# CS 188 Fall 2019 — Introduction to Artificial Intelligence — Practice Midterm

- You have approximately 110 minutes.
- The exam is closed book, closed calculator, and closed notes except your one-page crib sheet.
- 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.
- For multiple choice questions with *circular bubbles*, you should only mark ONE option; for those with *check-boxes*, you should mark ALL that apply (which can range from zero to all options). FILL in your answer COMPLETELY

| | |
|---|---|
| First name | |
| Last name | |
| SID | |
| Name of person on your left | |
| Name of person on your right | |

**Your Discussion/Exam Prep* TA (fill all that apply):**

- ☐ Shizhan (Tu)
- ☐ Carl (Tu)
- ☐ Emma (Tu)
- ☐ Mesut* (Tu)
- ☐ Jesse (Tu)
- ☐ Cathy (Tu)
- ☐ Peyrin* (Tu)
- ☐ Andy (Tu)
- ☐ Wilson (W)
- ☐ Ryan (W)
- ☐ Lindsay (W)
- ☐ Gokul* (W)
- ☐ Rachel (W)
- ☐ Henry* (W)
- ☐ Alan (W)
- ☐ Andreea (W)
- ☐ Chandan (W)
- ☐ Sherman* (W)
- ☐ Mike (W)
- ☐ Danny* (W)
- ☐ Jinkyu (W)
- ☐ Lawrence (W)
- ☐ Albert (W)

**For staff use only:**

| | | |
|---|---|---|
| Q1. | Potpourri | /10 |
| Q2. | Pushing Boxes | /12 |
| Q3. | Search Algorithms | /12 |
| Q4. | CSPs: Potluck Pandemonium | /13 |
| Q5. | Variants of Trees | /8 |
| Q6. | Reward Shaping | /21 |
| Q7. | Q-uagmire | /13 |
| | Total | /89 |

To earn the extra credit, one of the following has to hold true. Please circle and sign.

**A** I spent 110 or more minutes on the practice midterm.

**B** I spent fewer than 110 minutes on the practice midterm, but I believe I have solved all the questions.

**Signature:** _____

To submit the practice midterm, scan and upload the PDF to Gradescope.

SID: _____

# Q1. [10 pts] Potpourri

**(a)** Each True/False question is worth 2 points. Leaving a question blank is worth 0 points. **Answering incorrectly is worth −2 points.**

**(i)** [2.0 pts] [*true* or *false*] There exists some value of $k > 0$ such that the heuristic $h(n) = k$ is admissible.

**(ii)** [2.0 pts] [*true* or *false*] $A^*$ tree search using the heuristic $h(n) = k$ for some $k > 0$ is guaranteed to find the optimal solution.

**(b)** [2 pts] Consider a one-person game, where the one player's actions have non-deterministic outcomes. The player gets +1 utility for winning and -1 for losing. Mark *all* of the approaches that can be used to model and solve this game.

- ☐ Minimax with terminal values equal to +1 for wins and -1 for losses
- ☐ Expectimax with terminal values equal to +1 for wins and -1 for losses
- ☐ Value iteration with all rewards set to 0, except wins and losses, which are set to +1 and -1
- ☐ None of the above

**(c)** Suppose we run value iteration in an MDP with only non-negative rewards (that is, $R(s, a, s') \geq 0$ for any $(s, a, s')$). Let the values on the $k$th iteration be $V_k(s)$ and the optimal values be $V^*(s)$. Initially, the values are 0 (that is, $V_0(s) = 0$ for any $s$).

**(i)** [1 pt] Mark *all* of the options that are *guaranteed* to be true.

- ☐ For any $s, a, s'$, $V_1(s) = R(s, a, s')$
- ☐ For any $s, a, s'$, $V_1(s) \leq R(s, a, s')$
- ☐ For any $s, a, s'$, $V_1(s) \geq R(s, a, s')$
- ☐ None of the above are guaranteed to be true.

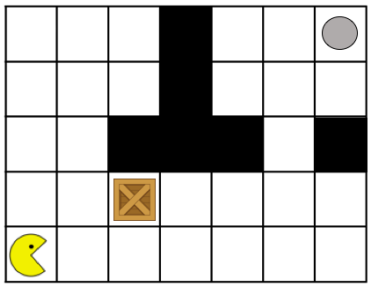
**(ii)** [1 pt] Mark *all* of the options that are *guaranteed* to be true.

- ☐ For any $k, s$, $V_k(s) = V^*(s)$
- ☐ For any $k, s$, $V_k(s) \leq V^*(s)$
- ☐ For any $k, s$, $V_k(s) \geq V^*(s)$
- ☐ None of the above are guaranteed to be true.

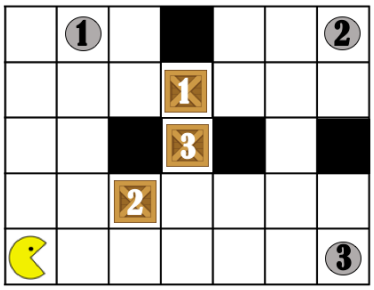
**(d)** [2 pts] Consider an arbitrary MDP where we perform $Q$-learning. Mark *all* of the options below in which we are guaranteed to learn the *optimal* $Q$-values. Assume that the learning rate $\alpha$ is reduced to 0 appropriately.

- ☐ During learning, the agent acts according to a suboptimal policy $\pi$. The learning phase continues until convergence.
- ☐ During learning, the agent chooses from the available actions at random. The learning phase continues until convergence.
- ☐ During learning, in state $s$, the agent chooses the action $a$ that it has chosen least often in state $s$, breaking ties randomly. The learning phase continues until convergence.
- ☐ During learning, in state $s$, the agent chooses the action $a$ that it has chosen most often in state $s$, breaking ties randomly. The learning phase continues until convergence.
- ☐ During learning, the agent always chooses from the available actions at random. The learning phase continues until each $(s, a)$ pair has been seen at least 10 times.

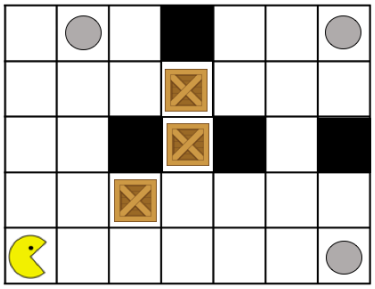
# Q2. [12 pts] Pushing Boxes



(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 $N$.

_____

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

_____

**(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 $N$.

_____

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

- ☐ $d_{\text{Man}}$(Pacman, button 1) + $d_{\text{Man}}$(Pacman, button 2) + $d_{\text{Man}}$(Pacman, button 3) - 3
- ☐ $d_{\text{Man}}$(box 1, button 1) + $d_{\text{Man}}$(box 2, button 2) + $d_{\text{Man}}$(box 3, button 3)
- ☐ $d_{\text{Man}}$(box 1, box 2) + $d_{\text{Man}}$(box 1, box 3)
- ☐ min($d_{\text{Man}}$(box 1, button 1), $d_{\text{Man}}$(box 2, button 2), $d_{\text{Man}}$(box 3, button 3))
- ☐ None of the above

**(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 $N$.

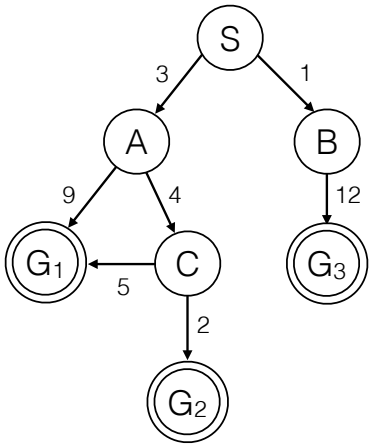
_____

**(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

SID: _____

# Q3. [12 pts] Search Algorithms




| | A | B | C | S |
|---|---|---|---|---|
| H-1 | 0 | 0 | 0 | 0 |
| H-2 | 6 | 7 | 1 | 7 |
| H-3 | 7 | 7 | 1 | 7 |
| H-4 | 4 | 7 | 1 | 7 |

**(a)** Consider the search graph and heuristics shown above. Select **all** of the goals that **could** be returned by each of the search algorithms below. For this question, if there is a tie on the fringe, assume the tie is broken **randomly**.

**(i)** [1 pt] DFS $G_1$ ◯ $G_2$ ◯ $G_3$ ◯

**(ii)** [1 pt] BFS $G_1$ ◯ $G_2$ ◯ $G_3$ ◯

**(iii)** [1 pt] UCS $G_1$ ◯ $G_2$ ◯ $G_3$ ◯

**(iv)** [1 pt] Greedy with H-1 $G_1$ ◯ $G_2$ ◯ $G_3$ ◯

**(v)** [1 pt] Greedy with H-2 $G_1$ ◯ $G_2$ ◯ $G_3$ ◯

**(vi)** [1 pt] Greedy with H-3 $G_1$ ◯ $G_2$ ◯ $G_3$ ◯

**(vii)** [1 pt] A* with H-2 $G_1$ ◯ $G_2$ ◯ $G_3$ ◯

**(viii)** [1 pt] A* with H-3 $G_1$ ◯ $G_2$ ◯ $G_3$ ◯

**(b)** For each heuristic, indicate whether it is consistent, admissible, or neither (select more than one option if appropriate):

**(i)** [1 pt] H-1 Consistent ◯ Admissible ◯ Neither ◯

**(ii)** [1 pt] H-2 Consistent ◯ Admissible ◯ Neither ◯

**(iii)** [1 pt] H-3 Consistent ◯ Admissible ◯ Neither ◯

**(iv)** [1 pt] H-4 Consistent ◯ Admissible ◯ Neither ◯

# Q4. [13 pts] CSPs: Potluck Pandemonium

The potluck is coming up and the staff haven't figured out what to bring yet! They've pooled their resources and determined that they can bring some subset of the following items.

1. Pho
2. Apricots
3. Frozen Yogurt
4. Fried Rice
5. Apple Pie
6. Animal Crackers

There are five people on the course staff: Taylor, Jonathan, Faraz, Brian, and Alvin. Each of them will only bring one item to the potluck.

1. If (F)araz brings the same item as someone else, it cannot be (B)rian.
2. (A)lvin has pho-phobia so he won't bring Pho, but he'll be okay if someone else brings it.
3. (B)rian is no longer allowed near a stove, so he can only bring items 2, 3, or 6.
4. (F)araz literally can't even; he won't bring items 2, 4, or 6.
5. (J)onathan was busy, so he didn't see the last third of the list. Therefore, he will only bring item 1, 2, 3, or 4.
6. (T)aylor will only bring an item that is before an item that (J)onathan brings.
7. (T)aylor is allergic to animal crackers, so he won't bring item 6. (If someone else brings it, he'll just stay away from that table.)
8. (F)araz and (J)onathan will only bring items that have the same first letter (e.g. Frozen Yogurt and Fried Rice).
9. (B)rian will only bring an item that is after an item that (A)lvin brings on the list.
10. (J)onathan and (T)aylor want to be unique; they won't bring the same item as anyone else.

**This page is repeated as the second-to-last page of this midterm for you to rip out and use for reference as you work through the problem.**

SID: _____

**(a)** [1 pt] Which of the listed constraints are unary constraints?

i ◯ ii ◯ iii ◯ iv ◯ v ◯

vi ◯ vii ◯ viii ◯ ix ◯ x ◯

**(b)** [2 pts] Rewrite implicit constraint viii. as an explicit constraint.

**(c)** [1 pt] How many edges are there in the constraint graph for this CSP?

**(d)** [2 pts] The table below shows the variable domains after all unary constraints have been enforced.

| | | | | | | |
|---|---|---|---|---|---|---|
| **A** | | 2 | 3 | 4 | 5 | 6 |
| **B** | | 2 | 3 | | | 6 |
| **F** | 1 | | 3 | | 5 | |
| **J** | 1 | 2 | 3 | 4 | | |
| **T** | 1 | 2 | 3 | 4 | 5 | |

Following the Minimum Remaining Values heuristic, which variable should we assign first? Break all ties alphabetically.

A ◯ B ◯ F ◯ J ◯ T ◯

**(e)** To decouple this from the previous question, assume that we choose to assign (F)araz first. In this question, we will choose which value to assign to using the Least Constraining Value method.

To determine the number of remaining values, enforce arc consistency to prune the domains. Then, count the total number of possible assignments (**not** the total number of remaining values). It may help you to enforce arc consistency twice, once before assigning values to (F)araz, and then again after assigning a value.

The domains after enforcing unary constraints are reproduced in each subquestion. The grids are provided as scratch space and **will not** be graded. Only numbers written in the blanks will be graded. The second grid is provided as a back-up in case you mess up on the first one. More grids are also provided on the second-to-last page of the exam.

**(i)** [2 pts] Assigning F = _____ results in _____ possible assignments.

| | | | | | | |
|---|---|---|---|---|---|---|
| **A** | | 2 | 3 | 4 | 5 | 6 |
| **B** | | 2 | 3 | | | 6 |
| **F** | 1 | | 3 | | 5 | |
| **J** | 1 | 2 | 3 | 4 | | |
| **T** | 1 | 2 | 3 | 4 | 5 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| **A** | | 2 | 3 | 4 | 5 | 6 |
| **B** | | 2 | 3 | | | 6 |
| **F** | 1 | | 3 | | 5 | |
| **J** | 1 | 2 | 3 | 4 | | |
| **T** | 1 | 2 | 3 | 4 | 5 | |

**(ii)** [2 pts] Assigning F = _____ results in _____ possible assignments.

| | | | | | | |
|---|---|---|---|---|---|---|
| **A** | | 2 | 3 | 4 | 5 | 6 |
| **B** | | 2 | 3 | | | 6 |
| **F** | 1 | | 3 | | 5 | |
| **J** | 1 | 2 | 3 | 4 | | |
| **T** | 1 | 2 | 3 | 4 | 5 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| **A** | | 2 | 3 | 4 | 5 | 6 |
| **B** | | 2 | 3 | | | 6 |
| **F** | 1 | | 3 | | 5 | |
| **J** | 1 | 2 | 3 | 4 | | |
| **T** | 1 | 2 | 3 | 4 | 5 | |

**(iii)** [2 pts] Assigning F = _____ results in _____ possible assignments.

| | | | | | | |
|---|---|---|---|---|---|---|
| **A** | | 2 | 3 | 4 | 5 | 6 |
| **B** | | 2 | 3 | | | 6 |
| **F** | 1 | | 3 | | 5 | |
| **J** | 1 | 2 | 3 | 4 | | |
| **T** | 1 | 2 | 3 | 4 | 5 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| **A** | | 2 | 3 | 4 | 5 | 6 |
| **B** | | 2 | 3 | | | 6 |
| **F** | 1 | | 3 | | 5 | |
| **J** | 1 | 2 | 3 | 4 | | |
| **T** | 1 | 2 | 3 | 4 | 5 | |

**(iv)** [1 pt] Using the LCV method, which value should we assign to F? If there is a tie, choose the lower number. (e.g. If both 1 and 2 have the same value, then fill 1.)

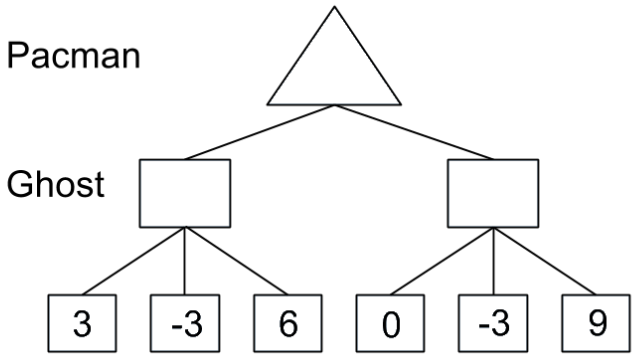
1 ◯ 2 ◯ 3 ◯ 4 ◯ 5 ◯ 6 ◯

SID: _____

# Q5. [8 pts] Variants of Trees

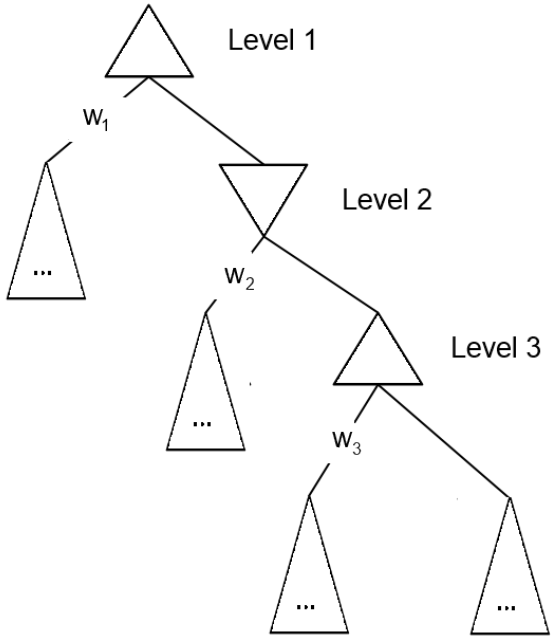
**(a)** Pacman is going to play against a careless ghost, which makes a move that is optimal for Pacman $\frac{1}{3}$ of the time, and makes a move that that minimizes Pacman's utility the other $\frac{2}{3}$ of the time.

**(i)** [2 pts] Fill in the correct utility values in the game tree below where Pacman is the maximizer:




**(ii)** [2 pts] Draw a complete game tree for the game above that contains only max nodes, min nodes, and chance nodes.

**(b)** Consider a modification of alpha-beta pruning where, rather than keeping track of a single value for $\alpha$ and $\beta$, you instead keep a list containing the best value, $w_i$, for the minimizer/maximizer (depending on the level) at each level up to and including the current level. Assume that the root node is always a max node. For example, consider the following game tree in which the first 3 levels are shown. When the considering the right child of the node at level 3, you have access to $w_1$, $w_2$, and $w_3$.




**(i)** [1 pt] Under this new scenario, what is the pruning condition for a max node at the $n^{th}$ level of the tree (in terms of $v$ and $w_1 ... w_n$)?

**(ii)** [1 pt] What is the pruning condition for a min node at the $n^{th}$ level of the tree?

**(iii)** [2 pts] What is the relationship between $\alpha$, $\beta$ and the list of $w_1 ... w_n$ at a max node at the $n^{th}$ level of the tree?

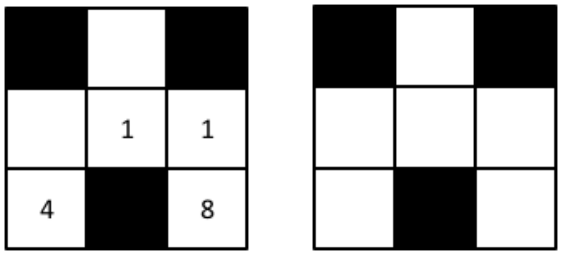
- ◯ $\sum_i w_i = \alpha + \beta$
- ◯ $\max_i w_i = \alpha, \quad \min_i w_i = \beta$
- ◯ $\min_i w_i = \alpha, \quad \max_i w_i = \beta$
- ◯ $w_n = \alpha, \quad w_{n-1} = \beta$
- ◯ $w_{n-1} = \alpha, \quad w_n = \beta$
- ◯ None of the above. The relationship is _____

SID: _____

# Q6. [21 pts] Reward Shaping

Consider the following Gridworld-like environment. The robot can move deterministically Up, Down, Right, or Left, or at any time it can exit to a terminal state (where it remains). The reward for any non-exit action is always 0. If the robot is on a square with a number written on it, it receives a reward of that size **on Exiting**. If the robot exits from any square without a number written on it, it receives a reward of 0 (and still exits to a terminal state). Note that when it is on any of the squares (including numbered squares), it can either move Up, Down, Right, Left or Exit. However, it only receives a non-zero reward when it Exits on a numbered square. **The robot is not required to exit on a numbered square; it can also move off of that square. However, if it does not exit, it does not get the reward.**

**(a)** [3 pts] Draw an arrow in **each** square (including numbered squares) in the following board on the right to indicate the optimal policy PacBot will calculate with the discount factor $\gamma = 0.5$ in the board on the left. (For example, if PacBot would move Down from the square in the middle on the left board, draw a down arrow in that square on the right board.) If PacBot's policy would be to exit from a particular square, draw an X instead of an arrow in that square.



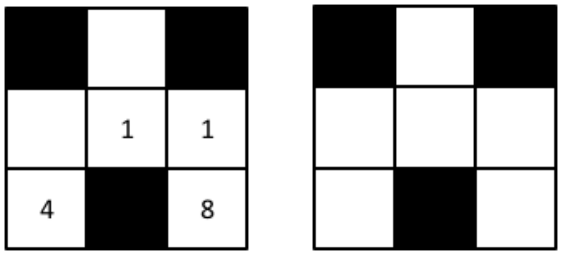

PacBot now operates in a new environment with an additional reward function $F(s, a, s')$, which is added to the original reward function $R(s, a, s')$ for every $(s, a, s')$ triplet, so that the new reward function $R'(s, a, s') = R(s, a, s') + F(s, a, s')$.

**(b)** [3 pts] Consider an additional reward $F_1$ that favors moving toward numbered squares. Let $d(s)$ be defined as the Manhattan distance from $s$ to the nearest numbered square. If $s$ is numbered, $d(s) = 0$.

$$F_1(s, a, s') = 6\left(d(s) - \frac{1}{2}d(s')\right).$$

$F_1$ is always 0 when $s'$ is a terminal state (equivalently, when $a$ is the Exit action).

Fill in the diagram as in (a) in the following board to indicate the optimal policy PacBot will calculate with the discount factor $\gamma = 0.5$ and the modified reward function $R'_1(s, a, s') = R(s, a, s') + F_1(s, a, s')$.



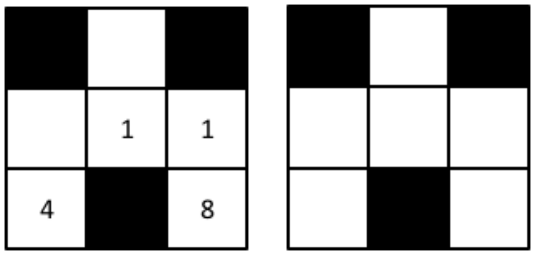

**(c)** [1 pt] If the robot now executes this policy $\pi$ in the **original** environment, without the extra added rewards $F$, what is $V^{\pi}(s)$ where $s$ is the top-most state?

**(d)** [3 pts] Consider a different artificial reward that also favors moving toward numbered squares in a slightly different way:

$$F_2(s,a,s') = \begin{cases} 6 & d(s') < d(s) \text{ i.e. } s' \text{ is closer to a numbered square than } s \text{ is,} \\ 0 & d(s') \geq d(s). \end{cases}$$

$F_2$ is always 0 when $s'$ is a terminal state (equivalently, when $a$ is the Exit action).

Fill in the diagram on the right as in (a) to indicate the optimal policy PacBot will calculate with the discount factor $\gamma = 0.5$ and the modified reward function $R_2'(s,a,s') = R(s,a,s') + F_2(s,a,s')$ in the board on the left.




**(e)** [1 pt] If the robot now executes this policy $\pi$ in the **original** environment, without the extra added rewards $F$, what is $V^\pi(s)$ where $s$ is the top-most state?

**(f)** [4 pts] For each of the following conditions on $F(s,a,s')$, state whether the condition is necessary and/or sufficient for the set of optimal policies to be unchanged in a general Gridworld-like MDP (i.e. an MDP following the rules laid out at the beginning of this question, but with any arbitrary board configuration) by adding $F$ to the reward function. Assume $\gamma = 1$, all states are reachable from all non-terminal states, and there is at least one positive number on the board. Note that the set of optimal policies is unchanged between a pair of MDPs when a policy is optimal in one MDP if and only if it is also optimal in the other.

**(i)** [1 pt] Condition 1: If $M$ is the maximum number on the board, then in the modified MDP, the set of all optimal policies is all policies such that no matter where you start, you will exit from a square showing $M$.

☐ necessary ☐ sufficient ☐ neither

**(ii)** [1 pt] Condition 2: If $M$ is the maximum number on the board, $|F(s,a,s')| \leq M$ for all $s, a, s'$ with $s'$ not a terminal state.

☐ necessary ☐ sufficient ☐ neither

**(iii)** [1 pt] Condition 3: The value function is unchanged; that is, $V'(s) = V(s)$ for all $s$, where $V'$ is the value function in the modified environment.

☐ necessary ☐ sufficient ☐ neither

**(iv)** [1 pt] Condition 4: The Q-value function is unchanged; that is, $Q'(s,a) = Q(s,a)$ for all $s$ and $a$, where $Q'$ is the Q-value function in the modified environment.

☐ necessary ☐ sufficient ☐ neither

SID: _____

Consider the following new Gridworld-like environment consisting of 8 states all in a row with all squares blank except the rightmost, which shows a 1. We restrict actions to Right, Left, and Exit. (The rules are exactly the same as in (a), except that the Up and Down actions are no longer available).


**(g)** In this environment, initialize a policy $\pi_0$ in which we exit at every state.

**(i)** [1 pt] Fill in the following diagram with values for $V^{\pi_0}$, the result of running policy evaluation with $\gamma = 1$. (For example, if the leftmost square has value 4 when following this policy, write a 4 in that square on the diagram. Note that this is the same MDP as above, but the 1 has been removed for your convenience. The robot still receives a reward of 1 for exiting from the rightmost square.)

**(ii)** [1 pt] Let $\pi_1$ be the new policy after one step of policy improvement. Break ties in the following order: Stop, Left, Right. As in (a), draw an arrow or an X in each square to represent this policy.

**(iii)** [1 pt] How many iterations of policy iteration are necessary to converge on the optimal policy?

**(h)** We now reintroduce the extra reward $F_1(s, a, s') = 6\left(d(s) - \frac{1}{2}d(s')\right)$ from part (b). For the rest of this question, we are working in a modified version of the long environment above where the reward function is $R'(s, a, s') = R(s, a, s') + F_1(s, a, s')$.

Once again, initialize a policy $\pi_0$ in which we exit at every state.

**(i)** [1 pt] As in (g), fill in the following diagram with values for $V^{\pi_0}$, the result of running policy evaluation with $\gamma = 1$ in the **modified** environment.

**(ii)** [1 pt] Let $\pi_1$ be the new policy after one step of policy improvement in the **modified** environment. Break ties in the following order: Stop, Left, Right. As in (a), draw an arrow or an X in each square to represent this policy.

**(iii)** [1 pt] How many iterations of policy iteration are necessary to converge on the optimal policy?

# Q7. [13 pts] Q-uagmire

Consider an unknown MDP with three states ($A$, $B$ and $C$) and two actions ($\leftarrow$ and $\rightarrow$). Suppose the agent chooses actions according to some policy $\pi$ in the unknown MDP, collecting a dataset consisting of samples $(s, a, s', r)$ representing taking action $a$ in state $s$ resulting in a transition to state $s'$ and a reward of $r$.

| $s$ | $a$ | $s'$ | $r$ |
|---|---|---|---|
| $A$ | $\rightarrow$ | $B$ | 2 |
| $C$ | $\leftarrow$ | $B$ | 2 |
| $B$ | $\rightarrow$ | $C$ | $-2$ |
| $A$ | $\rightarrow$ | $B$ | 4 |

You may assume a discount factor of $\gamma = 1$.

**(a)** Recall the update function of $Q$-learning is:

$$Q(s_t, a_t) \leftarrow (1-\alpha)Q(s_t, a_t) + \alpha\left(r_t + \gamma \max_{a'} Q(s_{t+1}, a')\right)$$

Assume that all $Q$-values are initialized to 0, and use a learning rate of $\alpha = \frac{1}{2}$.

**(i)** [4 pts] Run $Q$-learning on the above experience table and fill in the following $Q$-values:

$Q(A, \rightarrow) =$ _____ $Q(B, \rightarrow) =$ _____

**(ii)** [2 pts] After running $Q$-learning and producing the above $Q$-values, you construct a policy $\pi_Q$ that maximizes the $Q$-value in a given state:

$$\pi_Q(s) = \arg\max_a Q(s, a).$$

What are the actions chosen by the policy in states $A$ and $B$?

$\pi_Q(A)$ is equal to:

- ◯ $\pi_Q(A) = \leftarrow$.
- ◯ $\pi_Q(A) = \rightarrow$.
- ◯ $\pi_Q(A) =$ Undefined.

$\pi_Q(B)$ is equal to:

- ◯ $\pi_Q(B) = \leftarrow$.
- ◯ $\pi_Q(B) = \rightarrow$.
- ◯ $\pi_Q(B) =$ Undefined.

**(b)** [3 pts] Use the empirical frequency count model-based reinforcement learning method described in lectures to estimate the transition function $\hat{T}(s, a, s')$ and reward function $\hat{R}(s, a, s')$. (Do not use pseudocounts; if a transition is not observed, it has a count of 0.)

Write down the following quantities. You may write N/A for undefined quantities.

$\hat{T}(A, \rightarrow, B) =$ _____ $\hat{R}(A, \rightarrow, B) =$ _____

$\hat{T}(B, \rightarrow, A) =$ _____ $\hat{R}(B, \rightarrow, A) =$ _____

$\hat{T}(B, \leftarrow, A) =$ _____ $\hat{R}(B, \leftarrow, A) =$ _____

**(c)** This question considers properties of reinforcement learning algorithms for *arbitrary* discrete MDPs; you do not need to refer to the MDP considered in the previous parts.

**(i)** [2 pts] Which of the following methods, at convergence, provide enough information to obtain an optimal policy? (Assume adequate exploration.)

- ☐ Model-based learning of $T(s, a, s')$ and $R(s, a, s')$.
- ☐ Direct Evaluation to estimate $V(s)$.
- ☐ Temporal Difference learning to estimate $V(s)$.
- ☐ Q-Learning to estimate $Q(s, a)$.

**(ii)** [2 pts] In the limit of infinite timesteps, under which of the following exploration policies is $Q$-learning guaranteed to converge to the optimal Q-values for all state? (You may assume the learning rate $\alpha$ is chosen appropriately, and that the MDP is ergodic: i.e., every state is reachable from every other state with non-zero probability.)

- ☐ A fixed policy taking actions uniformly at random.
- ☐ A greedy policy.
- ☐ An $\epsilon$-greedy policy
- ☐ A fixed optimal policy.