DRAFT

Birla Institute of Technology & Science, Pilani
Work Integrated Learning Programmes Division
Second Semester 2022-2023
Mid-Semester Test (EC-2 Makeup)

Course No. : AIMLCZG512 **Course Title**:Deep Reinforcement Learning
Nature of Exam : Closed Book **Weightage** : 30% **No. of Pages** = 2 ; **No. Of Questions =** 6;
Duration : 2 Hours; **Date of Exam**: _____

Note to Students:

1. Answer all the questions.
2. Write your name and sign at the end of all the pages.
3. Assumptions made if any, should be stated clearly at the beginning of your answer.

Sample answers are provided in this key. Other ways of contextualizing would be duly considered if they are within the technical validity of the subject. Students are expected to give detailed answers pertaining to the mark distribution for each question.

NOTE: This solution is our first draft, might have some errors, corrected later. Verify if the solutions given is correct and provide correct answers if there are mistakes. Thank you

Question -1 [2 + 2 + 1 = 5 Marks]

An eCommerce platform deploys a recommendation engine that selects one of 3 brands (Adidas, Nike, Sketchers) to present to a user during each visit. The user preferences vary over time due to trends in the market. The platform receives a reward of **1** for a successful purchase and **0** otherwise.

(a) What is the most suitable (be very specific) way to model this problem using one of the approaches taught to you and why [2 Marks]?
(b) What is the most appropriate approach to compute action values? Explain. [1.5 Marks]
(c) What are the hyperparameters involved in your model? Write down their impacts in learning. [1.5 Marks]

Answer Key:

(a) The best way to model this is as a non-stationary MAB. Each brand (Adidas, Nike, Sketchers) represents an arm of the bandit. The user's preferences change over time, making the reward distribution non-stationary. The agent must learn to recommend the most promising brand based on recent rewards (clicks/purchases).
(b) Since user preferences vary, The true probabilities p(a) drift over time.
E.g. Nike might be trendy this month, Sketchers next month. This makes it a non-stationary bandit problem. Compute action values through Epsilon-greedy. Explore occasionally → try all brands. Exploit mostly → show the brand with the highest recent success rate.
(c) Epsilon, Discount factor, Number of arms(brands).

Higher epsilon Learns faster about changes in trends.
Lower epsilon more exploitation: May miss new trends if it sticks to an old favorite.

Marking Scheme :

a) 1.0 Marks for stating MAB & Non Stationary; if non-stationarity is not in the answer, award 0.5 marks only;

1.0 Marks for reasoning that includes what arms are and why the problem is non-stationary.

b) 0.5 m for the choice of algorithm. 1 m for reasoning.
c) 0.5m for each hyperparamter

Question -2 [2 + 1 + 2 = 5 Marks]

A robot delivers parcels across a grid-like town and returns to the depot. It receives +10 for successful delivery, -10 for wrong delivery, and -1 for each movement. Each episode ends when all parcels are delivered.

(a) Model the task as an episodic MDP.
(b) If using the Monte-Carlo first-visit method, define what constitutes an episode and how it updates state values.
(c) What would the return be for an episode with 3 successful deliveries in 5 steps? Compare fully discounted vs undiscounted return for the episode. State your observation.

Answer key :

a) S = robot position + parcels status
A = {North, South, East, West, PickUp, DropOff}
P = deterministic transitions in grid + parcel handling
- Moving:
  Robot moves one cell in the chosen direction (unless blocked).
- PickUp:
  If robot is at depot and parcel is waiting, → robot picks it up.
- DropOff:
  If robot is at parcel's correct delivery location, → the parcel is marked as delivered.
  If dropped at the wrong location → delivery fails.

$$R(s,a,s') = \begin{cases} +10, & \text{if delivery is successful in } s' \\ -10, & \text{if delivery is incorrect in } s' \\ -1, & \text{for any move or other step} \end{cases}$$

γ = e.g. 0.9
Episode ends once all parcels delivered

b) An **episode** = one robot delivery mission from start to finish.
First-visit MC averages returns only from the **first time a state appears** each episode.
c) 3 successful deliveries = 3 X 10 =30
Each step cost = 5 X-1 = -5
Total Undiscounted return = 25
Fully discounted return : Consider γ = 0.9, then total return would be 21.041.

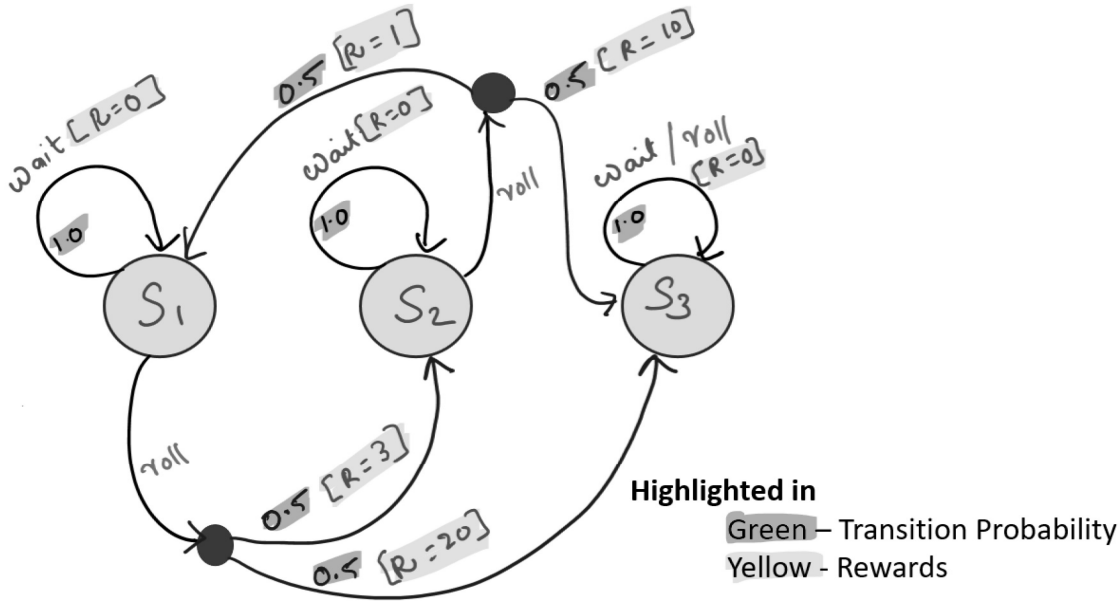
The undiscounted return counts all rewards equally → 25.
The discounted return reduces the value of future rewards → about 21.04.
Discounting prefers faster deliveries because early rewards contribute more to the total return than later ones.

Marking scheme :

a) 0.5 m for each component of MDP
b) 0.5 m for episode, 0.5 m for return calculation
c) 0.5 m each for discounted and undiscounted return calculation. 1m for comparison between both and reason.

Question -3 [3+2 = 5 Marks]

An agent plays a simplified board game with 3 states: S1, S2, and S3. The agent can take two actions at each state: `roll` or `wait`. The dynamics of this system are presented in the following drawing.



(a) Assuming the initial values of states are 0's and $\gamma = 0.5$, simulate 2 iterations of the synchronous value iteration algorithm. Show the values in a neat table. [3 Marks]
(b) Write down the policy using the results in (a) [2 Marks]

Answer:

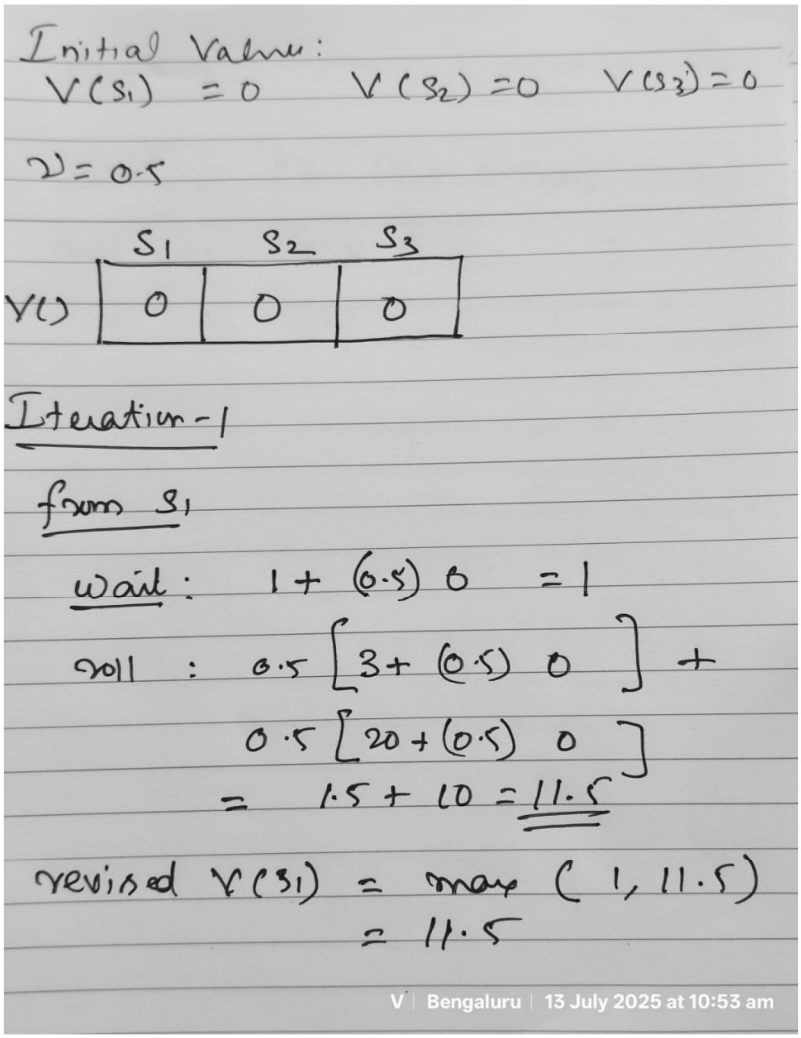
Part (a) - 0.5 x 6 entries = 3 marks.

Final Table

Iteration	V(S1)	V(S2)	V(S3)
Initial Values	0	0	0
1 (see iteration #1 calculations below)	11.5 [give 0.5 marks if computation is shown]	0.5 [give 0.5 marks if computation is shown]	0 [give 0.5 marks if computation is shown]
2(see iteration #2 calculations below)	11.625 [give 0.5 marks if computation is shown]	3.5 [give 0.5 marks if computation is shown]	0 [give 0.5 marks if computation is shown]

Sample computation for Iteration 1 - state 1

DRAFT



Part (b) - 2 marks for this anwer. In the policy table, reduce 0.5 for each mistakes. Note that for state S3, students can write the same in different ways. They can present the policy as a π function. Look into these variations.

Based on the values from iteration 2, the optimal policy π is derived by choosing the action that maximized the value at each state.

Policy:

State	Optimal Action
S1	roll
S2	roll
S3	wait or roll

Question -4 [2 + 2 + 1 = 5 Marks]

A reinforcement learning agent evaluates patient symptoms and suggests a treatment plan. A healthcare practitioner then assesses the effectiveness of the agent's recommendation based on the observed outcomes following the implementation of the treatment. It is observed that the agent recommends observation without medication 30% of the time. This approach leads to patient improvement in 70% of those cases, but worsens the patient's condition in the remaining 30%. Conversely, the agent recommends prescribing medication 70% of the time, which results in patient improvement in 60% of those cases. The practitioner provides feedback to the agent in the form of a reward: +1 when the patient's condition improves, and -1 when it deteriorates.

(a) Formulate the given problem as a Markov Decision Process (MDP). Show the model dynamics using neat transition diagram. [2 Marks]
(b) Write the Bellman optimality equation used for value estimation per state. [2 Marks]
(c) What will be the impact if the reward of 0 is designed instead of -1 on observation of deterioration in the health condition? [1 Marks]

Marking Scheme :

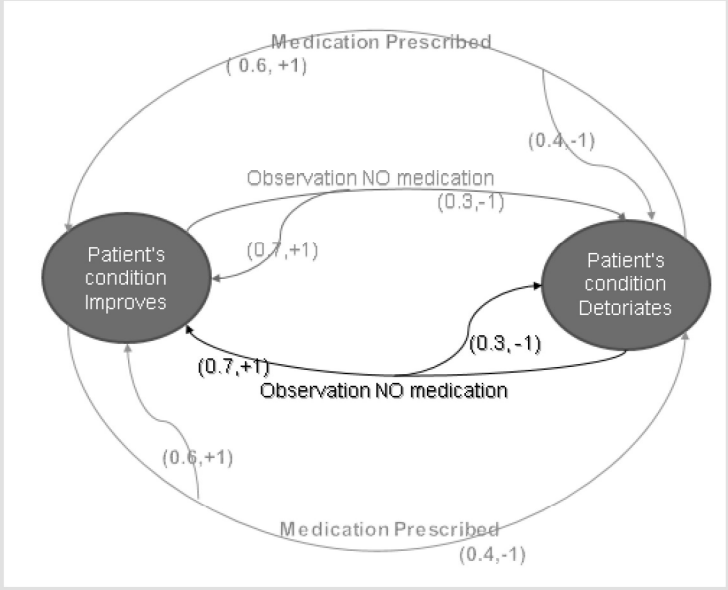
(a) 1m - State transtions & correct probability representation
1m - Correct Reward representation in all the transition
Partial marking : 0.5m - If the transitions were not correctly identified but only the states are identified

(b) 1m - Equation for "improves"
1m - Equation for "detoriates"
Partial marking = 1m : if both are incorrect . eg., instead of MAX if the "SUM" is used.

c) 1m - As per the answer key. NO partial marks for incorrect answer.

Answer Key:

a) P(Medication | State) = 0.7, P(No Medication | State) = 0.3



b)

$$\mathbf{v}_*(Improves) = MAX\begin{Bmatrix} (p(improves|improves, medication)[r(improves, medication, improves) + \gamma \mathbf{v}_*(improves)] + p(detoriates|improves, medication)[r(improves, medication, detoriates) + \gamma \mathbf{v}_*(detoriates)]), \\ (p(improves|improves, NOMedication)[r(improves, NOMedication, improves) + \gamma \mathbf{v}_*(improves)] + p(detoriates|improves, NOMedication)[r(improves, NOMedication, detoriates) + \gamma \mathbf{v}_*(detoriates)]), \end{Bmatrix}$$

$$= MAX\begin{Bmatrix} (0.6[1 + \gamma \mathbf{v}_*(improves)] + (0.4[-1 + \gamma \mathbf{v}_*(detoriates)], \\ (0.7[1 + \gamma \mathbf{v}_*(improves)] + (0.3[-1 + \gamma \mathbf{v}_*(detoriates)] \end{Bmatrix}$$

Same as that of above

$$\mathbf{v}_*(detoriates) = MAX\begin{Bmatrix} (p(improves|detoriates, medication)[r(detoriates, medication, improves) + \gamma \mathbf{v}_*(improves)] + p(detoriates|detoriates, medication)[r(detoriates, medication, detoriates) + \gamma \mathbf{v}_*(detoriates)]), \\ (p(improves|detoriates, NOMedication)[r(detoriates, NOMedication, improves) + \gamma \mathbf{v}_*(improves)] + p(detoriates|detoriates, NOMedication)[r(detoriates, NOMedication, detoriates) + \gamma \mathbf{v}_*(detoriates)]), \end{Bmatrix}$$

$$= MAX\begin{Bmatrix} (0.6[1 + \gamma \mathbf{v}_*(improves)] + (0.4[-1 + \gamma \mathbf{v}_*(detoriates)], \\ (0.7[1 + \gamma \mathbf{v}_*(improves)] + (0.3[-1 + \gamma \mathbf{v}_*(detoriates)] \end{Bmatrix}$$

c) Instead of "-1" is "0" reward is designed then the agent is not penalized when its recommendation worsens the patient's condition which is fatal in the medical domain.

Question -5 [2 +1 +2 = 5 Marks]

In the healthcare domain, a patient aims to predict whether a healthcare professional will recommend laboratory tests (yes = 1, No = 2) during each hospital visit, regardless of the patient's health condition. The professional's feedback rating for each visit is recorded as shown below.

Visit	Lab Test Recommended?	Feedback Rating
1	1	1
2	2	5
3	2	1
4	2	5
5	1	10
6	1	5
7	1	1
8	2	5

(a) Predict whether a lab test will be recommended or not during the 9th visit, framing this as a Multi-Armed Bandit (MAB) problem with UCB parameter C = 2. [2 Marks]

(b) How can more exploration be incorporated into the UCB? State the changes in a). [1 Marks]

(c) In the context of the tabulated observations, if ε-greedy action selection were used with ε = 0.7 instead of UCB, specify the time steps that correspond to actions chosen through purely random exploration. [2 Marks]

Answer Key :

a)

DRAFT

Total number of visits (t=8)	Last Time period visited Nt(a)	Lab test recommended?	Average of Reward	UCB Value after t=8
4	7	Yes = 1	(1+10+5+1)/ 4 = 4.25	4.45 + 2 = 5.69
4	8	No = 2	(5+5+5+1)/4 = 4	4 + 2 = 5.44

b) Increase the control parameter C and /or Using the results of UCB values as input Q-values, E-greedy based explorations can be set up.

c)

Timesteps	Your Answer	Explanation
1	Definetly Random	Initial selection. All Q(a) are same. Not reward maximizing
2	Definetly Random	Random. Not reward maximizing because the greedy action is "1"
3	Possibly Random	Could be random as action "2" might have been selected either in the random selection step or greedy selection step algorithm
4	Possibly Random	Could be random as action "2" might have been selected either in the random selection step or greedy selection step algorithm
5	Definetly Random	Random. Not reward maximizing because the greedy action is "2"
6	Possibly Random	This is a greedy action selection. But this could be have also been selected from a random choice. Hence this is possibly random
7	Possibly Random	This is a greedy action selection. But this could be have also been selected from a random choice. Hence this is possibly random
8	Definetly Random	Random. Not reward maximizing because the greedy action is "1"

Marking Scheme:

a) **1m - UCB value is same for both actions ie., 2*0.72=1.44**
   0.5m - Average values of actions to find Q(action) before adding the UCB part
   0.5m - Correct choice ie., Max(5.69,5.44) correspond to action "Yes=1" Lab test recommended
   Partial marking = 0.5m - if none of the above answer is correct but UCB was tried by the student
b) **1m - Any one valid answer from the answer key**
c) **1m - Correct answers for all the time steps**
   1m - For short justification for at least one case of "Definitely random" and once case of "possibly random" or for the first three time steps
   Partial Marking = 0.5 m - If complete answer is not attempted or incorrect but atleast four time steps are tried

Question -6 [2 +2 +1 = 5 Marks]

(a) Write the pseudocode for policy iteration. [2 Marks]
(b) Explain in what conditions policy iteration is more efficient than value iteration. [2 Marks]
(c) Comment on the convergence guarantee.[1 Marks]

Answer key :

a)

Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Loop:
      $\Delta \leftarrow 0$
      Loop for each $s \in \mathcal{S}$:
         $v \leftarrow V(s)$
         $V(s) \leftarrow \sum_{s',r} p(s', r|s, \pi(s))\big[r + \gamma V(s')\big]$
         $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
      old-action $\leftarrow \pi(s)$
      $\pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s', r|s, a)\big[r + \gamma V(s')\big]$
      If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

b) 1. Smaller State Spaces

- In small or moderate-sized problems, PI converges in fewer iterations because it makes big leaps toward the optimal policy.

2. Policies Stabilize Quickly

- If the optimal policy is found after just a few policy improvements, PI can be extremely fast because:
  - It evaluates a policy precisely.
  - Once the policy stops changing, we're done.

3. Low Discount Factor (γ is small)

- With smaller γ, values stabilize faster during policy evaluation.
- So policy evaluation steps require fewer sweeps, making PI very efficient.

4. When Exact Policy Evaluation is Cheap

- If solving the policy evaluation step (e.g. by solving linear equations) is computationally cheap, PI may outperform VI.
- E.g. small matrices, sparse transitions.

c) Policy Iteration always converges to an optimal policy and the optimal value function for any finite MDP in a finite number of iterations. There are only |A|^|S| possible deterministic policies.

Marking Scheme :

a) 1m - Policy initialization and evaluation

1m - policy improvement

b) 1m each for 2 points. Policy stabilizes quickly is a mandatory answer. The other 1 point can be any of the three given.

c) 0.5 m - yes converges

0.5 m - finite MDP and finite iterations