CS181 Practice Questions: More MDPs and Reinforcement Learning

1. **Q and V practice**

Consider an MDP with two states (A and B) and two actions (□ and △). You have the following transition and reward functions with $\gamma = 0.5$.

s	a	s'	$P(s' \mid a, s)$
A	□	A	0.5
A	□	B	0.5
A	△	A	1
A	△	B	0
B	□	A	1
B	□	B	0
B	△	A	1
B	△	B	0

s	a	$R(s, a)$
A	□	3
A	△	1
B	□	0
B	△	1

Complete the following table. (Hint: It's pretty easy to set up a spreadsheet to propagate the information for value iteration.)

k To Go	$Q(A, \square)$	$Q(A, \triangle)$	$Q(B, \square)$	$Q(B, \triangle)$	$\pi_k(A)$	$\pi_k(B)$	$V_k(A)$	$V_k(B)$
0	–	–	–	–	–	–	0	0
1	3	1	0	1	□	△	3	1
2	4	2.5	1.5	1.5	□	$\star$	4	1.5
3	4.38	3	2	1.75	□	□	4.38	2
4	4.59	3.19	2.19	2	□	□	4.59	2.19
5	4.70	3.30	2.30	2.09	□	□	4.70	2.30

2. **Stationary Policy**

In the previous problem, is the policy stationary? If not, why?

The policy is not stationary: in state B it is better to take action $\triangle$ in the last step, but with more steps to go this is not better than action $\square$.

3. **Value Iteration for Survival (MIT 6.034)**

Consider the following MDP:

Start State	Action	Probability	Next State
hungry, tired	rested	1.0	hungry, rested
	hunt	0.8	hungry, tired
		0.2	full, tired
hungry, rested	rest	1.0	hungry, rested
	hunt	0.8	full, rested
		0.2	hungry, tired
full, tired	rest	1.0	hungry, rested
	hunt	0.8	hungry, tired
		0.2	full, tired
full, rested	rest	1.0	hungry, rested
	hunt	0.8	full, tired
		0.2	hungry, tired

In this world, you care about being rested and well fed. You can either rest or hunt. Your reward function is

State	Reward
hungry, tired	0
hungry, rested	1
full, tired	1
full, rested	2

(a) Given a discounting constant γ, what is the value of the (hungry, tired) state for a policy that always rests? Give your answer in terms of γ. (Hint: use geometric series.)

(b) Assume that $Q((\text{hungry}, \text{tired}), \text{rest}) = a$ and $Q((\text{hungry}, \text{tired}), \text{hunt}) = b$ and that $b > a$. Write an expression for $Q((\text{hungry}, \text{rested}), \text{hunt})$ in terms of these quantities, γ, and $V^\star(\text{full}, \text{rested})$.

(a) Constantly resting causes one to go to (hungry, rested) and stick there. This has reward 1 at every time step, discounted by gamma, so the value is:

$$\sum_{t=1}^{\infty} \gamma^t = \frac{1}{1-\gamma} - 1$$

(b) Writing Bellman's equation:

$$Q((\text{hungry}, \text{rested}), \text{hunt}) = 1 + \gamma \times (0.8 \times V^\star(\text{full}, \text{rested}) + 0.2 \times b) \qquad (1)$$

4. **Q Learning**

Imagine that you are taking actions in an MDP that are selected according to the optimal policy. Explain whether or not Q-learning will only learn optimal Q-values.

Q-learning will learn the optimal Q-values as long as the policy explores all states. The optimal policy may not necessarily do that, however.

5. **Q Learning**

Suppose you are standing on a linear board on which you can take action L or R to walk left or right, or S to sleep. If you walk, you have probability p_a that you actually walk to the next square, where $a \in \{L, R\}$. With probability $1 - p_a$, however, your cat distracts you and you stay on the same square. Staying on a square gives you reward r_i. Your learning rate is $\alpha = 0.5$ and $\gamma = 0.5$.
You are on square 1, you choose $a = L$ and receive $r = 4$. What is your updated value of $Q(1, L)$? Assume that your initial Q-values are all zero.

All Q values are initially 0. $Q(1, L) = 0 + 0.5(4 + .5 \times 0 - 0) = 2$

6. **Q learning**

Continuing the previous problem, in the next step, you are on square 0, you choose $a = R$, receive $r = 3$ and end up in $s = 1$. What is your updated value of $Q(0, R)$?

$Q(2, W) = 0 + 0.5(3 + 0.5 \times 2 - 0) = 2$

7. **Model Based Reinforcement Learning**

How do we compute the Maximum-Likelihood estimate of the expected reward from taking action a in state s?

We take the average of the rewards received from taking action a when we are in state s. We can keep track of $N(s,a)$, the number of times that we visit state s and perform a and $R^{\text{total}}(s,a)$, the sum of the rewards received from taking action a in state s. Then the ML estimate is given by

$$\frac{R^{\text{total}}(s,a)}{N(s,a)}$$

8. **ϵ-Greedy**

Why do we generally use an ϵ-Greedy algorithm when choosing the current action during the Q-Learning algorithm? Describe how you would change the value of ϵ over time to encourage early exploration/learning and good decision making after the state space was explored.

First we recall that in the ϵ-greedy algorithm we choose the best action, via

$$\max_{a' \in A} Q(s, a'),$$

with probability $1 - \epsilon$ and a random action otherwise. We use this because it encourages exploration of the state space by introducing randomness into the choice of action at each step, which we recall is necessary for RL to work with. In the beginning, we would choose a high value of ϵ, which would encourage a lot of random exploration of the state space, and as time went on, we could slowly decrease the value of ϵ to encourage taking the action that maximizes the Q value in the current state, which we expect to be a good action given that we've explored the state space sufficiently to closely approximate the true Q function.

9. **Convergence of Q-Learning**

Suppose we have a deterministic world where each state-action pair (s, a) is visited infinitely often. Consider an interval during which every such pair is visited. Suppose that the largest error in the approximation of $\hat{Q}_n$ after n iterations is e_n, i.e.

$$e_n = \max_{s,a} |\hat{Q}_n(s, a) - Q(s, a)|$$

Show that e_{n+1} is bounded above by γe_n, where γ is the usual discount factor.

$$\begin{aligned}
e_{n+1} &= \max_{s,a} |\hat{Q}_{n+1}(s, a) - Q(s, a)| \\
&= |(r + \gamma \max_{a'} \hat{Q}_n(s', a')) - (r + \gamma \max_{a'} Q(s', a'))| \\
&= \gamma |\max_{a'} \hat{Q}_n(s', a') - \gamma \max_{a'} Q(s', a')| \\
&\leq \gamma \max_{a'} |\hat{Q}_n(s', a') - Q(s', a')| \\
&\leq \gamma \max_{s'', a'} |\hat{Q}_n(s'', a) - Q(s'', a')| \\
&= \gamma e_n
\end{aligned}$$