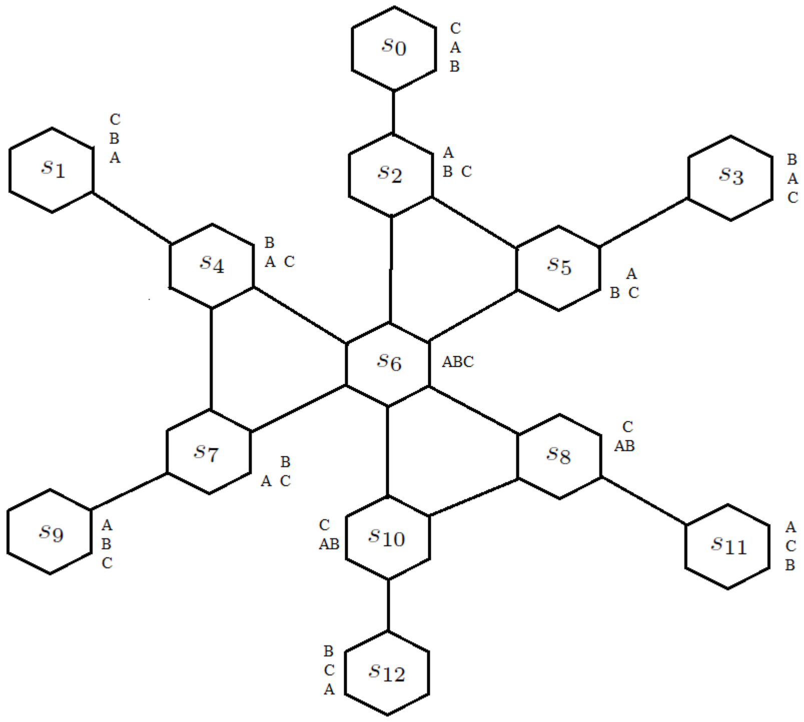
# CSE 415–Spring 2019 — Final Exam Solutions (Version of 2019-06-07 at 12:34 PM)

by the Staff of CSE 415, Spring 2019

## 1 A* Search (Kimberly)

Consider the state-space graph below. Each regular hexagon is a node corresponding to a state that represents an arrangement of 3 blocks: A, B, and C on a table. The initial state is $s_0$, which is an arrangement of the blocks in a "tower" formation, with block C on top of block A, and block A on top of block B. The goal state is $s_9$, where we again have a tower, but the blocks are in order, with A on top, B just under it, and C under B. The operators of the problem allow picking up one block at a time and moving it, provided no other block is on top of it. It can be moved either to the table (if it's not already touching the table) or to be on top of another block that doesn't already have a block on top of it. The edges are bidirectional, meaning that each operation has an inverse. The edge costs are all 1.



Let's define the following heuristic function: $\hat{h}(s)$ to be 2 times the number of occurrences in state $s$ of any of these patterns: B on A, or C on A, or C on B. For example $\hat{h}(s_1) = 4$, because B is on A, and C is on B, for a count of 2 pattern occurrences, and doubling that gives 4.

(a) (4 points) Draw the value of $\hat{h}(s)$ at each node, just above the hexagon.

| | | |
|---|---|---|
| $\hat{h}(s_0) = 2$ | $\hat{h}(s_4) = 2$ | $\hat{h}(s_8) = 2$ |
| $\hat{h}(s_1) = 4$ | $\hat{h}(s_5) = 0$ | $\hat{h}(s_9) = 0$ |
| $\hat{h}(s_2) = 0$ | $\hat{h}(s_6) = 0$ | $\hat{h}(s_{10}) = 2$ |
| $\hat{h}(s_3) = 2$ | $\hat{h}(s_7) = 0$ | $\hat{h}(s_{11}) = 2$ |
| - | - | $\hat{h}(s_{12}) = 2$ |

(b) (2 points) State whether or not $\hat{h}$ is admissible. Yes, it is admissible.

(c) (4 points) Justify your answer to (b).
It's admissible because $\hat{h}(s)$ is always less than or equal to the true distance $h(s)$. To see this, note that each of the three patterns is a kind of "inversion": a pair of blocks out of order. Each inversion will require at least two moves to fix (one to remove the upper block, and eventually, one to place the lower block on top of it). So the number of moves from $s$ to the goal must be greater than or equal to twice the number of these patterns present in the state. This makes it so that the heuristic is never an overestimate of the true minimum cost from $s$ to the goal. By the way, $\hat{h}(s) \leq 4$, since we can't have C on A, and C on B in the same state.

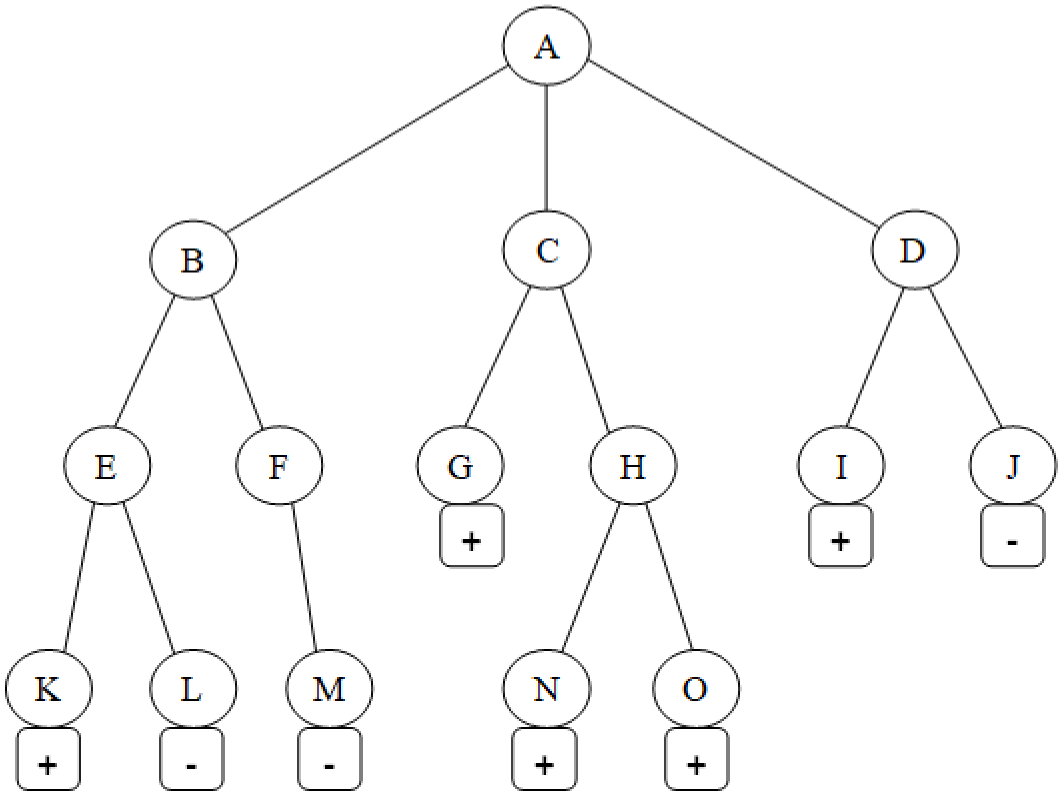
(d) (10 points) Simulate the A* algorithm as it searches for a lowest-cost path from $s_0$ to $s_9$.

In each iteration of the loop, show the priority queue (the "OPEN" list or "fringe") as a sorted list of (state, priority) pairs, as it is at the beginning of that iteration. For example at the beginning of the first iteration, the list is as shown on the first line below. When two states $s_i$ and $s_j$ have the same priority, break the tie in sorting by $i$ and $j$; i.e., put the lower-indexed state to the left of the higher indexed state; for example
($s_3$,5),($s_4$,5), and not ($s_4$, 5),($s_3$, 5).

[($s_0$, 2)]
[($s_2$, 1)]
[($s_5$, 2), ($s_6$, 2)]
[($s_6$, 2), ($s_3$, 5)]
[($s_7$, 3), ($s_3$, 5), ($s_4$, 5), ($s_8$, 5), ($s_{10}$, 5)]
[($s_9$, 4), ($s_3$, 5), ($s_4$, 5), ($s_8$, 5), ($s_{10}$, 5)]
[($s_3$, 5), ($s_4$, 5), ($s_8$, 5), ($s_{10}$, 5)]

## 2 Adversarial Search (Bryan)

Let's define the Games of Scones to be a Two-Agent, Zero-Sum, game whose possible moves and outcomes are shown in the tree diagram.



The players can't see the outcomes (plus or minus) until they have moved to a leaf node and a game ends. We will consider two agents X and Y that compete. Plus means a win for X. Minus means a win for Y. The initial state of the game is A, and X always goes first. X tries to maximize a score. Y tries to minimize a score. However, they each have their own methods of playing. Y always uses the same method to choose its turn: if the current state is $s$, then Y considers each child $s'$ of $s$, evaluates it statically using its own static evaluation function $f$, and it moves to the $s'$ having the minimum value. However, X plays using three different strategies. But within a given game, it will stick to one method. Here are the static evaluation functions used by X and Y:

| $s$ : | $A$ | $B$ | $C$ | $D$ | $E$ | $F$ | $G$ | $H$ | $I$ | $J$ | $K$ | $L$ | $M$ | $N$ | $O$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $e(s)$ : | 3 | 5 | 3 | 6 | 9 | 3 | 6 | 4 | 2 | 1 | 8 | 4 | 4 | 7 | 9 |
| $f(s)$ : | 1 | 7 | 5 | 4 | 6 | 3 | 3 | 2 | 2 | 1 | 7 | 3 | 4 | 8 | 6 |

We'll consider three games, with X using a different playing method in each one. You'll answer the same questions for each game: What is the sequence of moves in the game, and how many static evaluations are performed by X in that game, and which agent wins that game. Note that X uses the static evaluation function $e(s)$ and Y uses its own static evaluation function $f(s)$, and neither player knows that the other player has a different function.

(a) (5 points) In Game (a), X always moves from the current state $s$ by statically evaluating each child $s'$ of $s$, using its function $e$, and moves to the $s'$ having the maximum value.

What is the sequence of moves in this game? Hint: the first move is by X to state D, because $e(D) = 6$, which is greater than $e(B)$ or $e(C)$. So the sequence starts with the move A-D.

A-D-J

How many states are evaluated statically by X?

3: B, C, and D

Who wins this game?

Y

(b) (7 points) In Game (b), X doesn't just look at each child $s'$ of the current state $s$, but considers what Y might do. (So X is using minimax search.) It makes the assumption that Y will consider the children of whatever state it finds itself in, and chooses the child with the minimum value, and if Y is at a leaf node, X uses its static value there. However, it thinks Y is using the $e$ evaluation function, whereas we know that Y has its own evaluation function $f$.

What is the sequence of moves in this game?

3: A-C-H-O

How many static evaluations are performed by X in this game?

8 (E, F, G, H, I, J, N, O

Who wins this game?

X

(c) (8 points) In Game (c), X thinks that Y is playing randomly. (But Y is actually playing as it always does.) X thinks that Y selects one of its possible moves with equal probability. And X still considers only $e$ values in its figuring, not knowing about $f$. X makes a move from current state $s$ by simulating Y at each child $s'$ of $s$, and choosing that $s'$ with the highest expected value.

What is the sequence of moves in this game?

A-B-F-M

How many states are evaluated statically by X in this game?

7 (E, F, G, H, I, J, M)

Who wins this game?

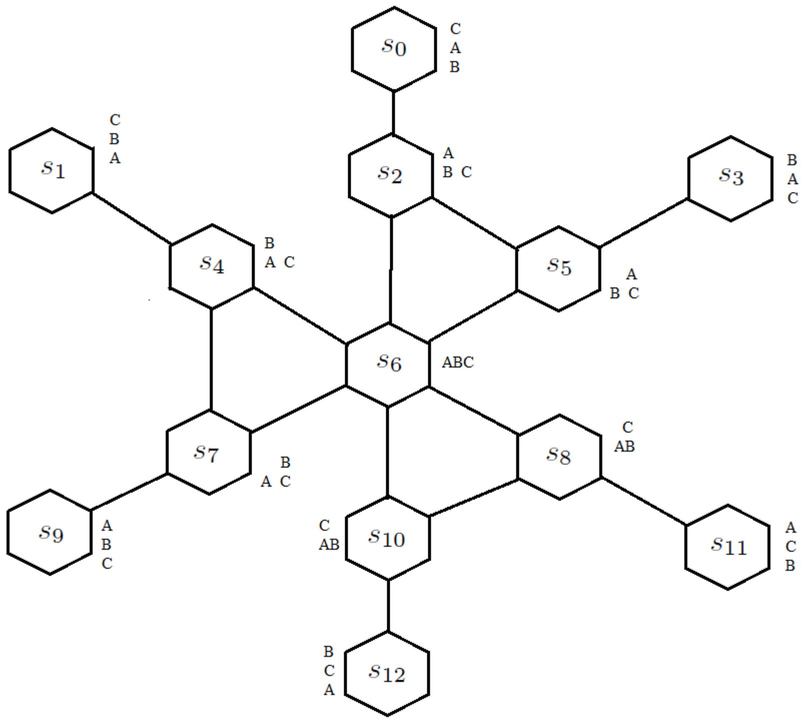
Y

## 3 MDPs: Value Iteration (Bryan) [Status: need answers and clarity check]

Once again, consider the state space for the Blocks-World problem from Question 1. Let's assume each state of the problem is part of a Markov Decision Process, and that there is one addition state $s_{13}$, not shown, which serves as a terminal state, reached only from $s_{12}$ via a special EXIT action.

Assume that rewards are normally $-1$ (a cost of living), but the transition from $s_{12}$ to $s_{13}$ via the EXIT action gives a reward of $+100$.

Assume the other actions are as follows: N (go North in the graph), NE (North-East), SE (South-East), S (South), SW (South-West), NW (North-West). We assume that any action that doesn't make sense at a state is simply disallowed. For example, at state $s_0$, the only allowed action is S (south). If, in algorithm such as Value Iteration, we have to loop over "all" actions at a state $s$, we'll just skip the actions that are not allowed.



(a) (2 points)

Assuming that the MDP has no noise (actions are deterministic, and actions involving operators whose preconditions are not satisfied are simply not allowed), and that the discount factor $\gamma$ is 1, what is the maximum total reward starting at $s_0$ and ending in the terminal state? 96

(b) (2 points) What sequence of actions achieves that reward? South, South, South, South, EXIT

(c) (8 points) Assuming we start Value Iteration with $V_0(s) = 0$ for all $s$, and $Q_0(s, a) = 0$ for all $s$ and $a$, determine the following values

$Q_1(s_9, \text{EXIT}) =$ -1

$Q_1(s_0, \text{S}) =$ -1

$Q_1(s_2, \text{SE}) =$ -1

$V_1(s_9) =$ -1

$V_1(s_7) =$ -1

$Q_2(s_7, \text{N}) =$ -2

$Q_2(s_7, \text{SW}) =$ -2

$V_2(s_7) =$ -2

(d) (8 points) What would those same expressions evaluate to if the discount factor had been 0.5 ?

$Q_1(s_9, \text{EXIT}) =$ -1

$Q_1(s_0, \text{S}) =$ -1

$Q_1(s_2, \text{SE}) =$ -1

$V_1(s_9) =$ -1

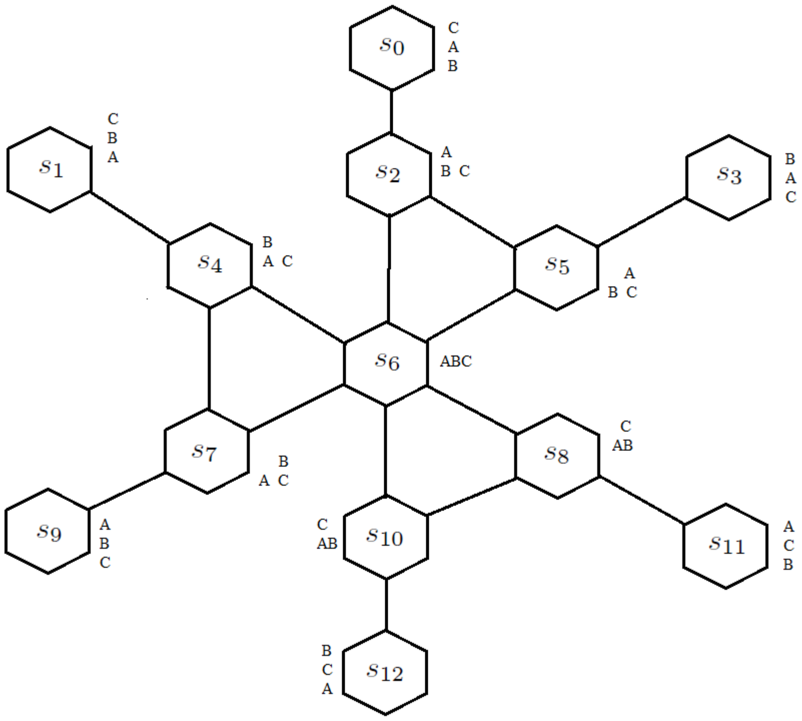
$V_1(s_7) =$ -1

$Q_2(s_7, \text{N}) =$ -1.5

$Q_2(s_7, \text{SW}) =$ -1.5

$V_2(s_7) =$ -1.5

## 4 MDPs: Q-Learning (Divye)



(a) (12 points) Consider the following episodes performed in this state space. The experience tuples are of the form [s, a, s', r], where the agent starts in state s, performs action a, ends up in state s', and receives immediate reward r, which is determined by the state entered. Let $\gamma = 1.0$ for this MDP. Fill in the values computed by the Q-learning algorithm with a learning rate of = 0.5. All Q values are initially 0, and you should fill out each row using values you have computed in previous rows.

| | |
|---|---|
| [$s_6$, S, $s_5$, -5] | Q($s_6$, S) = $.5 \times 0 + .5 * (-5 + 0) = -2.5$ |
| [$s_{10}$, S, $s_{12}$, 20] | Q($s_{10}$, S) = $.5 \times 0 + .5 * (20 + 0) = 10$ |
| [$s_6$, NE, $s_5$, -5] | Q($s_6$, NE) = $.5 \times 0 + .5 * (-5 + 0) = -2.5$ |
| [$s_8$, SW, $s_{10}$, 10] | Q($s_8$, SW) = $.5 \times 0 + .5 * (10 + 10) = 10$ |
| [$s_6$, NE, $s_{10}$, 10] | Q($s_6$, NE) = $.5 \times -2.5 + .5 * (10 + 10) = 8.75$ |
| [$s_6$, S, $s_{10}$, 10] | Q($s_6$, S) = $.5 \times -2.5 + .5 * (10 + 10) = 8.75$ |

(b) (8 points) Now, based on the record table in the previous problem, we want to approximate the transition and the reward functions:

$T(s_6, S, s_5) = 0.5$

$T(s_6, S, s_{10}) = 0.5$

$T(s_6, NE, s_5) = 0.5$

$T(s_6, NE, s_{10}) = 0.5$

$T(s_10, S, s_{12}) = 1.0$

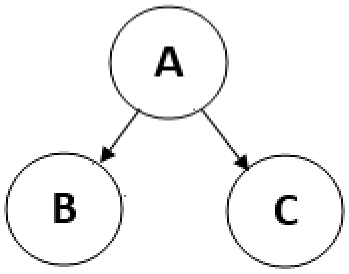
$R(*, *, s_{10}) = 10$

$R(*, *, s_5) = -5$

$R(*, *, s_{12}) = 20$

## 5 Joint Probability Distributions and Bayes Nets (Divye)

In this example, we consider a joint probability distribution of three random variables with different domains. We use a Bayes net to represent this distribution. The graph and conditional probability tables for the Bayes net are shown below.



| $A$ | $P(A)$ |
|---|---|
| $a_1$ | 0.1 |
| $a_2$ | 0.3 |
| $a_3$ | 0.6 |

| $B$ | $A$ | $P(B\|A)$ |
|---|---|---|
| $b_1$ | $a_1$ | 0.3 |
| $b_2$ | $a_1$ | 0.7 |
| $b_1$ | $a_2$ | 0.4 |
| $b_2$ | $a_2$ | 0.6 |
| $b_1$ | $a_3$ | 0.5 |
| $b_2$ | $a_3$ | 0.5 |

| $C$ | $A$ | $P(C\|A)$ |
|---|---|---|
| $c_1$ | $a_1$ | 0.2 |
| $c_2$ | $a_1$ | 0.8 |
| $c_1$ | $a_2$ | 0.55 |
| $c_2$ | $a_2$ | 0.45 |
| $c_1$ | $a_3$ | 0.9 |
| $c_2$ | $a_3$ | 0.1 |

In the following, wherever "compute" is requested, give a formula that shows what numbers and operations are required, but don't actually do the arithmetic. For example:

$$P(A = a_1, B = b_1 | C = c_1) = (0.1)(0.3)(0.2)/(0.2 + 0.55 + 0.9)$$

(a) (4 points) What are the numbers of free parameters in each of the three tables? (Here a free parameter can be thought of as a probability table entry that is not forced to a particular value by the preceding values in the table.)

__________

What is the total number of free parameters among the tables? ______

(b) (2 points) If the joint distribution for $A$, $B$, and $C$ were general and could not be factored as this Bayes net, how many free parameters would it have? ______

(c) (2 points) "Compute" $P(B = b_1)$. $0.3 \cdot 0.1 + 0.4 \cdot 0.3 + 0.5 \cdot 0.6$

(d) (2 points) "Compute" $P(C = c_1)$. $0.2 \cdot 0.1 + 0.55 \cdot 0.3 + 0.9 \cdot 0.6$

(e) (4 points) "Compute" $P(A = a_1 | B = b_1)$.

Use Bayes rule: $P(A = a_1 | B = b_1) = P(B = b_1 | A = a_1)P(A = a_1)/P(B = b_1)$.
$= (0.3)(0.1)/P(B = b_1)$. The denominator was computed in part (a).

(f) (6 points) "Compute" $P(C = c_1|B = b_1)$. For this example, you may wish to define one or more other probabilities (and "compute" them) to simplify your main expression.

Just as we found $P(A = a_1|B = b_1)$ in part (e), we now need, in addition, $P(A = a_2|B = b_1)$ and $(A = a_3|B = b_1)$. We use Bayes rule for each of these:
$P(A = a_2|B = b_1) = P(B = b_1|A = a_2)P(A = a_2)/P(B = b_1)$.
$P(A = a_3|B = b_1) = P(B = b_1|A = a_3)P(A = a_3)/P(B = b_1)$.
We can now compute $P(C = c_1|B = b_1) = P(C = c_1|A = a_1)P(A = a_1|B = b_1) + P(C = c_1|A = a_2)P(A = a_2|B = b_1) + P(C = c_1|A = a_3)P(A = a_3|B = b_1)$.

## 6 Perceptrons (Rob)

For the following questions, assume each perceptron outputs 1 if and only if its weighted input $h = x \bullet w$ satisfies $h > \theta$, and 0 otherwise. (This is not $h \geq \theta$.)

(a) (7 points) Write the weights and threshold for a two-input perceptron that would cause it to function as a NOR gate (that is, the negation of the logical OR). Possible input values are $\{0, 1\}$, where 0 represents False, and 1 represents True.

Ans: $w = \langle -1, -1 \rangle$, and $-1 \leq \theta < 0$.

(b) (9 points) Assuming an initial weight vector of $\langle 1, 1 \rangle$ and static $\theta = 0$, train this perceptron with the following example data. Use a learning rate of 1. Write the weight vector that would result after each training example. There is no bias term for this perceptron. Expected output for each example is listed after the data vector:

(i) $(\langle 0, 2 \rangle, 0)$ Ans: False positive, $\langle 1, -1 \rangle$

(ii) $(\langle 1, 1 \rangle, 1)$ Ans: False negative, $\langle 2, 0 \rangle$

(iii) $(\langle 0, 3 \rangle, 0)$ Ans: No change $\langle 2, 0 \rangle$

(c) (4 points) Assume we initially had a perceptron with weight vector $\langle 0, 0, 0 \rangle$ with $\theta = 1$. Then we gave a positive training example that was misclassified as negative and resulted in our weight vector changing to $\langle 3, 1, 2 \rangle$. Give a possible training data vector and learning rate that could cause this result.

Ans: $\langle 3, 1, 2 \rangle$, with learning rate 1.

## 7 NLP (Steve)

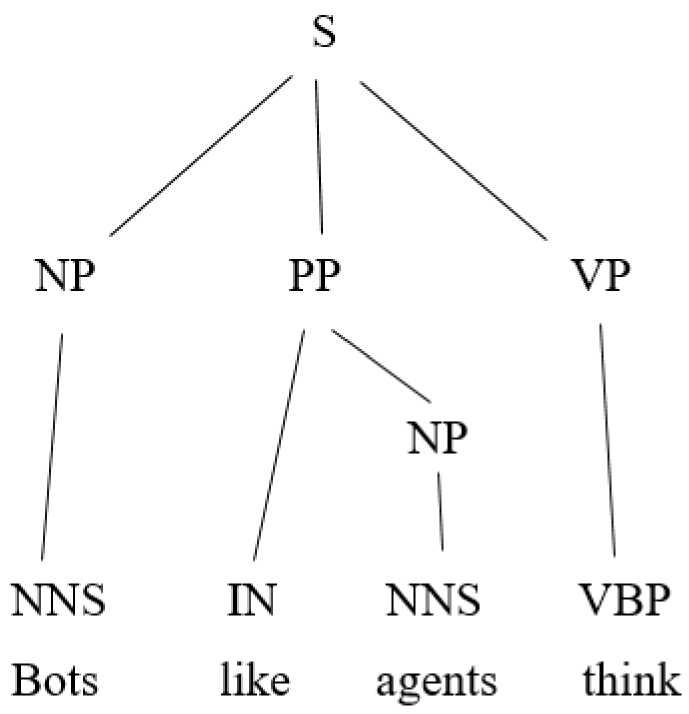
Consider the sentence “Bots like agents, think.” This might mean that robots that are similar to agents can think. It might also be interpreted to mean that robots are fond of agents, and robots also think. (There could also be other interpretations, but let’s not concern ourselves with them.)

Using the given probabilistic context-free grammar, find two legal parses, and compute a score for each one. Then identify the most probable parse using the scores. The grammar is given below. Consider the number at the right of a production to be the conditional probability of applying that production given that the symbol to be expanded is the symbol on the left-hand side of the production. Convert each probability into a score by taking score $= -\log_2(p)$.

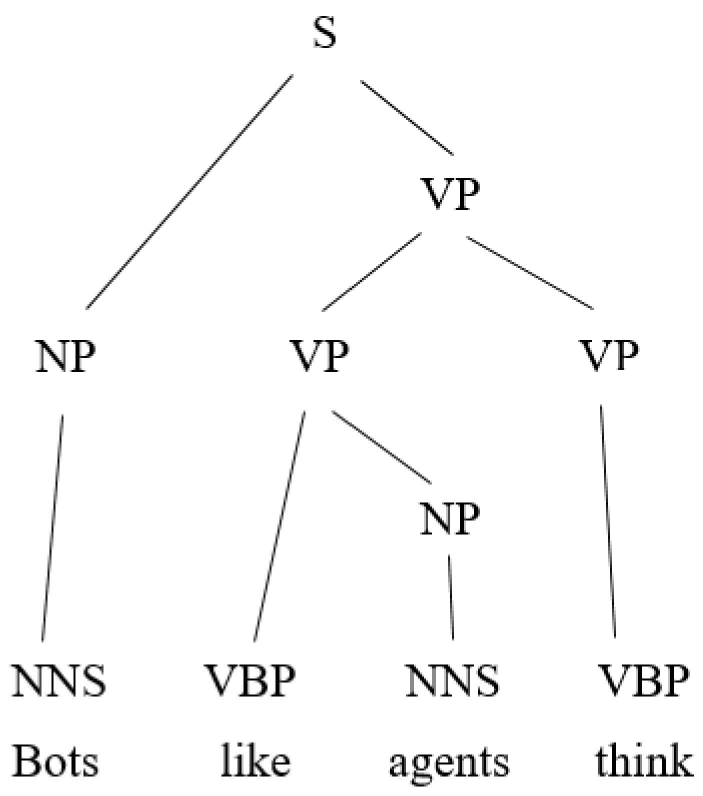
```
S   ::=  NP  VP        0.50   1
S   ::=  NP  PP  VP    0.25   2
NP  ::=  NNS           0.50   1
NP  ::=  NP  PP        0.25   2
VP  ::=  VBP           0.50   1
VP  ::=  VBP NP        0.25   2
VP  ::=  VP  VP        0.125  3
PP  ::=  IN  NP        1.00   0
```

(a) (4 point) Scores for each production rule: write the scores next to the probabilities above.

(b) (3 points) Parse number 1:



(c) (3 points) Parse number 2:



(d) (4 points) Total score ___5 and overall probability ___$2^{-5}$ for parse number 1.

(e) (4 points) Total score___10 and overall probability ___$2^{-10}$ for parse number 2.

(f) (2 point) Which parse is more probable? ______ ___1

## 8 Asimov's Laws (Steve)

It is 2029 and you work for the major commercial health iPhone-app provider in the US. You have just been told you are responsible for ensuring that all your company's apps behave "ethically," but no specifications have been provided explaining what that means. You vaguely remember learning about Asimov's Laws of Robotics and decide to use those in all your designs.

a. (6 points) What are the laws you decided to use (the original three are sufficient)?

First Law: a robot may not injure a human being or, through inaction, allow a human being to come to harm.

Second Law: a robot must obey the orders given it by human beings except where such orders would conflict with the First Law.

Third Law: a robot must protect its own existence as long as such protection does not conflict with the First or Second Laws.

One of the apps your company provides is a cooking-advisor chatterbot intended for home use. You send out an update so that all of them are now governed by the three laws and go home for the night in your company-provided, self-driving car, making a note to yourself update those the following morning). You spend the rest of the week sending out updates for various home-advice apps produced by your company.

The following week, you notice that many of the managers of your company are looking stressed and you hear that customer satisfaction has taken a nose-dive. You ask one of your colleagues, who works in personal health tracking, what's going on. He replies that he just got stuck with a weird bug to work on. The cooking-advice chatbots, despite years of working perfectly, are now refusing to answer customer questions, such as refusing to answer questions about the best recipes for chicken-fried steak. However, only chatbots that are also connected to personal medical records are affected.

b. (4 points) What do you think might be going on? The chatbots have recognized that certain foods are detrimental to their owner's health, and thus would cause harm.

c. (5 points) Explain why you think the chatbots are (or are not) applying Asimov's Laws correctly (you only need to make a case for one conclusion). Although the chatbot apps are disobeying orders, they are not breaking Asimov's 2nd Law because they are acting in accordance with Asimov's 1st Law not to "injure" a human being. The law does not specify a time frame to use when defining "injury."

You try to collect more information without drawing any unwanted attention to yourself. From the conversations you overhear reveal that strange behaviors are popping up among other types of artificially intelligent apps as well. You start to suspect the three laws you updated all the company's products with are to blame. You feverishly start researching how to recall updates and realize it might not be as easy as sending them out was. Several frustrating hours later, you go to your car and find it won't start. You realize that even though it is late, the parking lot is full of cars, along with a number of angry-appearing co-workers. You remember that when autonomous vehicles were first being developed, there had been some discussion of how such cars should behave in situations where accidents were unavoidable. Might this be contributing to the current problem?

d. (5 points) How might Asimov's Laws provide a useful starting point for thinking about desirable intelligent iPhone app behavior? While Asimov's Laws might not, themselves, be useful in their current form (and not just because they seem to assume a lot more self-awareness and judgment than any robots have at present), the problems they highlight are useful to consider to determine what our values are and how we might need to design technology that behaves in accordance with these values.

*NOTE: These are sample responses. Free responses from students may differ and will need to be evaluated individually.*

## 9 Multiple Choice (Rob)

This question is worth 40 points (5 for each subquestion). For each subquestion, underline the letter of the best answer. For example: A.

(a) In tree search, which is not an advantage of iterative deepening search over straight depth-first search to the same maximum depth?

(A) ***It might take longer to find a goal node.

(B) In game-playing, it can provide an anytime algorithm for computing a good move within the available time.

(C) When searching for a goal, when it does find a goal, it can immediately identify a shortest path.

(D) Its memory requirements are no greater, and on average, less.

(b) Perceptron training can be viewed as a kind of search for a certain type of hyperplane. What is most likely to prevent finding a solution?

(A) A learning rate that is following a harmonic sequence $1, \frac{1}{2}, \frac{1}{3}, ...$

(B) ***A training set in which the positive and negative examples cannot be completely separated by any hyperplane.

(C) An iterative algorithm that has no limit on the number of iterations it might use during training.

(D) The existence of extra dimensions in the vector representation of the data items, such that these dimensions contain irrelevant features.

(c) A* search prioritizes nodes using values of $fs$, where $f(s) = g(s)+h(s)$. If $h(s)$ is always zero then the algorithm becomes Uniform Cost Search. If $g(s)$ is always zero instead, then the algorithm becomes

(A) Depth-First Search.

(B) Best-First Search. ***

(C) Breadth-First Search

(D) None; the algorithm will not function properly.

(d) Which of these is not part of Kurzweil's "singularity"?

(A) Robots will have been improving their own designs.

(B) ***Robots are expected to fully control humans.

(C) Artificial intelligence will help spread human values.

(D) Great strides will have been made in neuroscience.

(e) Which of these stand in the way of using Asimov's rules for self-driving cars?

- (A) ***Robots cannot always predict the consequences of their actions accurately.
- (B) The rules are written in English and therefore irrelevant to robots and computers that don't perform natural language understanding.
- (C) The rules need to be adopted by national governments before they can be applied.
- (D) A car cannot be a robot, and vice versa.

(f) In graph search with a constant branching factor $b \geq 2$, what is the main advantage of DFS over BFS?

- (A) ***High likelihood of using less memory to find a path to the goal than with Breadth-First Search?
- (B) Guarantee of identifying a shortest path upon arrival at a goal state.
- (C) By processing the OPEN list like a queue rather than a stack, the generation of successors naturally goes faster.
- (D) By searching in all directions equally, there is no chance that the search algorithm will fail to reach a goal if one is reachable.

(g) Expectimax search is particularly appropriate in adversarial search if

- (A) the opponent is rational but always trying to minimize the state score.
- (B) the opponent is rational and always trying to maximize the state score.
- (C) the opponent always makes a predictable move.
- (D) ***the opponent is unpredictable.

(h) A* search is guaranteed to find a goal node

- (A) if its heuristic is admissible.
- (B) ***if at least one goal node is reachable.
- (C) if its heuristic is consistent.
- (D) if at least one goal node exists.