CS221 Problem Workout Solutions

1) [CA session] Problem 1

(a) Sabina wants to go from her house (located at 1) to the gym (located at n). At each location s, she can either (i) deterministically walk forward to the next location $s+1$ (takes 1 unit of time) or (ii) wait for the bus. The bus comes with probability ϵ, in which case, she will reach the gym in $1+\alpha(n-s)$ units of time, where α is some parameter. If the bus doesn't come, well, she stays put, and that takes 1 unit of time.

We have formalized the problem as an MDP for you:

- State: $s \in \{1, 2, \ldots, n\}$ is Sabina's location
- Actions$(s) = \{\text{Walk}, \text{Bus}\}$
- $\text{Reward}(s, \text{Walk}, s') = \begin{cases} -1 & \text{if } s' = s+1 \\ -\infty & \text{otherwise} \end{cases}$
- $\text{Reward}(s, \text{Bus}, s') = \begin{cases} -1-\alpha(n-s) & \text{if } s' = n \\ -1 & \text{if } s' = s \\ -\infty & \text{otherwise} \end{cases}$
- $T(s'|s, \text{Walk}) = \begin{cases} 1 & \text{if } s' = s+1 \\ 0 & \text{otherwise} \end{cases}$
- $T(s'|s, \text{Bus}) = \begin{cases} \epsilon & \text{if } s' = n \\ 1-\epsilon & \text{if } s' = s \\ 0 & \text{otherwise} \end{cases}$
- IsEnd$(s) = \mathbf{1}[s = n]$

Compute a closed form expression for the value of the "always walk" policy and the "always wait for the bus" policy (using some or all of the variables ϵ, α, n). Assume a discount rate of $\gamma = 1$.

- $V_{\text{Walk}}(s) =$ _____

- $V_{\text{Bus}}(s) =$ _____

- For what values of ϵ (as a function of α and n) is it advantageous to walk rather than take the bus?

Solution Expected value for always walking:

$$V_{\text{Walk}} = -(n - s).$$

Expected value for always waiting for bus:

$$V_{\text{Bus}}(s) = \epsilon(-1 - \alpha(n - s)) + (1 - \epsilon)(-1 + V_{\text{Bus}}(s)).$$

Simplifying, we get:

$$V_{\text{Bus}}(s) = -\alpha(n - s) - \frac{1}{\epsilon}.$$

For walking to be preferable, we need $V_{\text{Walk}}(s) \geq V_{\text{Bus}}(s)$, or equivalently:

$$n - s \leq \alpha(n - s) + \frac{1}{\epsilon} \Leftrightarrow (1 - \alpha)(n - s) \leq \frac{1}{\epsilon} \Leftrightarrow \begin{cases} \epsilon \leq \frac{1}{(1-\alpha)(n-s)} & , \alpha < 1 \\ \epsilon > 0 & , \alpha \geq 1. \end{cases}$$

(b) Not surprisingly, buses operate strangely in this town, and we will now assume instead that Sabina doesn't know the reward function nor the transition probabilities. She decides to use reinforcement learning to find out. She starts by going around town using the two different modes of transportation:

s_0	a_1	r_1	s_1	a_2	r_2	s_2	a_3	r_3	s_3	a_4	r_4	s_4	a_5	r_5	s_5
1	Bus	-1	1	Bus	-1	1	Bus	3	3	Walk	1	4	Walk	1	5

Run the Q-learning algorithm once over the given data to compute an estimate of the optimal Q-value $Q_{\text{opt}}(s, a)$. Process the episodes from left to right. Assume all Q-values are initialized to zero, and use a learning rate of $\eta = 0.5$ and a discount of $\gamma = 1$.

- $\hat{Q}(1, \text{Walk}) =$ _____
- $\hat{Q}(1, \text{Bus}) =$ _____
- $\hat{Q}(3, \text{Walk}) =$ _____
- $\hat{Q}(3, \text{Bus}) =$ _____
- $\hat{Q}(4, \text{Walk}) =$ _____
- $\hat{Q}(4, \text{Bus}) =$ _____

Solution On each (s, a, r, s'), recall the Q-learning updates:

$$\hat{Q}_{opt}(s,a) \leftarrow (1-\eta)\hat{Q}_{opt}(s,a) + \eta(r + \gamma \max_{a' \in \text{Actions}(s')} \hat{Q}_{opt}(s', a')). \tag{1}$$

Now the concrete updates:

- On $(1, \text{Bus}, -1, 1)$: $\hat{Q}(1, \text{Bus}) = 0.5(0) + 0.5(-1 + 1(\max(0, 0))) = -0.5$
- On $(1, \text{Bus}, -1, 1)$: $\hat{Q}(1, \text{Bus}) = 0.5(-0.5) + 0.5(-1 + 1(\max(0, -0.5))) = -0.75$
- On $(1, \text{Bus}, 3, 3)$: $\hat{Q}(1, \text{Bus}) = 0.5(-0.75) + 0.5(3 + 1(\max(0, 0))) = 1.125$
- On $(3, \text{Walk}, 1, 4)$: $\hat{Q}(3, \text{Walk}) = 0.5(0) + 0.5(1 + 1(\max(0, 0))) = 0.5$
- On $(4, \text{Walk}, 1, 5)$: $\hat{Q}(4, \text{Walk}) = 0.5(0) + 0.5(1 + 1(\max(0, 0))) = 0.5$

The final values:

- $\hat{Q}(1, \text{Walk}) = 0$
- $\hat{Q}(1, \text{Bus}) = 1.125$
- $\hat{Q}(3, \text{Walk}) = 0.5$
- $\hat{Q}(3, \text{Bus}) = 0$
- $\hat{Q}(4, \text{Walk}) = 0.5$
- $\hat{Q}(4, \text{Bus}) = 0$

2) [CA Session] Problem 2

You're programming a self-driving car that can take you from home (position 1) to school (position n). At each time step, the car has a current position $x \in \{1, \ldots, n\}$ and a current velocity $v \in \{0, \ldots, m\}$. The car starts with $v = 0$, and at each time step, the car can either increase the velocity by 1, decrease it by 1, or keep it the same; this new velocity is used to advance x to the new position. The velocity is not allowed to exceed the speed limit m nor return to 0.

In addition, to prevent people from recklessly cruising down Serra Mall, the university has installed speed bumps at a subset of the n locations. The speed bumps are located at $B \subseteq \{1, \ldots, n\}$. The car is not allowed to enter, leave, or pass over a speed bump with velocity more than $k \in \{1, \ldots, m\}$. **Your goal is to arrive at position n with velocity 1 in the smallest number of time steps.**

Figure 1 shows an example with $n = 9$ positions and one speed bump $B = \{5\}$. If the maximum speed is $m = 3$ and $k = 1$ for a speed bump, then an example of a legal path is the following:

$$(1,0) \xrightarrow{+1} (2,1) \xrightarrow{+1} (4,2) \xrightarrow{-1} (5,1) \xrightarrow{0} (6,1) \xrightarrow{+1} (8,2) \xrightarrow{-1} (9,1)$$

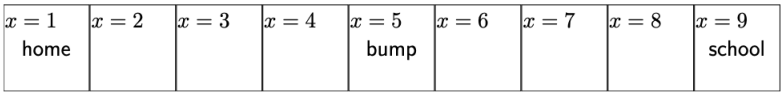


Figure 1: An example of a legal path that takes 6 time steps with $m = 3$ and $k = 1$. We show the position-velocity pairs (x, v) at each time step, and each number above an arrow is an acceleration (change in velocity).

(a) It turns out that you were so excited about the AI algorithms that you didn't really pay much attention to the brakes of the car. As a result, when you try to decrease the velocity by 1, with some failure probability α, the velocity actually stays the same. To simplify our lives, assume there are no speed bumps. Assume a reward of R if we get to school (at a velocity of 1) but -1 if we pass the school, with a cost of 1 per time step. Let us formulate the resulting problem as an MDP:

- $s_{\text{start}} = (1, 0)$
- $\text{Actions}((x, v)) = \{a \in \{+1, 0, -1\} : x + v + a \leq n \wedge v + a \leq m \wedge (v + a \leq k \vee \{x, \ldots, x + v + a\} \cap B = \emptyset)\}$. Suppose we want to apply acceleration a. First, we want to make sure we don't exceed the school ($x + v + a \leq n$) or go out of the velocity range ($v + a \leq m$). Next, we want to make sure that we're not entering, passing through, or leaving any speed bumps at a velocity greater than k. This is captured logically by ensuring a safe speed ($v + a \leq k$) or checking that there are no speed bumps between x and the new location $x + v + a$.

- $T((x', v')|(x, v), a) =$ (to be filled out by you below)
- $\text{Reward}((x, v), a, (x', v')) = R \cdot \mathbb{1}[x' = n \wedge v' = 1] - 1$
- $\text{IsEnd}((x, v)) = \mathbb{1}[x \geq n]$

(i) Fill out the definition of the transition probabilities T:

$T((x', v')|(x, v), a) =$

Solution

$$T((x', v')|(x, v), a) = \begin{cases} \alpha & \text{if } x' = x + v' \text{ and } v' = v \text{ and } a = -1 \\ 1 - \alpha & \text{if } x' = x + v' \text{ and } v' = v + a \text{ and } a = -1 \\ 1 & \text{if } x' = x + v' \text{ and } v' = v + a \text{ and } a \neq -1 \\ 0 & \text{otherwise.} \end{cases}$$

(ii) Let us explore the effect of unreliable brakes. Consider the example in Figure2.

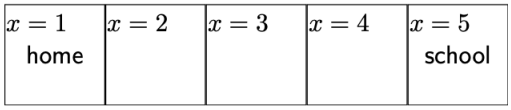


Figure 2: An small driving environment without speed bumps.

Consider two policies:

- π_1: always move with velocity 1:

$$\pi_1((1,0)) = +1 \quad \pi_1((2,1)) = 0 \quad \pi_1((3,1)) = 0 \quad \pi_1((4,1)) = 0.$$

- π_2: speed up and slow down:

$$\pi_2((1,0)) = +1 \quad \pi_2((2,1)) = +1 \quad \pi_2((4,2)) = -1.$$

Compute the expected utility of π_1 as a function of α and R (with discount $\gamma = 1$).

Solution The policy π_1 deterministically obtains reward $\boxed{R-4}$. Using $Reward(x', v')$ we have $Reward(2,1) + Reward(3,1) + Reward(4,1) + Reward(5,1) = -1 - 1 - 1 - 1 + R$

Compute the expected utility of π_2 as a function of α and R (with discount $\gamma = 1$).

Solution The policy π_2 obtains reward $\boxed{(1-\alpha)R - 3}$. We only get reward R if we are able to break at the end so: $(1-\alpha)(Reward(2,1) + Reward(4,2) + Reward(5,1)) + \alpha$
$(Reward(2,1) + Reward(4,2) + Reward(6,2)) = (1-\alpha)(R-3) + \alpha(-3) = (1-\alpha)R - 3$

For what values of α and R does π_2 obtain higher expected reward than π_1? Your answer should be an expression relating α and R.

Solution Therefore, π_2 is better when $(1-\alpha)R - 3 > R - 4$, which is precisely when $\boxed{\alpha < 1/R}$.

(b) Bad news: you realize that your brakes are not only faulty, but that you don't know how often they fail (α is unknown). Circle all of the following algorithms that can be used to compute the optimal policy in this setting:

model-based value iteration model-free Monte Carlo SARSA Q-learning

Solution **Model-based value iteration** would estimate the transition probabilities, which can be used to compute the optimal policy. **Q-learning** can be used to directly estimate the value of the optimal policy. Model-free Monte Carlo and SARSA can only be used to compute the value of a fixed policy.

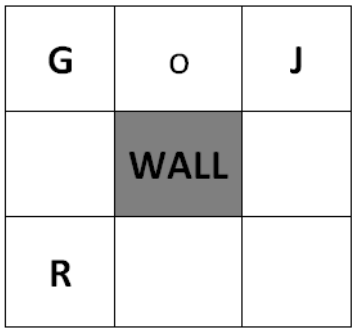
3) [CA session] Problem 3

After finally meeting up, Romeo (R) and Juliet (J) decide to try to catch a goose (G) to keep as a pet. Eventually, they chase it into a 3×3 hedge maze show below. Now they play the following turn-based game:

(a) The Goose moves either Down or Right.

(b) Romeo moves either Up or Right.

(c) Juliet moves either Left or Down.



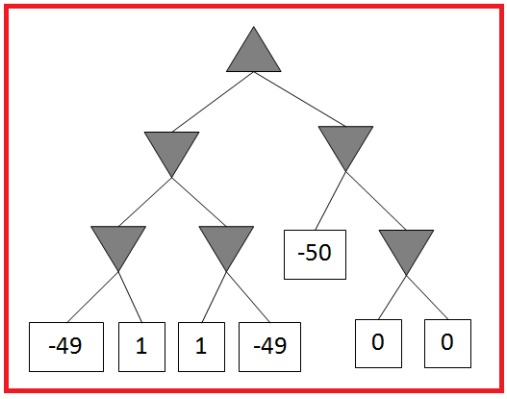
Participants: Goose (G), Romeo (R), Juliet (J), bread (o)

If the Goose enters the square with bread, it gets a reward 1. If either Romeo or Juliet enters the same square as the Goose, they catch it and the Goose gets a reward of -50. The game ends when either the Goose has been caught or everyone has moved once. Note that it is possible for the Goose to get both rewards.

Construct a depth one minimax tree for the above situation, with the Goose as the maximizer and Juliet and Romeo as the minimizers. Use up-triangles Δ for max nodes, down-triangles ∇ for min nodes, and square nodes for the leaves. Label each node with its minimax value.

What is the minimax value of the game if Romeo defects and becomes a maximizer?

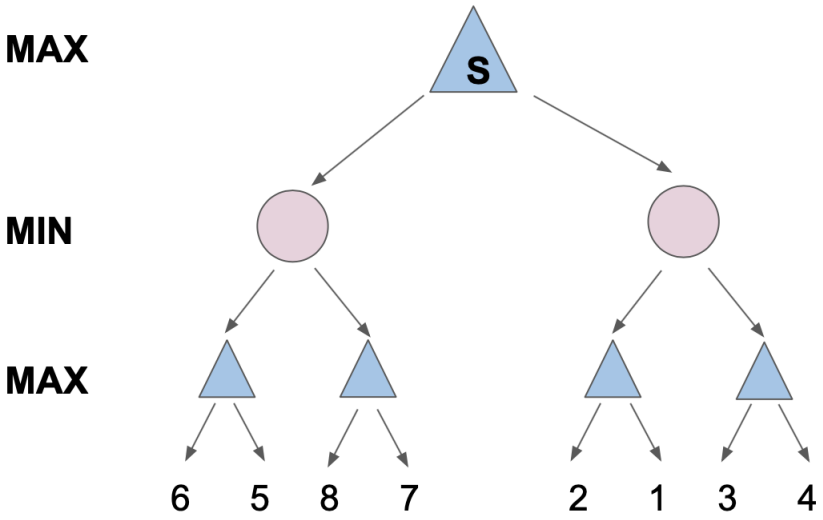
Solution Here is the minimax tree:



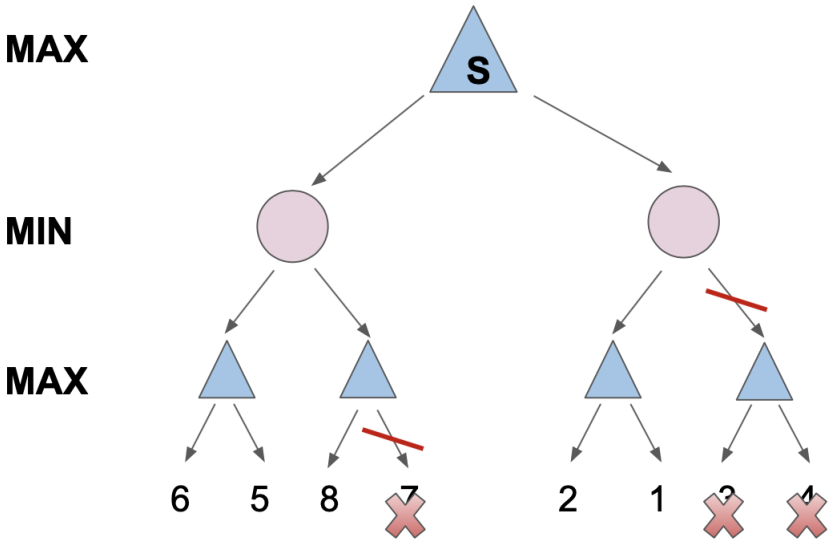
The value of the game is -49 (the goose might as well go for the bread before it gets caught). If Romeo defects, then the value of the game is 0 (the Goose moves towards Romeo).

4) **[CA session] Problem 4**

Consider running alpha-beta pruning on the following minimax tree. The children of each node will be expanded from left to right. Which nodes will be pruned (thus not being visited)?



Solution The leaves with values 7, 3 and 4 will be pruned.



5) [CA Session] Problem 5

You're programming a self-driving car that can take you from home (position 1) to school (position n). At each time step, the car has a current position $x \in \{1, \ldots, n\}$ and a current velocity $v \in \{0, \ldots, m\}$. The car starts with $v = 0$, and at each time step, the car can either increase the velocity by 1, decrease it by 1, or keep it the same; this new velocity is used to advance x to the new position. The velocity is not allowed to exceed the speed limit m nor return to 0.

In addition, to prevent people from recklessly cruising down Serra Mall, the university has installed speed bumps at a subset of the n locations. The speed bumps are located at $B \subseteq \{1, \ldots, n\}$. The car is not allowed to enter, leave, or pass over a speed bump with velocity more than $k \in \{1, \ldots, m\}$. **Your goal is to arrive at position n with velocity 1 in the smallest number of time steps.**

Now let's add more information to this problem:

The university wants to remove the old speed bumps and install a single new speed bump at location $b \in \{1, \ldots, n\}$ to maximize the time it takes for the car to go from position 1 to n.

Let $T(\pi, B)$ be the time it takes to get from 1 to n if the car follows policy π if speed bumps B are present. If π violates the speed limit, define $T(\pi, B) = \infty$.

To simplify, assume $n = 6$ and $k = 1$. Again, there is exactly one speed bump. That is, $B = \{b\}$ with $b \in \{1, \ldots, n\}$.

$x = 1$ home	$x = 2$	$x = 3$	$x = 4$	$x = 5$	$x = 6$ school

Figure: The university will add a speed bump somewhere

(i) [5 points] Compute the worst case driving time, assuming you get to adapt your policy to the university's choice of speed bump location b: $\max_b \min_\pi T(\pi, \{b\})$. What values of b attain the maximum?

Solution Note that with $n = 6$, there are only two places where one can travel at a velocity of 2, from 2 to 4 or 3 to 5; in these cases, there can't be any speed bumps there. So if the speed bump is placed at $b \in \{1, 2, 5, 6\}$, the optimal policy has space to speed up to a velocity of 2 around the bump, so the total time is 4. However, if the speed bump is placed at $\boxed{b \in \{3, 4\}}$, then the optimal policy is to travel at a velocity of 1 the whole way which results in a total time of $\boxed{5}$, which is the worst case. Most common error was missing one of the cases for b. Also, there were a number of off-by-one errors

(takes only 5 units to get from 1 to 6, not 6).

(ii) [5 points] Compute the best possible time assuming that you have to choose your policy before the university chooses the speed bump: $\min_\pi \max_b T(\pi, \{b\})$. Make sure to explain your reasoning.

Solution If we choose any policy that has velocity of 2, the university can place the speed bump in the appropriate place that results in a time of ∞. Therefore, we must choose a policy that only has velocity 1, which results in a time of $\boxed{5}$. Students should not assume that the university will definitely place speed bumps at $b \in \{3, 4\}$, but it's fine to acknowledge this as a possibility in your reasoning.