2	0

Then, go through each policy and see if you have contradictions.

For example, for (i),

$$Q(14, hit) = -w1 - w2 > Q(14, stay) = 0 \Rightarrow -w1 - w2 > 0$$

.

However,

$$Q(18, hit) = w1 + w2 > Q(18, stay) = 0 \Rightarrow w1 + w2 > 0$$

.

Obviously, both cannot be true. So (i) fails. Try this for the other policies.

(d) (1 pt) What is the minimum number N of features needed to be able to encode all policies in this manner? Briefly justify your answer (no more than one sentence!).

The minimum number is 21. Every policy corresponds to a "quadrant" of the space of possible Q-value differences $\{(Q(s, \text{hit}) - Q(s, \text{stay}))_{s=0,2,\ldots,21}\}$. Since this is a 21-dimensional space, 21 vectors of the form $(f_i(s, \text{hit}) - f_i(s, \text{stay}))_{s=0,2,\ldots,21}$ are necessary to span the space and cover every quadrant.

7. (12 points) Mumps Outbreak There has been an outbreak of mumps at UC Berkeley. You feel fine, but you're worried that you might already be infected and therefore won't be healthy enough to take your CS 188 midterm. You decide to use Bayes nets to analyze the probability that you've contracted the mumps.

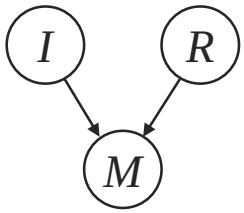
You first think about the following two factors:

- You think you have immunity from the mumps ($+i$) due to being vaccinated recently, but the vaccine is not completely effective, so you might not be immune ($-i$).
- Your roommate didn't feel well yesterday, and though you aren't sure yet, you suspect they might have the mumps ($+r$).

Denote these random variables by I and R. Let the random variable M take the value $+m$ if you have the mumps, and $-m$ if you do not. You write down the following Bayes net to describe your chances of being sick:

I	$P(I)$
$+i$	0.8
$-i$	0.2

R	$P(R)$
$+r$	0.4
$-r$	0.6



I	R	M	$P(M\|I,R)$
$+i$	$+r$	$+m$	0
$+i$	$+r$	$-m$	1.0
$+i$	$-r$	$+m$	0
$+i$	$-r$	$-m$	1.0
$-i$	$+r$	$+m$	0.7
$-i$	$+r$	$-m$	0.3
$-i$	$-r$	$+m$	0.2
$-i$	$-r$	$-m$	0.8

(a) (2 pt) Fill in the following table with the joint distribution over I, M, and R, $P(I, M, R)$.

I	R	M	$P(I, R, M)$
$+i$	$+r$	$+m$	0
$+i$	$+r$	$-m$	0.32
$+i$	$-r$	$+m$	0
$+i$	$-r$	$-m$	0.48
$-i$	$+r$	$+m$	0.056
$-i$	$+r$	$-m$	0.024
$-i$	$-r$	$+m$	0.024
$-i$	$-r$	$-m$	0.096

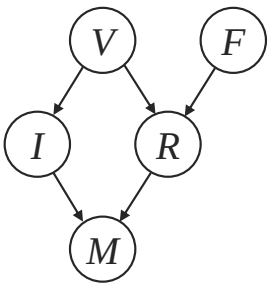
(b) (2 pt) What is the marginal probability $P(+m)$ that you have the mumps?

$$P(+m) = \sum_{i,r} P(i, r, +m) = P(+i, +r, +m) + P(+i, -r, +m) + P(-i, +r, +m) + P(-i, -r, +m)$$
$$= 0 + 0 + 0.056 + 0.024 = 0.08 = \frac{8}{100}$$

(c) (3 pt) Assuming you do have the mumps, you're concerned that your roommate may have the disease as well. What is the probability $P(+r \mid +m)$ that your roommate has the mumps given that you have the mumps? Note that you still don't know whether or not you have immunity.

$$P(+r \mid +m) = \frac{P(+r, +m)}{P(+m)} = \frac{\sum_i P(i, +r, +m)}{P(+m)} = \frac{0 + 0.056}{0.080} = 0.143 = \frac{7}{10}$$

You're still not sure if you have enough information about your chances of having the mumps, so you decide to include two new variables in the Bayes net. Your roommate went to a frat party over the weekend, and there's some chance another person at the party had the mumps $(+f)$. Furthermore, both you and your roommate were vaccinated at a clinic that reported a vaccine mix-up. Whether or not you got the right vaccine $(+v$ or $-v)$ has ramifications for both your immunity (I) and the probability that your roommate has since contracted the disease (R). Accounting for these, you draw the modified Bayes net shown on the right.



(d) (5 pt) Circle all of the following statements which are *guaranteed* to be true for this Bayes net:

(i) $V \perp\!\!\!\perp M \mid I,\ R$

(ii) $V \perp\!\!\!\perp M \mid R$

(iii) $M \perp\!\!\!\perp F \mid R$

(i), (iv), (vi). Use the Bayes ball algorithm.

(iv) $V \perp\!\!\!\perp F$

(v) $V \perp\!\!\!\perp F \mid M$

(vi) $V \perp\!\!\!\perp F \mid I$

8. (6 points) The Sound of (Silent) Music

You are an exobiologist, studying the wide range of life in the universe. You are also an avid dancer and have an excellent model of the way species invent dancing. The key variables are:

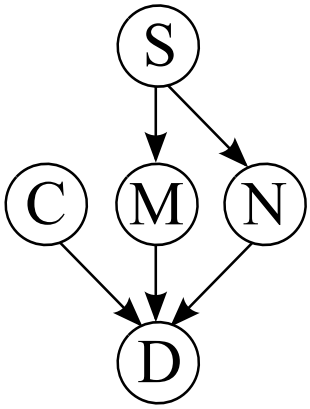
Sound sensing (S): Whether or not a species has the ability to sense sound

Cold climate (C): Whether or not the native planet of the species has cold weather

Music (M): Whether or not the species invented music

Non-verbal communication (N): Whether or not the species has any form of non-verbal communication

You model the relationships between these variables and **dancing (D)** using the Bayes net specified to the right.



You want to know how likely it is for a dancing, sound-sensing species to invent music, according to this Bayes net. However, you're a doctor, not a dynamic programmer, and you can't bear the thought of variable elimination this late in the exam. So, you decide to do inference via sampling.

You use prior sampling to draw the samples below:

S	C	M	N	D
$-s$	$+c$	$+m$	$-n$	$+d$
$+s$	$+c$	$-m$	$-n$	$-d$
$+s$	$-c$	$-m$	$+n$	$-d$
$+s$	$+c$	$+m$	$-n$	$+d$
$+s$	$+c$	$-m$	$+n$	$+d$
$+s$	$-c$	$-m$	$+n$	$-d$
$+s$	$-c$	$-m$	$-n$	$-d$
$+s$	$+c$	$+m$	$+n$	$+d$
$+s$	$+c$	$-m$	$+n$	$-d$
$-s$	$-c$	$-m$	$-n$	$-d$

(a) (2 pt) Based on rejection sampling using the samples above, what is the answer to your query, $P(-m \mid +d, +s)$?

1/3.
Simply find all the rows in the table above with $+d$ and $+s$, and count the number of times $-m$ occurs divided by the total number of such rows.

While your sampling method has worked fairly well in many cases, for rare cases (like species that can't sense sound) your results are less accurate as rejection sampling rejects almost all of the data. You decide to use likelihood weighting instead. The conditional probabilities of the Bayes net are listed below.

C	M	N	D	P(D | C , M , N)
$+c$	$+m$	$+n$	$+d$	0.9
$+c$	$+m$	$+n$	$-d$	0.1
$+c$	$+m$	$-n$	$+d$	0.8
$+c$	$+m$	$-n$	$-d$	0.2
$+c$	$-m$	$+n$	$+d$	0.8
$+c$	$-m$	$+n$	$-d$	0.2
$+c$	$-m$	$-n$	$+d$	0.2
$+c$	$-m$	$-n$	$-d$	0.8
$-c$	$+m$	$+n$	$+d$	0.8
$-c$	$+m$	$+n$	$-d$	0.2
$-c$	$+m$	$-n$	$+d$	0.5
$-c$	$+m$	$-n$	$-d$	0.5
$-c$	$-m$	$+n$	$+d$	0.6
$-c$	$-m$	$+n$	$-d$	0.4
$-c$	$-m$	$-n$	$+d$	0.1
$-c$	$-m$	$-n$	$-d$	0.9

S	M	P(M | S)
$+s$	$+m$	0.8
$+s$	$-m$	0.2
$-s$	$+m$	0.1
$-s$	$-m$	0.9

S	P(S)
$+s$	0.9
$-s$	0.1

S	N	P(N | S)
$+s$	$+n$	0.7
$+s$	$-n$	0.3
$-s$	$+n$	0.9
$-s$	$-n$	0.1

C	P(C)
$+c$	0.5
$-c$	0.5

You now wish to compute the probability that a species that has no sound-sensing ($-s$) or dancing ($-d$) nonetheless has music ($+m$), using likelihood weighting. I.e., you want $P(+m \mid -s, -d)$.

(b) (2 pt) You draw the samples below, using likelihood weighting. For each of these samples, indicate its weight.

S	C	M	N	D	weight
$-s$	$+c$	$+m$	$+n$	$-d$	0.01
$-s$	$+c$	$-m$	$-n$	$-d$	0.08
$-s$	$-c$	$-m$	$+n$	$-d$	0.04
$-s$	$+c$	$+m$	$+n$	$-d$	0.01

(c) (2 pt) Compute the answer to your query, $P(+m \mid -s, -d)$, using likelihood weighting with these samples.

1/7
Divide the sum of the weights corresponding to the $+m$ rows by the total sum of the weights.