CS5180 – Ex2

Sudhendra Kambhamettu
NUID: 002786797

Q1.

To design the four rooms environment as an MDP, let's consider the following to establish the that the environment is indeed Markovian in nature.

Markovian principle states that – "The dynamics of the stochastic process only depends on its current state and time and it is independent of the other states it had before". In other words, "In this case, the random variables R_t and S_t have well defined discrete probability distributions dependent only on the preceding state and action" – shown in [RLeq3.2].

Let's observe the properties of the 4 rooms environment given in ex0 & define some characteristics.

1. State Space (S): The state is defined by the (x, y) coordinates of the agent within the grid where $x, y \in [0,10]$ with a defined set of walls. The agent can be present in any of the states within this grid.
2. Action space (A): There are 4 distinct actions the agent can take – Up, Down, Left & Right.
3. Transition Probability (P): There is a stochastic element to the environment where the agent "slips" 10% of the time moving in two of its perpendicular directions to its current state & 80% of the time, the agent takes the intended action. If any move leads to collisions into the walls, the state does not change & finally, the agent is teleported to the coordinates (0,0) once it reaches the goal state.
4. Reward Function (R): The agent receives a reward of +1 when it reaches the goal state and 0 in every other state in the environment.

This environment therefore satisfies the criteria of having a defined state space, action space, reward function and a transition probability which only depend on the immediately preceding (current) state & action which clearly satisfies the Markov principle where the goal is to maximize the reward it gets.

a. As described above, the state and action spaces can be formally defined as follows.

$$S = \{(x, y) \mid 0 \le x < 11, 0 \le y < 11, \text{and } (x, y) \text{ is not a wall position}\}$$
$$A = \{left: 0, down: 1, right: 2, up: 3\}$$

b. In order to compute the total number of non-zero rows i.e., the conditional probability (the dynamics function) is non-zero. Which essentially means the transitions which are not possible i.e., have a zero probability of ever occurring in any step throughout all episodes/trials should not be considered.
We can formulate a simple formula to compute the number of non-zero rows.
Let non-zero rows be N_{nz} and the number of possible transitions is T_p then,

$$N_{nz} = |S - W| \times |A| \times T_p \quad - eq1$$

Where S represents the number of possible states and W represents all the states which are walls. Now, computing each value for the above equation.

$$|S - W| = |121 - 17| = 104$$
$$|A| = 4$$
$$T_p = 3$$

For each state-action pair, there could be up to 3 (T_p) possible transitions due to the 80% chance of moving in the intended direction and 10% chances for each perpendicular direction.
By plugging these values back into the equation 1, we get –

$$\therefore N_{nz} = 104 \times 4 \times 3 = \mathbf{1248}$$

$$Plus\ one\ goal\ state = 1248 + 1 = \mathbf{1249}$$

c. Table of Probability values (transition table) is enclosed with the assignment as a separate transition_probabilities.csv file. Kindly check that for results including the ex2.py code.

Q2.

a. This question discusses the pole-balancing task as an episodic task rather than a continuous on-going task with 0 reward for all the time steps and -1 in case of failure. In this case, the return (G_t) would be the sum of all discounted future rewards. One can see that in such a setting, the return from any timestep would be a function of how far the failure occurs in the future discounted by a factor of γ.

To derive an equation for the episodic setting, let's assume that failure occurs at time step T and G_t at time t (for any $t < T$) can be modelled mathematically as follows – Generally, the equation for discounted return is:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

For an episodic setting since there is a termination state, we can modify the discounted return equation as follows –

$$G_t = -\gamma^{T-t}$$

i.e., this equation becomes absolutely finite and since the reward at any time step is 0 except for failure, the discounted return becomes a negative quantity. This clearly contrasts with the continuing formulation of the pole-balancing task, where the returns could accumulate indefinitely.

Therefore, in the episodic formulation, the return emphasizes the penalty for failure adjusted by how immediate the failure is, given the discount rate. This setup makes early failure (relative to the start of an episode) more penalizing on a discounted basis than failure later on, aiming to encourage strategies that delay failure as long as possible.

b. In formulating the maze solving problem as giving 0 rewards for all time steps & +1 for solving the maze, the reward signal essentially encourages policies which solve the maze irrespective of solving it sooner or later. Since the return is not discounted, the agent does not prioritize solving the maze sooner or more efficiently but to simply maximize the return which is always constant (+1) at the termination of an episode. Which means that there is no intermediate feedback to the agent & thus it cannot differentiate between good and bad actions in relation to the mazes' exit.
Therefore, the agent struggles to improve overall. Modifying the reward signal could help improving the agent's performance.

Q3.

a. The equation for discounted return is as follows:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} - (1)$$

Otherwise written as –

$$G_t = R_{t+1} + \gamma G_{t+1} - (2)$$

Using the values given in the question and plugging them into eq 1 for G_5 –

$$G_5 = 0 \ \because Terminal\ state$$

Using this to plug into eq 2.

$$G_4 = R_5 + \gamma G_5 = 2 + 0.5 \times 0 = 2$$
$$G_3 = R_4 + \gamma G_4 = 3 + 0.5 \times 2 = 4$$

$$G_2 = R_3 + \gamma G_3 = 6 + 0.5 \times 4 = 8$$
$$G_1 = R_2 + \gamma G_2 = 2 + 0.5 \times 8 = 6$$
$$G_0 = R_1 + \gamma G_1 = -1 + 0.5 \times 6 = 2$$

b. Here it's given that $R_1 = 2$ and all the subsequent rewards are 7.
For an infinite sequence of constant rewards, the equation for discounted rewards becomes –
$$G_t = R_{t+1} \sum_{k=0}^{\infty} \gamma^k = R_t \left(\frac{1}{1-\gamma}\right)$$
Plugging in the corresponding values into this equation, we get –
$$G_1 = 7 \times \left(\frac{1}{1-0.9}\right) = 7 \times 10 = 70$$
$$\therefore G_0 = R_1 + \gamma G_1 = 2 + 0.9 \times 70 = \mathbf{65}$$

Q4.

Initially, the agent has to choose between the two deterministic actions & it will choose so based on the expected return of choosing each state. By initial examination of the given 101 x 3 world given, it becomes evident that choosing **Down** will make the total return maximum and choosing **Up** will give a negative expected return.

This seems to be a classic example for prioritizing future rewards over immediate rewards. Close examination of the discounted reward equation makes it clear that choosing a low discount factor makes the return "myopic" in being concerned only with maximizing immediate rewards whereas a high discount factor makes the return "hyperopic" in being concerned only with maximizing the future expected rewards.

In this case, it is beneficial for us to prioritize future rewards over immediate ones, therefore choosing a higher gamma value (discount factor) will ensure the agents success in maximizing the returns.

In this case we can write the discounted return equation as follows –

$$G_t = R_{t+1} + \gamma G_{t+1}$$

Modifying this equation to –

$$G_t = R_{t+1} + \gamma R_{t+1} \sum_{k=0}^{\infty} \gamma^k$$

$\sum_{k=0}^{\infty} \gamma^k$ is a geometric progression and the sum can be equated to the following expression:

$$\sum_{k=0}^{\infty} \gamma^k = \frac{1-\gamma^{k+1}}{1-\gamma}$$

Substituting the values given in the question to derive return for Up or Down –

For Up: $50\gamma - \sum_{k=2}^{101} \gamma^k = 50\gamma - \gamma^2 \left(\frac{1-\gamma^{100}}{1-\gamma}\right)$

For Down: $-50\gamma + \sum_{k=2}^{101} \gamma^k = -50\gamma + \gamma^2 \left(\frac{1-\gamma^{100}}{1-\gamma}\right)$

Q5.

a. We know that

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

Now, let's add a constant C to the equation. We get –

$$G_t^* = G_t \sum_{k=0}^{\infty} \gamma^k C$$

$$G_t^* = G_t + \frac{C}{1-\gamma}$$

Therefore, v_c can be written as,

$$v_c = E[G_t^* | S_t = s] = E\left[G_t + \frac{C}{1-\gamma} \middle| S_t = s\right] = E[G_t | S_t = s] + \frac{C}{1-\gamma}$$

he addition of $\frac{C}{1-\gamma}$ is uniform across all states, which means the difference in value between any two states under policy π remains unchanged. Thus, the policy's decision-making, which depends on the relative values of actions, is unaffected by the addition of a constant to all rewards.

b. Adding a constant **c** to all rewards in an episodic task, such as maze running, affects the task differently compared to a continuing task. In episodic tasks, the cumulative impact of adding **c** to each reward is finite, as episodes end. This could alter the total expected reward for an episode and potentially change an agent's decisions if the added constant significantly impacts the relative value of reaching the goal sooner. For example, if **c** is positive and large enough, it might encourage exploration by making the agent less averse to taking longer paths that might lead to the goal, as intermediate rewards are increased. Conversely, in a continuing task, the effect of adding **c** uniformly distributes across all states due to discounting, preserving policy decisions.

Q6.

A. The Bellman Equation is written as follows –

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r | s, a)[r + \gamma v_\pi(s')]$$

For the center state lets substitute the values and we get –

$$v_\pi(center) = 0.25[0.9 \times 2.3] + 0.25[0.9 \times 0.7] + 0.25[0.9 \times 0.4] + 0.25[0.9 \times -0.4]$$

$$= 0.68 \approx \mathbf{0.7}$$

B. As we know, the bellman equation is written above,
The random policy will be –

$$\pi(a|s) = 0.5$$

Let's substitute the known given values into the bellman equation –

$$v_\pi(s) = 0.5 \times (0 + 0.9 \times 19.8) + 0.5 \times (0 + 0.9 \times 19.8)$$

$$v_\pi(s) = 0.9 \times 19.8 = 17.82$$

$$\therefore v_\pi(s) = \mathbf{17.82}$$

Q7.

a. Looking at the information provided, I can guess that the value functions estimates will be as follows: V(A) = 0.5, V(L) = 0, V(R) = 0

Lets verify.

Given the values, $\gamma = 1, \pi(a|s) = 0.5, p(s',r|s,a) = 1$

Substituting the values in the bellman equation we get –

$$V(A) = 0.5 \times (0 + 1 \times 0) + 0.5 \times (1 + 1 \times 0) = 0.5$$

$$\therefore \textit{Guess is consistent.}$$

b. Again, analyzing the figure given I can take the following guess
V(L) = 0, V(A) = 1/6, V(B) = 2/6, V(C) = 3/6, V(D) = 4/6, V(E) = 5/6, V(R) = 0

Given values $\gamma = 1, \pi(a|s) = 0.5$
Lets verify for state **E**

Substituting them back into bellman equation we get –

$$V(E) = 0.5 \times \left(0 + 1 \times \frac{4}{6}\right) + 0.5 \times (1 + 1 \times 0) = \frac{5}{6}$$
$$V(D) = 0.5 \times \left(0 + 1 \times \frac{3}{6}\right) + 0.5 \times \left(0 + 1 \times \frac{5}{6}\right) = \frac{4}{6}$$
$$V(C) = 0.5 \times \left(0 + 1 \times \frac{2}{6}\right) + 0.5 \times \left(0 + 1 \times \frac{4}{6}\right) = \frac{3}{6}$$
$$V(B) = 0.5 \times \left(0 + 1 \times \frac{1}{6}\right) + 0.5 \times \left(0 + 1 \times \frac{3}{6}\right) = \frac{2}{6}$$
$$V(A) = 0.5 \times (0 + 1 \times 0) + 0.5 \times \left(0 + 1 \times \frac{2}{6}\right) = \frac{1}{6}$$

$$\therefore \textit{Guess is consistent.}$$

c. We can similarly guess for an arbitrary number of states as follows:

$$V(R) = 1, V(L) = 0, V(\textit{every other state}) = \frac{1}{n-k} \ (k \textit{ is pos. after } L)$$

Q8.

a. We can expand the Bellman equation for the states high and low. We get the following expansion –

For the state **high**

$$v_{high} = \pi(wait|high)[r_{wait} + \gamma v(high)] + \pi(search|high)\alpha[r_{search} + \gamma v(high)] + \pi(search|high)(1-\alpha)[r_{search} + \gamma v(low)]$$

For the state **low**

$$v_{low} = \pi(wait|low)[r_{wait} + \gamma v(low)] + \pi(search|low)(\beta)[r_{search} + \gamma v(low)] + \pi(search|low)(1-\beta)[-3 + \gamma v(high)] + \pi(recharge|low)[\gamma v(high)]$$

b. Solving the equations for low and high states. Let's substitute the given values into the bellman equation for the state low.

$$v_{low} = 0.5 \times [3 + 0.9 \times v(low)] + 0 \times 0.6 \times [10 + 0.9 \times v(low)] + 0 \times 0.4 \times [-3 + 0.9 \times v(high)] + 0.5 \times [0.9 \times v(high)]$$

$$v_{low} = 1.5 + 0.45 \times v_{low} + 0.45 \times v_{high} - eq1$$

Using this value to solve for the value of the state high,

$$v_{high} = 0.5 \times [3 + \ 0.9 \times v(high)] + \ 1 \times 0.8 \times [10 + \ 0.9 \times v(high)] + \ 1 \times 0.2 \times [10 + \ 0.9 \times v(low)]$$

Substituting the value of low state into the above equation & solving,

$$v_{high} = 1.5 + 0.45v_{high} + 0.72v_{high} + 0.18 \times \left(1.5 + 0.9 \times v_{higlowh}\right) - eq2$$

$$solving\ eq1\ \&\ eq\ 2\ we\ get -$$

$$\therefore v_{high} = \ \mathbf{79.03}$$

And by back substituting we get –

$$v_{low} = \mathbf{67.4}$$

Hence verified.

c. We can substitute the given values into the following equation

$$v_{low} = \ \pi(wait|low)[r_{wait} + \ \gamma.v(low)] + \pi(search|low).\beta.[r_{search} + \ \gamma.v(low)] + \ \pi(search|low).(1-\beta).[-3 + \ \gamma.v(high)] + \ \pi(recharge|low)[\gamma.v(high)]$$

Given that $\pi(wait|low) = \theta$ and $\pi(recharge|low) = 1 - \theta$. We can substitute the values into the above equation to solve for θ –

$$= \ \theta[3 + \ 0.9 \times v(low)] + 0 \times 0.6 \times [10 + \ 0.9 \times v(low)] + \ 0 \times 0.4 \times [-3 + 0.9 \times v(high)] + \ (1-\theta) \times 0.9 \times v_{high}$$

By solving for v_{high} and v_{low} while iterating over θ we get optimal i.e., highest values for v_{high} & v_{low} for $\boldsymbol{\theta} = \ \mathbf{0}$.

For a $\theta = 0$ implies that the optimal policy dictates for the robot to always recharge instead of waiting in the low state.