CS 188
Fall 2019

Midterm Review RL Solutions

1 Odds and Ends

(a) Can all MDPs be solved using expectimax search? Justify your answer.

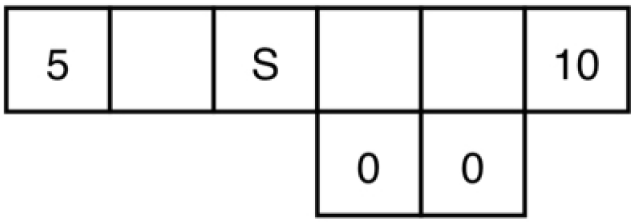
No, MDPs with self loops lead to infinite expectimax trees. Unlike search problems, this issue cannot be addressed with a graph-search variant.

(b) Why might Q-learning be superior to TD learning of values?

(a) If you use temporal difference learning on the values, it is hard to extract a policy from the learned values. Specifically, you would need to know the transition model T and reward function R. For Q-learning, the policy can be extracted directly by taking $\pi(s) = \arg\max_a Q(s, a)$.

(b) While TD learning learns values for a particular policy (online policy evaluation), Q-learning is *off-policy*: it will converge to Q-values that give you the optimal policy, even if the actions you used to explore were sub-optimal. The caveats to this are that you need to explore enough, and you need conditions on the learning rate α (has to eventually be small enough, but not decrease too fast)

Q2. MDPs and RL

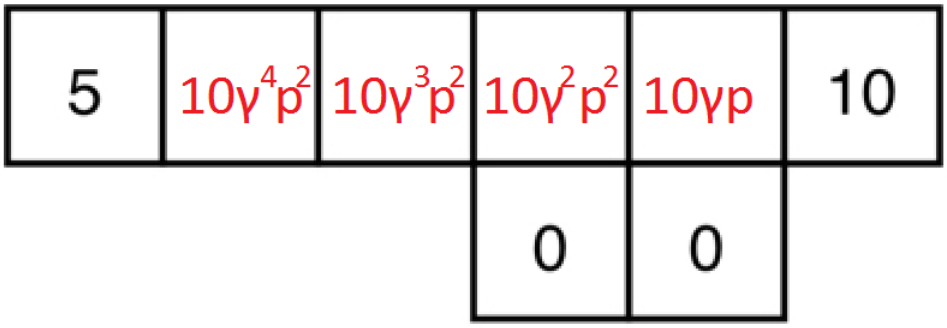


Consider the above gridworld. An agent is currently on grid cell S, and would like to collect the rewards that lie on both sides of it. If the agent is on a numbered square, its only available action is to Exit, and when it exits it gets reward equal to the number on the square. On any other (non-numbered) square, its available actions are to move East and West. Note that North and South are never available actions.

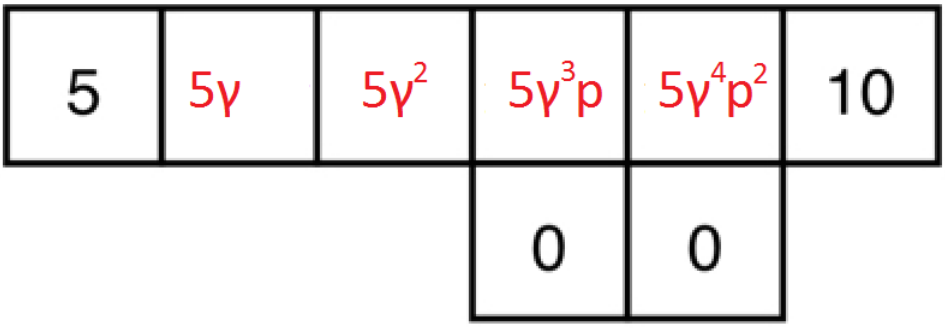
If the agent is in a square with an adjacent square downward, it does not always move successfully: when the agent is in one of these squares and takes a move action, it will only succeed with probability p. With probability $1-p$, the move action will fail and the agent will instead move downwards. If the agent is not in a square with an adjacent space below, it will always move successfully.

For parts (a) and (b), we are using discount factor $\gamma \in [0, 1]$.

(a) Consider the policy π_{East}, which is to always move East (right) when possible, and to Exit when that is the only available action. For each non-numbered state x in the diagram below, fill in $V^{\pi_{\text{East}}}(x)$ in terms of γ and p.



(b) Consider the policy π_{West}, which is to always move West (left) when possible, and to Exit when that is the only available action. For each non-numbered state x in the diagram below, fill in $V^{\pi_{\text{West}}}(x)$ in terms of γ and p.


5
5γ
5γ2
5γ3p
5γ4p2
10
0
0

(c) For what range of values of p *in terms of* γ is it optimal for the agent to go West (left) from the start state (S)?

We want $5\gamma^2 \geq 10\gamma^3 p^2$, which we can solve to get:
Range: $p \in [0, \frac{1}{\sqrt{2\gamma}}]$

(d) For what range of values of p *in terms of* γ is π_{West} the optimal policy?

We need, for each of the four cells, to have the value of that cell under π_{West} to be at least as large as π_{East}.
Intuitively, the farther east we are, the higher the value of moving east, and the lower the value of moving west (since the discount factor penalizes far-away rewards).
Thus, if moving west is the optimal policy, we want to focus our attention on the rightmost cell.
At the rightmost cell, in order for moving west to be optimal, then $V^{\pi_{\text{East}}}(s) \leq V^{\pi_{\text{West}}}(s)$, which is $10\gamma p \leq 5\gamma^4 p^2$, or $p \geq \frac{2}{\gamma^3}$.
However, since γ ranges from 0 to 1, the right side of this expression ranges from 2 to ∞, which means p (a probability, and thus bounded by 1) has no valid value.
Range: $\emptyset$

(e) For what range of values of p *in terms of* γ is π_{East} the optimal policy?

We follow the same logic as in the previous part. Specifically, we focus on the leftmost cell, where the condition for π_{East} to be the optimal policy is: $10\gamma^4 p^2 \geq 5\gamma$, which simplifies to $p \geq \frac{1}{\sqrt{2\gamma^3}}$. Combined with our bound on any probability being in the range $[0, 1]$, we get:

Range: $p \in \left[\frac{1}{\sqrt{2\gamma^3}}, 1\right]$, which could be an empty set depending on γ.

Recall that in approximate Q-learning, the Q-value is a weighted sum of features: $Q(s,a) = \sum_i w_i f_i(s,a)$. To derive a weight update equation, we first defined the loss function $L_2 = \frac{1}{2}(y - \sum_k w_k f_k(x))^2$ and found $dL_2/dw_m = -(y - \sum_k w_k f_k(x)) f_m(x)$. Our label y in this set up is $r + \gamma \max_a Q(s', a')$. Putting this all together, we derived the gradient descent update rule for w_m as $w_m \leftarrow w_m + \alpha \left(r + \gamma \max_a Q(s', a') - Q(s,a)\right) f_m(s,a)$.

In the following question, you will derive the gradient descent update rule for w_m using a different loss function:

$$L_1 = \left| y - \sum_k w_k f_k(x) \right|$$

(f) Find dL_1/dw_m. Show work to have a chance at receiving partial credit. Ignore the non-differentiable point.

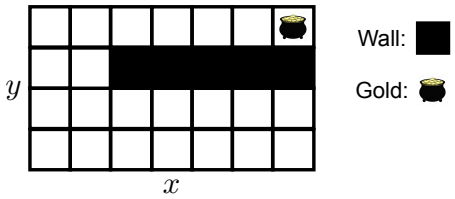
Note that the derivative of $|x|$ is -1 if $x < 0$ and 1 if $x > 0$. So for L_1, we have:

$$\frac{dL_1}{dw_m} = \begin{cases} -f_m(x) & y - \sum_k w_k f_k(x) > 0 \\ f_m(x) & y - \sum_k w_k f_k(x) < 0 \end{cases}$$

(g) Write the gradient descent update rule for w_m, using the L_1 loss function.

$$\begin{aligned} w_m &\leftarrow w_m - \alpha dL_1/dw_m \\ &\leftarrow \begin{cases} w_m + \alpha f_m(x) & y - \sum_k w_k f_k(x) > 0 \\ w_m - \alpha f_m(x) & y - \sum_k w_k f_k(x) < 0 \end{cases} \end{aligned}$$

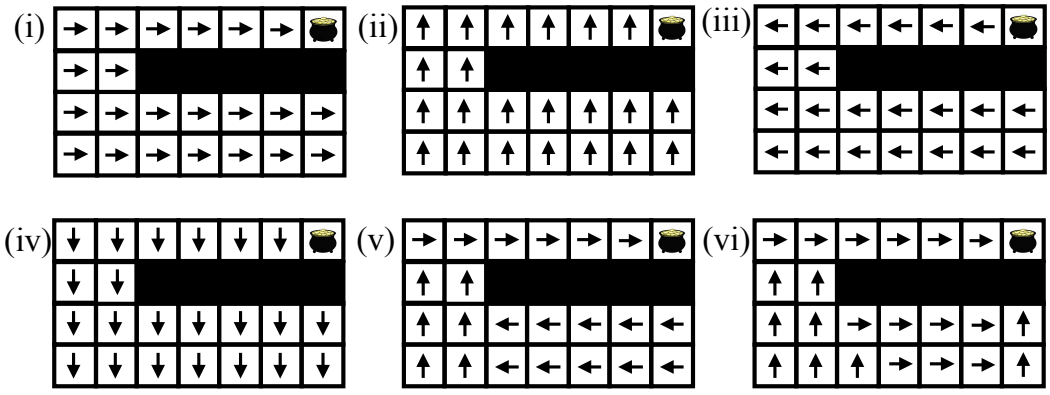
Q3. MDPs & RL



Consider the grid-world MDP above. The goal of the game is to reach the pot of gold. As soon as you land on the pot of gold you receive a reward and the game ends. Your agent can move around the grid by taking the following actions: North, South, East, West. Moving into a square that is not a wall is always successful. If you attempt to move into a grid location occupied by a wall or attempt to move off the board, you remain in your current grid location.

Our goal is to build a value function that assigns values to each grid location, but instead of keeping track of a separate number for each location, we are going to use features. Specifically, suppose we represent the value of state (x, y) (a grid location) as $V(x, y) = w^\top f(x, y)$. Here, $f(x, y)$ is a feature function that maps the grid location (x, y) to a vector of features and w is a weight vector that parameterizes our value function (note that entries in w can be any real number, positive or negative).

In the next few questions, we will look at various possible feature functions $f(x, y)$. We will think about the value functions that are representable using each set of features, and, further, think about which policies could be extracted from those value functions. Assume that when a policy is extracted from a value function, ties can be broken arbitrarily. In our definition of feature functions we will make use of the location of the pot of gold. Let the gold's location be (x^*, y^*). Keep in mind the policies (i), (ii), (iii), (iv), (v), and (vi) shown below.



(a) Suppose we use a single feature: the x-distance to the pot of gold. Specifically, suppose $f(x, y) = |x - x^*|$. Which of the policies could be extracted from a value function that is representable using this feature function? Assume the weights vector w is not allowed to be 0. Fill in all that apply.

● (i) ○ (ii) ● (iii) ○ (iv) ○ (v) ○ (vi)

(b) Suppose we use a single feature: the y-distance to the pot of gold. Specifically, suppose $f(x, y) = |y - y^*|$. Which of the policies could be extracted from a value function that is representable using this feature function? Assume the weights vector w is not allowed to be 0. Fill in all that apply.

○ (i) ● (ii) ○ (iii) ● (iv) ○ (v) ○ (vi)

(c) Suppose we use a single feature: the Manhattan distance to the pot of gold. Specifically, suppose $f(x, y) = |x - x^*| + |y - y^*|$. Which of the policies could be extracted from a value function that is representable using this feature function? Assume the weights vector w is not allowed to be 0. Fill in all that apply.

○ (i) ○ (ii) ○ (iii) ○ (iv) ○ (v) ● (vi)

(d) Suppose we use a single feature: the length of the shortest path to the pot of gold. Which of the policies could be extracted from a value function that is representable using this feature function? Assume the weights vector w is not allowed to be 0. Fill in all that apply.

○ (i) ○ (ii) ○ (iii) ○ (iv) ● (v) ○ (vi)

(e) Suppose we use two features: the x-distance to the pot of gold and the y-distance to the pot of gold. Specifically, suppose $f(x, y) = (|x - x^*|, |y - y^*|)$. Which of the policies could be extracted from a value function that is representable using this feature function? Assume the weights vector w must have at least one non-zero entry. Fill in all that apply.

● (i) ● (ii) ● (iii) ● (iv) ○ (v) ● (vi)