# CS 188 Fall 2017 — Introduction to Artificial Intelligence — Midterm 1

- 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.
- 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 | |
| edX username | |

| | |
|---|---|
| Name of person on your left | |
| Name of person on your right | |

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

☐ Brijen (Tu) ☐ Wenjing (Tu) ☐ Caryn (W) ☐ Andy* (Tu)

☐ Peter (Tu) ☐ Aaron (W) ☐ Anwar (W) ☐ Nikita* (Tu)

☐ David (Tu) ☐ Mitchell (W) ☐ Aarash (W) ☐ Daniel* (W)

☐ Nipun (Tu) ☐ Abhishek (W) ☐ Yuchen* (Tu) ☐ Shea* (W)

**For staff use only:**

| | | |
|---|---|---|
| Q1. | CSP: Air Traffic Control | /15 |
| Q2. | Utility | /10 |
| Q3. | State Representations and State Spaces | /14 |
| Q4. | Informed Search and Heuristics | /14 |
| Q5. | Strange MDPs | /16 |
| Q6. | Reinforcement Learning | /15 |
| Q7. | MedianMiniMax | /16 |
| | Total | /100 |

THIS PAGE IS INTENTIONALLY LEFT BLANK. NOTHING ON THIS PAGE WILL BE GRADED.

SID: _____

# Q1. [15 pts] CSP: Air Traffic Control

We have five planes: A, B, C, D, and E and two runways: international and domestic. We would like to schedule a time slot and runway for each aircraft to **either** land or take off. We have four time slots: $\{1, 2, 3, 4\}$ for each runway, during which we can schedule a landing or take off of a plane. We must find an assignment that meets the following constraints:

- Plane B has lost an engine and must land in time slot 1.
- Plane D can only arrive at the airport to land during or after time slot 3.
- Plane A is running low on fuel but can last until at most time slot 2.
- Plane D must land before plane C takes off, because some passengers must transfer from D to C.
- No two aircrafts can reserve the same time slot for the same runway.

**(a)** [3 pts] Complete the formulation of this problem as a CSP in terms of variables, domains, and constraints (both unary and binary). Constraints should be expressed implicitly using mathematical or logical notation rather than with words.

**Variables:** A, B, C, D, E for each plane. **Domains:** _____

**Constraints:** (you do not have to use all lines)

_____ _____

_____ _____

_____ _____

_____ _____

**(b)** For the following subparts, we add the following two constraints:

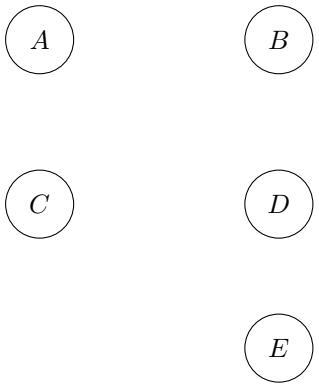
- Planes A, B, and C cater to international flights and can only use the international runway.
- Planes D and E cater to domestic flights and can only use the domestic runway.

**(i)** [2 pts] With the addition of the two constraints above, we completely reformulate the CSP. You are given the variables and domains of the new formulation. Complete the constraint graph for this problem given the original constraints and the two added ones.

**Variables:** A, B, C, D, E for each plane.

**Domains:** $\{1, 2, 3, 4\}$

**Constraint Graph:**



**(ii)** [4 pts] What are the domains of the variables after enforcing arc-consistency? Begin by enforcing unary constraints. (Cross out values that are no longer in the domain.)

| | | | | |
|---|---|---|---|---|
| A | 1 | 2 | 3 | 4 |
| B | 1 | 2 | 3 | 4 |
| C | 1 | 2 | 3 | 4 |
| D | 1 | 2 | 3 | 4 |
| E | 1 | 2 | 3 | 4 |

**(iii)** [4 pts] Arc-consistency can be rather expensive to enforce, and we believe that we can obtain faster solutions using only **forward-checking** on our variable assignments. Using the Minimum Remaining Values heuristic, perform backtracking search on the graph, breaking ties by picking lower values and characters first. List the $(variable, assignment)$ pairs in the order they occur (including the assignments that are reverted upon reaching a dead end). Enforce unary constraints before starting the search.

(You don't have to use this table, it won't be graded.)

| | | | | |
|---|---|---|---|---|
| A | 1 | 2 | 3 | 4 |
| B | 1 | 2 | 3 | 4 |
| C | 1 | 2 | 3 | 4 |
| D | 1 | 2 | 3 | 4 |
| E | 1 | 2 | 3 | 4 |

**Answer:**

**(c)** [2 pts] Suppose we have just one runway and $n$ planes, where no two planes can use the runway at once. We are assured that the constraint graph will always be tree-structured and that a solution exists. What is the runtime complexity in terms of the number of planes, $n$, of a CSP solver that runs arc-consistency and then assigns variables in a topological ordering?

- ◯ $O(1)$
- ◯ $O(n)$
- ◯ $O(n^2)$
- ◯ $O(n^3)$
- ◯ $O(n^n)$
- ◯ None of the Above

SID: _____

# Q2. [10 pts] Utility

**(a)** [4 pts] Pacwoman is enticed by a lottery at the local Ghostway. Ghostway is offering a lottery which rewards three different outcomes at equal probability: 2 dollars, 5 dollars and 6 dollars. The price to purchase a lottery ticket is 2 dollars. The ghost at the cash register, however, is now offering Pacwoman an option: give him a bribe of 1 dollar more and he'll manipulate the lottery so that she will not get the worst outcome (the rest of the options have equal likelihood). Pacwoman's utility is dependent on her net gain, $p$. For which of the following utility functions, should Pacwoman decide to bribe the ghost?

- ☐ $U(p) = p$
- ☐ $U(p) = p^2$
- ☐ None of the Above

**(b)** You are in a pack of 9 Pacpeople about to embark on a mission to steal food from the ghosts. In order to proceed, the pack needs a Pacleader to direct the mission.

Being a leader is very tiring, so the Pacleader would need to spend 5 food pellets, to consume for energy to lead. If the mission is successful, however, the Pacleader gets 1/5th of the food that is scavenged. The utility of the Pacleader $U_L(l, t)$ depends on their **individual food net gain** ($l$) and on the **total food** collected for the pack ($t$).

The remaining non-leading Pacmembers get an equal share of the remaining 4/5ths of the food. A Pacfollower gains utility $U_f(f)$, which is dependent on only their **individual food net gain** ($f$).

Let $X$ be the probability of going home with no food, otherwise the mission is successful and brings back 100 pellets. For what value of $X$, should you, a rational Pacperson, decide to step up and lead the pack? Assuming that if you don't, some other Pacmember will.

**(i)** [4 pts] Express your answer in terms of $X, U_L, U_f$ (You do not have to simplify for X):

**(ii)** [2 pts] Calculate an explicit expression for X given:

$$U_L(l, t) = 5l + \frac{t}{5}$$
$$U_f(f) = f^2 - 4f$$

where,
$l$ is the individual food net gain of the Pacleader
$t$ is the total amount of food scavenged for the pack
$f$ is the individual food net gain of a Pacfollower

# Q3. [14 pts] State Representations and State Spaces

For each part, state the size of a minimal state space for the problem. Give your answer as an expression that references problem variables. Below each term, state what information it encodes. For example, you could write $2 \times MN$ and write "whether a power pellet is in effect" under the 2 and "Pacman's position" under the $MN$. State spaces which are complete but not minimal will receive partial credit.

Each part is independent. A maze has **height $M$ and width $N$**. A Pacman can move *NORTH, SOUTH, EAST, or WEST*. There is initially a **pellet in every position** of the maze. The **goal is to eat all of the pellets.**

**(a)** [4 pts] Personal Space

In this part, there are $P$ Pacmen, numbered $1, \ldots, P$. Their turns cycle so Pacman 1 moves, then Pacman 2 moves, and so on. Pacman 1 moves again after Pacman $P$. Any time two Pacmen enter adjacent positions, the one with the lower number dies and is removed from the maze.

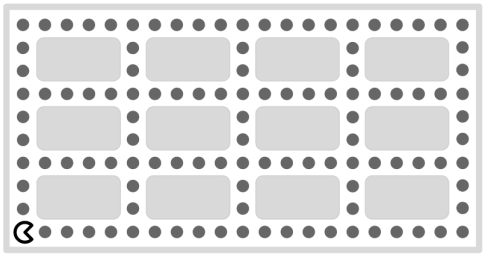
**Answer:**

**(b)** [4 pts] Road Not Taken

In this part, there is one Pacman. Whenever Pacman enters a position which he has visited previously, the maze is reset – each position gets refilled with food and the "visited status" of each position is reset as well.

**Answer:**

**(c)** [6 pts] Hallways



In this part, there is one Pacman. The walls are arranged such that they create a grid of $H$ hallways **total**, which connect at $I$ intersections. (In the example above, $H = 9$ and $I = 20$). In a single action, Pacman can move from one intersection into an adjacent intersection, eating all the dots along the way. Your answer should only depend on $I$ and $H$.

(note: $H$ = number of vertical hallways + number of horizontal hallways)

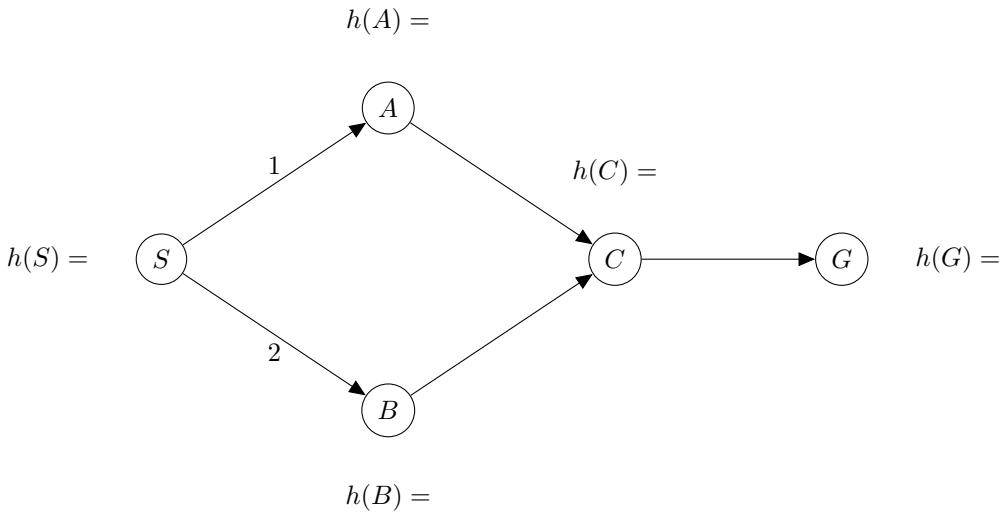
**Answer:**

SID: _____

# Q4. [14 pts] Informed Search and Heuristics

**(a)** [6 pts] Consider the state space shown below, with starting state $S$ and goal state $G$. Fill in a cost from the set $\{1, 2\}$ **for each blank edge** and a heuristic value from the set $\{0, 1, 2, 3\}$ **for each node** such that the following properties are satisfied:

- The heuristic is admissible but not consistent.
- The heuristic is monotonic non-increasing along paths from the start state to the goal state.
- A* graph search finds a suboptimal solution.
- You will never encounter ties (two elements in the fringe with the same priority) during execution of A*.



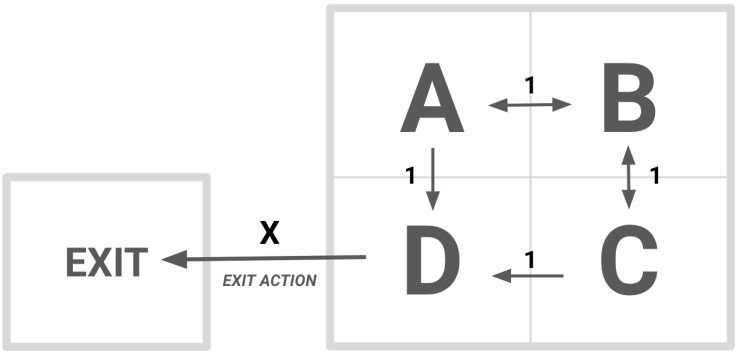

**(b)** [8 pts] **Don't spend all your time on this question.** As we saw in class, A* graph search with a consistent heuristic will always find an optimal solution when run on a problem with a finite state space. However, if we turn to problems with infinite state spaces, this property may no longer hold. Your task in this question is to provide a concrete example with an infinite state space where A* graph search fails to terminate or fails to find an optimal solution.

Specifically, you should describe the state space, the starting state, the goal state, the heuristic value at **each node**, and the cost of **each transition**. Your heuristic should be consistent, and all step costs should be strictly greater than zero ($cost \in \mathbb{R}_{>0}$) to avoid trivial paths with zero cost. To keep things simple, each state should have a finite number of successors, and the goal state should be reachable in a finite number of actions from the starting state.

You may want to start by drawing a diagram.

# Q5. [16 pts] Strange MDPs

In this MDP, the available actions at **state A, B, C** are *LEFT, RIGHT, UP,* and *DOWN* unless there is a wall in that direction. The only action at **state D** is the *EXIT ACTION* and gives the agent a **reward of** $x$. The **reward for non-exit actions is always 1**.




**(a)** [6 pts] Let all actions be deterministic. Assume $\gamma = \frac{1}{2}$. Express the following in terms of $x$.

$V^*(D) =$ $V^*(C) =$

$V^*(A) =$ $V^*(B) =$

**(b)** [6 pts] Let any non-exit action be successful with probability $= \frac{1}{2}$. Otherwise, the agent stays in the same state with reward $= 0$. The *EXIT ACTION* from the **state D** is still deterministic and will always succeed. Assume that $\gamma = \frac{1}{2}$.

For which value of $x$ does $Q^*(A, DOWN) = Q^*(A, RIGHT)$? Box your answer and justify/show your work.

**(c)** [4 pts] We now add one more layer of complexity. Turns out that the reward function is not guaranteed to give a particular reward when the agent takes an action. Every time an agent transitions from one state to another, once the agent reaches the new state $s'$, a fair 6-sided dice is rolled. If the dices lands with value $x$, the agent receives the reward $R(s, a, s') + x$. The sides of dice have value $1, 2, 3, 4, 5$ and $6$.

Write down the new bellman update equation for $V_{k+1}(s)$ in terms of $T(s, a, s')$, $R(s, a, s')$, $V_k(s')$, and $\gamma$.

SID: _____

# Q6. [15 pts] Reinforcement Learning

Imagine an unknown environments with four states (A, B, C, and X), two actions ($\leftarrow$ and $\rightarrow$). An agent acting in this environment has recorded the following episode:

| s | a | s' | r | Q-learning iteration numbers (for part b) |
|---|---|---|---|---|
| A | $\rightarrow$ | B | 0 | 1, 10, 19, ... |
| B | $\rightarrow$ | C | 0 | 2, 11, 20, ... |
| C | $\leftarrow$ | B | 0 | 3, 12, 21, ... |
| B | $\leftarrow$ | A | 0 | 4, 13, 22, ... |
| A | $\rightarrow$ | B | 0 | 5, 14, 23, ... |
| B | $\rightarrow$ | A | 0 | 6, 15, 24, ... |
| A | $\rightarrow$ | B | 0 | 7, 16, 25, ... |
| B | $\rightarrow$ | C | 0 | 8, 17, 26, ... |
| C | $\rightarrow$ | X | 1 | 9, 18, 27, ... |

**(a)** [4 pts] Consider running model-based reinforcement learning based on the episode above. Calculate the following quantities:

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

$\hat{R}(C, \rightarrow, X) =$ _____

**(b)** [5 pts] Now consider running Q-learning, repeating the above series of transitions in an infinite sequence. Each transition is seen at multiple iterations of Q-learning, with iteration numbers shown in the table above.

After which iteration of Q-learning do the following quantities first become nonzero? (If they always remain zero, write *never*).

$Q(A, \rightarrow)$? _____

$Q(B, \leftarrow)$? _____

**(c)** [6 pts] True/False: For each question, you will get positive points for correct answers, zero for blanks, and negative points for incorrect answers. Circle your answer **clearly**, or it will be considered incorrect.

**(i)** [1.5 pts] [*true* or *false*] In Q-learning, you do not learn the model.

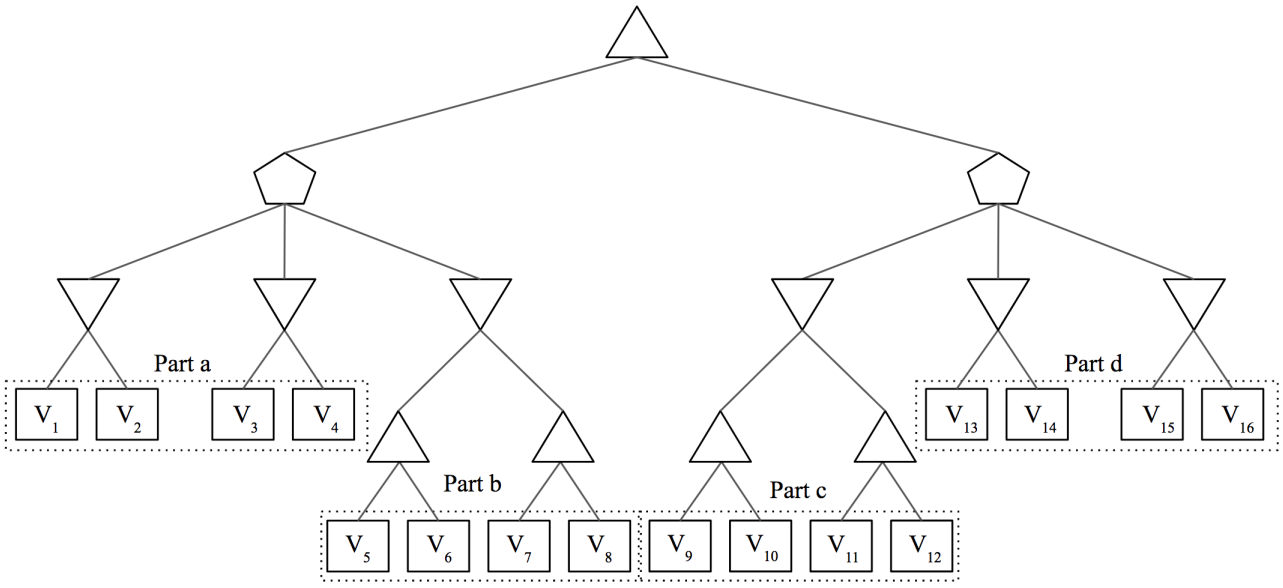
**(ii)** [1.5 pts] [*true* or *false*] For TD Learning, if I multiply all the rewards in my update by some nonzero scalar $p$, the algorithm is still guaranteed to find the optimal policy.

**(iii)** [1.5 pts] [*true* or *false*] In Direct Evaluation, you recalculate state values after each transition you experience.

**(iv)** [1.5 pts] [*true* or *false*] Q-learning requires that all samples must be from the optimal policy to find optimal q-values.

# Q7. [16 pts] MedianMiniMax

You're living in utopia! Despite living in utopia, you still believe that you need to maximize your utility in life, other people want to minimize your utility, and the world is a 0 sum game. But because you live in utopia, a benevolent social planner occasionally steps in and chooses an option that is a compromise. Essentially, the social planner (represented as the pentagon) is a median node that chooses the successor with median utility. Your struggle with your fellow citizens can be modelled as follows:




There are some nodes that we are sometimes able to prune. In each part, mark all of the terminal nodes such that **there exists a possible situation** for which the node **can be pruned**. In other words, you must consider **all** possible pruning situations. Assume that evaluation order is **left to right** and all $V_i$'s are **distinct.**

Note that as long as there exists ANY pruning situation (does not have to be the same situation for every node), you should mark the node as prunable. Also, alpha-beta pruning does not apply here, simply prune a sub-tree when you can reason that its value will not affect your final utility.

**(a)**
- ☐ $V_1$
- ☐ $V_2$
- ☐ $V_3$
- ☐ $V_4$
- ☐ None

**(b)**
- ☐ $V_5$
- ☐ $V_6$
- ☐ $V_7$
- ☐ $V_8$
- ☐ None

**(c)**
- ☐ $V_9$
- ☐ $V_{10}$
- ☐ $V_{11}$
- ☐ $V_{12}$
- ☐ None

**(d)**
- ☐ $V_{13}$
- ☐ $V_{14}$
- ☐ $V_{15}$
- ☐ $V_{16}$
- ☐ None