CS 188 Fall 2018

Introduction to Artificial Intelligence

Midterm 1

- You have 120 minutes. The time will be projected at the front of the room. You may not leave during the last 10 minutes of the exam.
- Do NOT open exams until told to. Write your SIDs in the top right corner of every page.
- If you need to go to the bathroom, bring us your exam, phone, and SID. We will record the time.
- In the interest of fairness, we want everyone to have access to the same information. To that end, we will not be answering questions about the content. If a clarification is needed, it will be projected at the front of the room. **Make sure to periodically check the clarifications**.
- The exam is closed book, closed laptop, and closed notes except your one-page cheat sheet. You are allowed a non-programmable calculator for this exam. Turn off and put away all other electronics.
- Mark your answers ON THE EXAM ITSELF IN THE DESIGNATED ANSWER AREAS. We will not grade anything on scratch paper.
- The last sheet in your exam packet is a sheet of scratch paper. Please detach it from your exam.
- For multiple choice questions:
  - ☐ means mark ALL options that apply
  - ○ means mark ONE choice
  - When selecting an answer, please fill in the bubble or square COMPLETELY (● and ■)

First name	
Last name	
SID	
Student to the right (SID and Name)	
Student to the left (SID and Name)	

For staff use only:

Q1.	Searching for a Password	/14
Q2.	Pushing Boxes	/12
Q3.	Simple Sudoku	/12
Q4.	Expectimin	/14
Q5.	Utility of Sugar	/14
Q6.	Card Decision Processes	/14
Q7.	Square World	/14
Q8.	Exploring the World	/6
	Total	/100

SID: ________

Q1. [14 pts] Searching for a Password

You are trying to recover a password to an encrypted file, by using search. You know that the password is up to 10 letters long and contains only the letters A, B, C.

You formulate a search problem:

- The initial state is the empty string.
- The successor function is to append one letter (A, B, or C) to the string.
- The goal test is to verify a candidate password using the decryption software. There are 6 correct passwords: AAACCC, ABBCC, BABAB, BCABACB, CBAC, and CBACB.
- Assume that **all ties are broken alphabetically**. For example, if there is a tie between states "A", "B", and "C", expand "A" first, then "B", then "C".

(a) [3 pts] From the six correct passwords below, select the one that will be returned by **depth-first search**:

- [✓ AAACCC]
- ○ ABBCC
- ○ BABAB
- ○ BCABACB
- ○ CBAC
- ○ CBACB

Depth-first search will expand states in alphabetical order: A, AA, AAA, . . . , AAAAAAAAAA, AAAAAAAAAB, AAAAAAAAAC, AAAAAAAAB, AAAAAAAABA, AAAAAAAABB, AAAAAAAABC, . . . With alphabetical tie-breaking, depth-first search will return the correct password that sorts first alphabetically: in this case, AAACCC.

(b) [3 pts] From the six correct passwords below, select the one that will be returned by **breadth-first search**:

- ○ AAACCC
- ○ ABBCC
- ○ BABAB
- ○ BCABACB
- [✓ CBAC]
- ○ CBACB

Breadth-first search will return the shortest correct password, CBAC. All of the other correct passwords are longer, so tie-breaking does not affect the answer.

(c) [4 pts] You suspect that some letters are more likely to occur in the password than others. You model this by setting $cost(A) = 1$, $cost(B) = 2$, $cost(C) = 3$. From the six correct passwords below, select the one that will be returned by **uniform cost search** using these costs:

- ○ AAACCC
- ○ ABBCC
- [✓ BABAB]
- ○ BCABACB
- ○ CBAC
- ○ CBACB

BABAB has the lowest cost (8) among the six goal states listed, so it will be returned by uniform cost search. All of the other correct passwords have higher cost, so tie-breaking does not affect the answer.

(d) [4 pts] Now suppose that all letters have cost 1, and that there is **a single correct password**, chosen uniformly at random from the state space. Candidate passwords can be checked using the decryption software, but the correct password is unknown. Which of the following statements is correct? *(The phrase "on average" reflects the fact that any password up to 10 letters long could be the correct one, all with equal probability.)*

○ Given any heuristic, A^* search will, on average, expand fewer states than depth-first search.

○ There exists a heuristic, using which A^* search will, on average, expand fewer states than depth-first search.

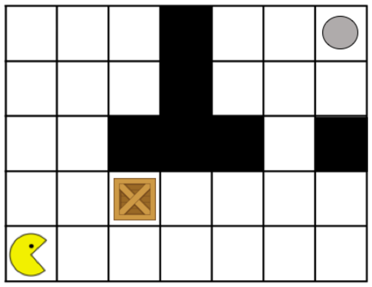
[✓ Given any heuristic, A^* search will, on average, expand the same number of states as depth-first search.]

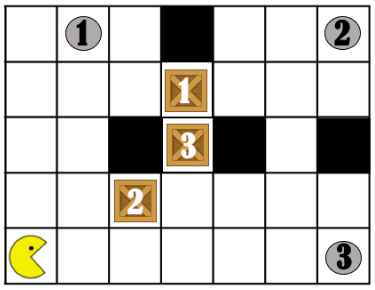
○ Given any heuristic, A^* search will, on average, expand more states than depth-first search.

○ There exists a heuristic, using which A^* search will, on average, expand more states than depth-first search.

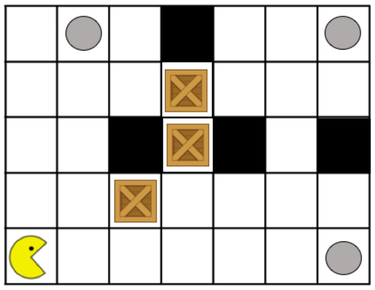
The correct password is unknown, so the heuristic we define can't be a function of the correct password. Moreover, only full passwords can be checked (i.e. there is no way of measuring partial progress towards the goal.) We have to keep checking candidate passwords until we find the correct one. Since any password could have been chosen, all with equal probability, exploring the state space in any order will (on average) be equally effective.

Q2. [12 pts] Pushing Boxes


(a) One box



(b) Numbered boxes and buttons


(c) Any box to any button

Pacman has to solve several levels of mazes by pushing boxes to circular buttons in the maze. Obviously, Pacman can only push a box (he does not have hands to pull it!). Pacman pushes a box by standing behind it and moving into its position. Pacman is not strong enough to push more than one box at a time. You can assume that the maze is $M \times N$ and that initially no box is upon any button. At each timestep, Pacman can just move either up, down, left, or right if he does not collide with any wall or the box that Pacman is pushing does not collide. Each action has a cost of 1. Actions that do not result in Pacman or a box being moved still have cost of 1. The figures display a possible configuration for each maze.

Note that for all parts of this question, d_{Man} is the Manhattan distance.

(a) In the first level, Pacman has to push a single box to a specific button (Figure 1a).

(i) [2 pts] What is the size of the minimal state space? Express your answer using the symbols M and N.

$\boxed{(\text{MN})^2}$ The mininmal state space corresponds to the position of the Pacman and the position of the box. Since each of them can be in MN positions the size of the state space is $(MN)^2$.

(ii) [2 pts] What is the branching factor? The answer should be a whole, positive number.

$\boxed{4}$ Pacman has 4 actions and the dynamics are deterministic. Therefore, the branching factor is 4.

(b) In the next level things get trickier for Pacman. Now, he has to push 3 boxes to 3 different buttons. Each box and button are numbered, and Pacman has to push the box to the button with the same number (Figure 1b).

(i) [2 pts] What is the size of the minimal state space? Express your answer using the symbols M and N.

$\boxed{(\text{MN})^4}$ The mininmal state space corresponds to the position of the Pacman and the position of the 3 boxes. Since each of them can be in MN positions the size of the state space is $(MN)^4$.

(ii) [2 pts] Which of the following heuristics are admissible?

☐ d_{Man}(Pacman, button 1) + d_{Man}(Pacman, button 2) + d_{Man}(Pacman, button 3) - 3

[✓] d_{Man}(box 1, button 1) + d_{Man}(box 2, button 2) + d_{Man}(box 3, button 3)

☐ d_{Man}(box 1, box 2) + d_{Man}(box 1, box 3)

[✓] min(d_{Man}(box 1, button 1), d_{Man}(box 2, button 2), d_{Man}(box 3, button 3))

☐ None of the above

The first one is not admissible since when, for instance, two of the boxes are placed and you are about to place the last one, the cost in that state is 1. However, the distance between Pacman and the buttons can be any number. The third one is not admissible neither because, for instance, the goal state does not have the value of 0.

(c) In the third maze, the 3 boxes can go to any of the 3 buttons (Figure 1c).

(i) [2 pts] What is the size of the minimal state space? Express your answer using the symbols M and N.

$MN\binom{MN}{3}$ The mininmal state space corresponds to the position of the Pacman and the position of the 3 boxes. Each of them can be in MN positions, and we do not care about knowing which box is where. Then size of the state space is $MN\frac{(MN)^3}{3!}$.

(ii) [2 pts] Which of the following heuristics are consistent?

☐ $\max_{ij} d_{\text{Man}}$(box i, button j)

■ $\min_{ij} d_{\text{Man}}$(box i, button j)

☐ $\max_{j} d_{\text{Man}}$(Pacman, button j)

■ $\min_{i} d_{\text{Man}}$(Pacman, box i) - 1

☐ None of the above

The first one is not consistent because is clearly not admissible. In a goal state the heuristic is not 0 but the maximum distance between any box and button. The third one is not consistent neither because is also not admissible; for instance, the heuristic in the goal state does not have a cost of 0.

SID: ________

Q3. [12 pts] Simple Sudoku

Pacman is playing a simplified version of a Sudoku puzzle. The board is a 4-by-4 square, and each box can have a number from 1 through 4. In each row and column, a number can only appear once. Furthermore, in each group of 2-by-2 boxes outlined with a solid border, each of the 4 numbers may only appear once as well. For example, in the boxes a, b, e, and f, each of the numbers 1 through 4 may only appear once. Note that the diagonals do not necessarily need to have each of the numbers 1 through 4.

In front of Pacman, he sees the board below. Notice that the board already has some boxes filled out! Box $b = 4$, $c = 2$, $g = 3$, $l = 2$, and $o = 1$.

a	b 4	c 2	d
e	f	g 3	h
i	j	k	l 2
m	n	o 1	p

Explicitly, we represent this simple Sudoku puzzle as a CSP which has the constraints:

1. Each box can only take on values 1, 2, 3, or 4.
2. 1, 2, 3, and 4 may only appear once in each row.
3. 1, 2, 3, and 4 may only appear once in each column.
4. 1, 2, 3, and 4 may only appear once in each set of 2-by-2 boxes with solid borders.
5. $b = 4$, $c = 2$, $g = 3$, $l = 2$, and $o = 1$.

(a) [4 pts] Pacman is very excited and decides to naively do backtracking using only forward checking. Assume that he solves the board from left to right, top to bottom (so he starts with box a and proceeds to box b, then c, etc), and assume that he has already enforced all unary constraints. Pacman assigns 3 to box a. If he runs forward checking, which boxes' domains should he attempt to prune?

[✓ d] [✓ e] [✓ f] ☐ h [✓ i] ☐ j ☐ k [✓ m] ☐ n ☐ p

In forward checking, we only prune the domains of variables that the assignment directly affects. Looking at the constraints, these are the ones that are in the same row/column, and the ones that are in its 2-by-2 grid.

(b) Pacman decides to start over and play a bit smarter. He now wishes to use arc-consistency to solve this Sudoku problem. Assume for all parts of the problem Pacman has already enforced all unary constraints, and Pacman has erased what was previously on the board.

(i) [4 pts] How many arcs are there in the queue prior to assigning any variables or enforcing arc consistency? Write your final, whole-number answer in the box below.

Answer: $\boxed{}$ (4+4)*4*3 + (4*4) = 112. There are 4 rows, 4 columns each with 4 permute 2 arcs. Additionally for each of the 4 2-by-2 boxes, there are the 4 diagonals.

(ii) [2 pts] Enforce the arc $d \to c$, from box d to box c. What are values remaining in the domain of box d?

- ■ 1
- ☐ 2
- ■ 3
- ■ 4

By the constraint that no two boxes in the same row can have the same number, after enforcing this arc we know that $d \neq 2$. We cannot say anything else just from enforcing this single arc.

(iii) [2 pts] After enforcing arc consistency, what is the domain of each box in the first row?

Block a: $\boxed{3}$ Block b: $\boxed{4}$

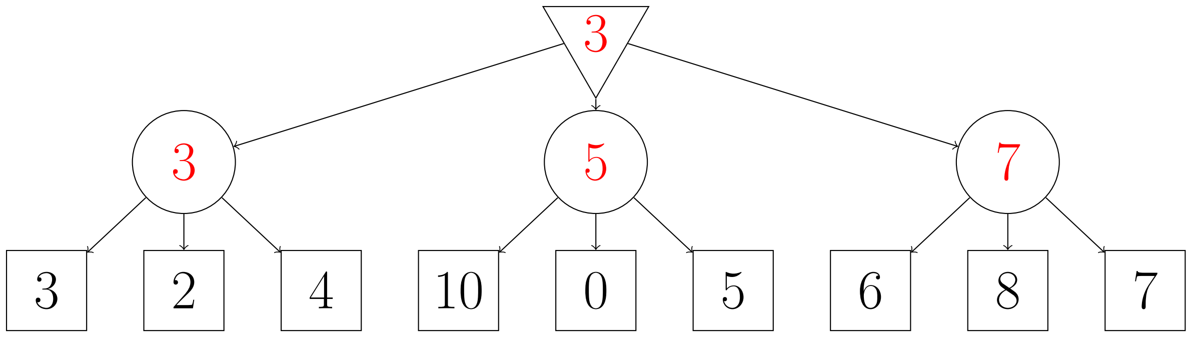
Block c: $\boxed{2}$ Block d: $\boxed{1}$

Because of unary constraints, we have domains of b and c containing 4 and 2 respectively. Enforcing the arcs in some order, (as long as we enforce $d \to c$, $d \to g$, and $d \to l$) we get that the domain of d contains only 1. Similarly, by constraining arcs $a \to b$, $a \to c$, and $a \to d$, we get that the domain of a can only be 3.

Q4. [14 pts] Expectimin

In this problem we model a game with a minimizing player and a random player. We call this combination "expectimin" to contrast it with expectimax with a maximizing and random player. Assume all children of expectation nodes have equal probability and sibling nodes are visited left to right for all parts of this question.

(a) [2 pts] Fill out the "expectimin" tree below.



The circles are expectation nodes and should be filled with the average of their children, and the triangle is the minimizer, which selects the minimum child value.

(b) [3 pts] Suppose that before solving the game we are given additional information that all values are non-negative and all nodes have exactly 3 children. Which leaf nodes in the tree above can be pruned with this information?

☐ 3 ☐ 2 ☐ 4 ☐ 10 ■ 0 ■ 5 ☐ 6 ☐ 8 ■ 7

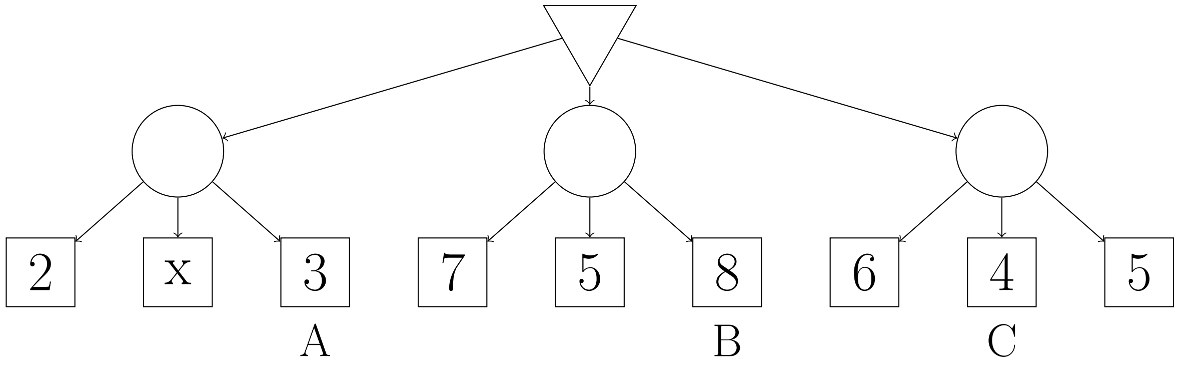
After the first expectation is determined to be 3, the minimizer is bounded to be ≤ 3. After the second expectation node sees 10 as its first child, it knows its value will $\geq 10/3$. Since these intervals do not overlap, we know the second expectation node will not be chosen by the minimizer and the rest of its children can be pruned. The third expectation node is bounded to $\geq 6/3 = 2$ after seeing 6, which still overlaps with ≤ 3, so we look at 8, after which the expectation is bounded to $\geq (6+8)/3 = 14/3$. Now the possible intervals are disjoint, so the last child can be pruned.

(c) [3 pts] In which of the following other games can we also use some form of pruning?

☐ Expectimax
☐ Expectimin
☐ Expectimax with all non-negative values and known number of children
■ Expectimax with all non-positive values and known number of children
☐ Expectimin with all non-positive values and known number of children

Expectimax and expectimin can't be pruned in general since any single child of an expectation node can arbitrarily change its value. Expectimax has maximizer nodes that will accumulate lower bounds, so the value ranges must help us give upper bounds on the expectations, which means the values must be bounded from above. Expectimin with non-positive values does not allow pruning since both the minimizer and expectation nodes will accumulate upper bounds.

(d) For each of the leaves labeled A, B, and C in the tree below, determine which values of x will cause the leaf to be pruned, given the information that all values are non-negative and all nodes have 3 children. Assume we do not prune on equality.



Below, select your answers as one of (1) an inequality of x with a constant, (2) "none" if no value of x will cause the pruning, or (3) "any" if the node will be pruned for all values of x. Fill in the bubble, then if you select one of the inequalities, fill in the blank with a number.

(i) [2 pts] A: ○ $x <$________ ○ $x >$________ [✓ None] ○ Any

No value of x can cause A to be pruned since the minimizer does not have any bound until after A is visited.

(ii) [2 pts] B: [✓ $x <$ 7] ○ $x >$________ ○ None ○ Any

To prune B, the value of the first expectation node must be less than the value of the second even if B were 0.

$$\frac{2+x+3}{3} < \frac{7+5}{3}$$
$$x < 7$$

(iii) [2 pts] C: [✓ $x <$ 1] ○ $x >$________ ○ None ○ Any

To prune C, the value of the first expectation node must have value less than the value of the third if both of the last two leaves had value 0.

$$\frac{2+x+3}{3} < \frac{6}{3}$$
$$x < 1$$

Q5. [14 pts] Utility of Sugar

(a) [4 pts] Every day after school Robert stops by the candy store to buy candy. He really likes Skittles, which cost \$4 a pack. His utility for a pack of Skittles is 30. KitKat bars cost \$1 each. His utility from KitKats is 6 for the first KitKat he buys, 4 for the second, 2 for the third, and 0 for any additional. He receives no utility if he doesn't buy anything at the store. The utility of m Skittle packs and n KitKats is equal to the following sum: utility of m Skittle packs PLUS utility of n KitKats.

In the table below, write the maximum total utility he can achieve by spending exactly each amount of money.

Robert:

\$0	\$1	\$2	\$3	\$4
0	6	10	12	30

With \$1, he buys a KitKat for 6 utility. With \$2 he buys another for 4 utility, giving 10 total, and with \$3, he buys 3 KitKats and gets $6 + 4 + 2 = 12$ utility. With \$4, he can buy a pack of Skittles for 30 utility and so does this instead of buying any KitKats.

For the remaining parts of this question, assume Sherry can achieve the following utilities with each amount of money when she goes to the candy store.

Sherry:

\$0	\$1	\$2	\$3	\$4
0	5	8	9	20

(b) Before Sherry goes to the store one afternoon, Juan offers her a game: they flip a coin and if it comes up heads he gives her a dollar; if it comes up tails she gives him a dollar. She has \$2, so she would end up with \$3 for heads or \$1 for tails.

(i) [2 pts] What is Sherry's expected utility at the candy store if she accepts his offer?

Answer: $\boxed{7}$

With probability .5 she ends up with \$1, which allows her to get utility 5, and with probability .5 she ends up with \$3 and 9 utility. The expected utility is $.5 * 5 + .5 * 9 = 7$.

(ii) [1 pt] Should she accept the game?

○ Yes [✓ No]

Since she can get utility 8 with the \$2 she has now, which is greater than the expectation 7 if she plays, she will not accept.

(iii) [1 pt] What do we call this type of behavior?

[✓ Risk averse]
○ Risk prone
○ Risk neutral

Risk averse behavior takes a certain outcome over an uncertain one with the same expected monetary value.

(c) The next day, Sherry again starts with \$2 and Juan offers her another coin flip game: if heads he gives her \$2 and if tails she gives him \$2.

(i) [2 pts] What is Sherry's expected utility at the candy store if she accepts his offer?

Answer: $\boxed{}$ $\boxed{10}$

The two equal probability outcomes in the lottery have utility $U(\$0) = 0$ and $U(\$4) = 20$, so the expected utility is $.5 * 0 + .5 * 20 = 10$

(ii) [1 pt] Should she accept the game?

[✓ Yes] No

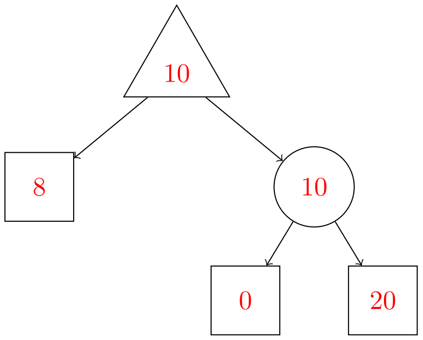
The expected utility for the game is greater than the utility of keeping \$2, so she accepts.

(iii) [1 pt] What do we call this type of behavior?

- Risk averse
- [✓ Risk prone]
- Risk neutral

Risk prone behavior takes a risk for a larger payoff over a certain one with the same expected monetary value.

(iv) [1 pt] For this scenario (from part **c**), fill in the expectimax tree below with the utility of each node (including value, chance, and maximization nodes).



The left value node is the certain outcome, which corresponds to refusing the game and getting 8 utility from the \$2 she has. The right subtree corresponds to the lottery between outcomes with utility 0 and 20, which has expected utility of 10. Sherry is maximizing her expected utility, so selects the greater option of 10. Note that the two children of the expectation node could be reversed.

(d) [1 pt] If someone is risk averse in one lottery but risk prone in another, does that mean they must be behaving irrationally?

Yes [✓ No]

As the situations in (b) and (c) showed, it is possible to be risk prone in one lottery and risk averse in another when the utility function is partially concave and partially convex.

Q6. [14 pts] Card Decision Processes

We have a card game, and there are three different cards: one has a value of 1, one a value of 2, and one a value of 3.

You have two actions in this game. You can either Draw or Stop. Draw will draw a card with face value 1, 2, or 3, each with probability $\frac{1}{3}$ (we assume we draw from a deck with replacement). You will bust if your hand's value goes above 5. This means that you immediately enter the terminal state "Done" and get 0 reward upon that transition.

Stop will immediately transition to the "Done" state, and receive a reward, which is the value of the cards in your hand. That is, $R(s, \underline{\text{Stop}}, \text{"}Done\text{"})$ will be equal to s, or your hand value.

The state in this MDP will be the value of the cards in your hand, and therefore all possible states are "0", "1", "2", "3", "4", "5", and "Done", which is all possible hand values and also the terminal state. The starting state will always be "0", because you never have any cards in your hand initially. "Done" is a terminal state that you will transition to upon doing the action Stop, which was elaborated above.

Discount factor $\gamma = 1$.

(a) [6 pts] Fill out the following table with the optimal value functions for each state.

States	0	1	2	3	4	5
$V^*(s)$	$\frac{32}{9}$	$\frac{11}{3}$	4	3	4	5

States	0	1	2	3	4	5
Iter. 0	0	0	0	0	0	0
Iter. 1	0	1	2	3	4	5
Iter. 2	2	3	4	3	4	5
Iter. 3	$\frac{10}{3}$	$\frac{11}{3}$	4	3	4	5
Iter. 4	$\frac{32}{9}$	$\frac{11}{3}$	4	3	4	5
Iter. 5	$\frac{32}{9}$	$\frac{11}{3}$	4	3	4	5

For all states, we have that $Q(s, Stop) = R(s, Stop, Done) + \gamma V(Done) = R(s, Stop, Done) = s$ and that $Q(s, Draw) = \sum_{i=1}^{3} \frac{1}{3}(R(s, Draw, s') + \gamma V_j(s')) = \sum_{i=1}^{3} \frac{1}{3} V_j(s')$. We do the update according to these rules.

In iteration 1, since all values are initialized to 0, we execute "Stop" at all states. For iteration 2, we find that it is better to "Draw" at state 2, since since $\frac{3+4+5}{3} > 2$, but not at state 4 or 5, since $\frac{5}{3} < 4$ and $0 < 5$, respectively. For iteration 2, we find it is better to draw at state 1, and for iteration 3 we follow a similar logic at iteration 4 for state 0.

Since values do not change between iteration 4 and 5, we find we have converged to the optimal values.

(b) [2 pts] How many iterations did it take to converge to the optimal values? We will initialize all value functions at iteration 0 to have the values 0. Take the iteration where all values first have the optimal value function as the iteration where we converged to the optimal values.

○ 0 ○ 1 ○ 2 ○ 3 [✓ 4] ○ 5 ○ ∞

Following the solution in part (a), we find that we have first calculated the optimal values in iteration 4.

(c) [3 pts] What was the optimal policy that you found above? "D" stands for the action "Draw" and "S" stands for the action "Stop".

States	0	1	2	3	4	5
$\pi^*(s)$	🟥 D ☐ S	🟥 D ☐ S	🟥 D ☐ S	🟥 D 🟥 S	☐ D 🟥 S	☐ D 🟥 S

The optimal value functions are below again.

States	0	1	2	3	4	5
$V^*(s)$	$\frac{32}{9}$	$\frac{11}{3}$	4	3	4	5

Remember that policy is defined as $\pi(s) = argmax_a Q(s, a)$.

We can see it is optimal to "Draw" in state 0, since we took $V(0) = Q(0, Draw) = max(Q(0, Draw), Q(0, Stop))$, and $\frac{32}{9} > 0$. We can also similarly see it is optimal to "Draw" in state 1 and 2, since for those states, $V(s) = Q(s, Draw)$.

In state 3, we have that $Q(3, Draw) = Q(3, Stop)$, and so both are optimal actions for the policy. We, however, gave credit for either option being marked if we assume the policy is a function that must output a determined output per input.

In state 4 and 5, it is optimal to "Stop" in 4 and 5, since $V(s) = Q(s, Stop)$.

Q7. [14 pts] Square World

In this question we will consider the following gridworld:

0	**0**	**0**	**0**	**0**	**0**
+1	**A**	**B**	**C**	**D**	**+100**
0	**0**	**0**	**0**	**0**	**0**

Every grid square corresponds to a state. If a state is annotated with a number, it means that after entering this state only one action is available, the Exit action, and this will result in the reward indicated by the number and the episode will end. There is no reward otherwise. For the other 4 states, named A, B, C, D, two actions are available: Left and Right. Assume $\gamma = 1$ in this problem.

(a) [4 pts] Assume that the failure probability for the actions Left and Right is 0.5, and in case of failure the agents moves up or down, and the episode is terminated after the Exit action.

What are the optimal values?

States	A	B	C	D
$V^*(s)$	6.25	12.5	25	50

The optimal policy is to go to the right from state A, B, C, and D. Thus,

$V^*(D) = 0.5 * 0 + 0.5 * 100 = 50$

$V^*(C) = 0.5 * 0 + 0.5 * V(D) = 25$

$V^*(B) = 0.5 * 0 + 0.5 * V(C) = 12.5$

$V^*(A) = 0.5 * 0 + 0.5 * V(B) = 6.25$

(b) [4 pts] Still assume the failure probability in the previous part. Now assume further that there is an *integer* living reward r when the episode is not terminated. Is there a value of r that would make the optimal policy **only** decide Left at state D? If so, what's the minimum value of r?

Answer: $\boxed{51}$

Let X0 represent each of the grid that has number 0. We make no distinction between them because $V(X0)$ is the same for all $X0$. Let X100 and X1 represent the states that have number 100 and 1, respectively.

At convergence we have the optimal value function V^* such that

$V^*(D) = \max\{r + 0.5 * V(X100) + 0.5 * V(X0), r + 0.5 * V(C) + 0.5 * V(X0)\}$

$V^*(C) = \max\{r + 0.5 * V(B) + 0.5 * V(X0), r + 0.5 * V(D) + 0.5 * V(X0)\}$

$V^*(B) = \max\{r + 0.5 * V(A) + 0.5 * V(X0), r + 0.5 * V(C) + 0.5 * V(X0)\}$

$V^*(A) = \max\{r + 0.5 * V(X1) + 0.5 * V(X0), r + 0.5 * V(B) + 0.5 * V(X0)\}$

Because the optimal policy decides only Left at D, we have $V(C) > V(X100)$. We can show that $V^*(x) = 2r$ for all $x \in A, B, C, D$. Thus, we have $2r > 100$ and the smallest integer is 51.

Another interpretation is to give the living reward when taking the Exit action from state $X0$ and $X100$. In this case, $V^*(x) = 3r$ for all $x \in A, B, C, D$, and $3r > r + 100$. So, the answer is still 51.

(c) [4 pts] Assume we collected the following episodes of experiences in the form (state, action, next state, reward): (we use $X1$ and $X100$ to denote the leftmost and rightmost states in the middle row and Done to indicate the terminal state after an Exit action).

$$(B, Left, A, 0), (A, Left, X1, 0), (X1, Exit, Done, +1)$$
$$(B, Right, C, 0), (C, Right, D, 0), (D, Right, X100, 0), (X100, Exit, Done, +100)$$

If we run Q-learning initializing all Q-values equal to 0, and with appropriate stepsizing, replaying each of the above episodes infinitely often till convergence, what will be the resulting values for:

(State, Action)	(B, Left)	(B, Right)	(C, Left)	(C, Right)
$Q^*(s,a)$	1	100	0	100

At convergence, $Q^*(B, Left) = Q^*(A, Left) = Q^*(Exit, Done) = 1$ and $Q^*(B, Right) = Q^*(C, Right) = Q^*(D, Right) = Q^*(X100, Exit) = 100$. Other state-action pairs will have 0 value because we have not seen them.

(d) [2 pts] Now we are trying to do feature-based Q-learning. Answer the below True or False question.

There exists a set of features that are functions of state only such that approximate Q-learning will converge to the optimal Q-values.

True [✓ False]

False, because optimal Q values in this gridworld depend on actions.

Q8. [6 pts] Exploring the World

In this question, our CS188 agent is stuck in a maze. We use Q-learning with an epsilon greedy strategy to solve the task. There are 4 actions available: north (N), east (E), south (S), and west (W).

(a) [2 pts] What is the probability of each action if the agent is following an epsilon greedy strategy and the best action in state s under the current policy is N? Given that we are following an epsilon-greedy algorithm, we have a value ϵ. Use this value ϵ in your answer. $p(a_i|s)$ is the probability of taking action a_i in state s.

$p(N|s)$ $\boxed{(1 - \epsilon) + 0.25\epsilon}$ $p(E|s)$ $\boxed{0.25\ \epsilon}$

$p(S|s)$ $\boxed{0.25\ \epsilon}$ eeeeeeeeeeeeeeeeeee $p(W|s)$ $\boxed{0.25\ \epsilon}$

The solution should place equal probabilities on E, S, and W, and the rest should be placed on N. The probability for N should decrease linearly with ϵ.

(b) [2 pts] We also modify the reward original reward function $R(s, a, s')$ to visit more states and choose new actions. Which of the following rewards would encourage the agent to visit unseen states and actions?

$N(s, a)$ refers to the number of times that you have visited state s and taken action a in your samples.

- ☐ $R(s, a, s') + \sqrt{N(s, a)}$
- ■ $R(s, a, s') + \sqrt{\frac{1}{N(s,a)+1}}$
- ■ $\sqrt{\frac{1}{N(s,a)+1}}$
- ■ $R(s, a, s') - \sqrt{N(s, a)}$
- ☐ $-\sqrt{\frac{1}{N(s,a)+1}}$
- ■ $\exp(R(s, a, s') - N(s, a))$

The modified reward should be a monotonically decreasing function of N.

(c) [2 pts] Which of the following modified rewards will eventually converge to the optimal policy with respect to the original reward function $R(s, a, s')$? $N(s, a)$ is the same as defined in part (b).

- ☐ $R(s, a, s') + \sqrt{N(s, a)}$
- ■ $R(s, a, s') + \sqrt{\frac{1}{N(s,a)+1}}$
- ☐ $\sqrt{\frac{1}{N(s,a)+1}}$
- ☐ $R(s, a, s') - \sqrt{N(s, a)}$
- ☐ $-\sqrt{\frac{1}{N(s,a)+1}}$
- ☐ $\exp(R(s, a, s') - N(s, a))$

The modified reward should converge to the original reward as N increases.

SCRATCH PAPER – INTENTIONALLY BLANK – PLEASE DETACH ME

SCRATCH PAPER – INTENTIONALLY BLANK – PLEASE DETACH ME