THIS PIECE OF STUDY MATERIAL HAS BEEN BROUGHT TO YOU BY

MUSTUDENTS UNITED

contributed by - Iram Mohd Ahmed Shaikh

of college - M.H Saboo Siddik College

Anjuman - I -Islam's

M.H SABOO SIDDIK COLLEGE OF ENGINEERING

8, Saboo Siddik Polytechnic Rd., Byculla, Mumbai – 400 008.

Department of Computer Science & Engineering (AI & ML)

(2024-2025)

For Assignment [5 Marks]

Class : CSE(AIML)-BE
Subject Name : RL
Subject Code : CSDOL8013

Assignment No:	1
Title:	Module 1,2,3
Date of Performance:	15/3/25
Date of Submission:	22/3/25
Roll No:	211714
Name of the Student:	Iram Mohd Ahmed Shaikh.

Evaluation:

Sr. No	Rubric	Marks
1	On time Submission & Completion (2)	02
2	Technical Content (2)	02
3	Presentation & Organization (1)	01
4	Total (5)	05

22/3/25

Signature of the Teacher:

Date:

5, 14, 23, 32, 41

RL (1)

1 Explain the concept of agent, environment, state, action, reward and policy in reinforcement learning.

1 Agent : The agent is the decision-making or learner in the RL system. It interact with the environment takes actions, and learns from the feedback (rewards or penalities) to improve its performance.

Example – A self-driving car, a chess-playing AI or a Robotic arm.

2 Environment : The environment is everything that the agent interact with. It respond to the agent's action & determine the new state and reward.

Example - For a self driving car, the environment includes the road, traffic and pedestrains.

In video game, the game world acts as the environment.

3 State (s) : A state is a representation of the current situation in the environment. It contains all the necessary information that the agent needs to make a decision.

Example -

In a chess game, the board configuration is the State

In self driving car, the state include the car's position, speed and, nearby obstacles.

Action (a) : An action is a move or decision taken by the agent that affects the environment. The set of possible actions depends on the problems.

Example - In chess, an action is moving a piece.

In a self-driving car, an action could be turning left, right or accelerating

Reward (r) : A reward is a numerical value that provides feedback on the agent's action. The goal of the agent is to maximize the cumulative reward over time.

Example - In a chess game, winning a match gives a high reward, while losing gives a negative reward.

In a self-driving car, avoiding a collision may give a reward, while crashing result in a penalty.

Policy (π) : A policy defines the agent's behavior by mapping state to actions. It guides the agent in selecting actions to maximize future rewards. Policies can be deterministic & Stochastic.

Example - In chess, a policy might suggest moving the queen to maximize board control.

In a robotic arm, a policy determines how to grip an object

Q2 What is Q-Learning in reinforcement learning?

Q-Learning is a model-free, off-policy reinforcement learning algorithm used to find the optimal action-selection policy. It helps an agent learn the best action to take in a given state by updating Q-values based on experience.

$$Q(s,a) = Q(s,a) + \alpha[r + \gamma \max Q(s',a') - Q(s,a)]$$

$Q(s,a)$: Current Q-value for state s and action a.

α (Learning Rate) : Control how much new information update the Q-value ($0 < \alpha \leq 1$).

r (Reward) : Immediate reward received after taking action a is state s.

γ (Discount Factor) : Determines the importance of future reward ($0 \leq \gamma \leq 1$).

$\max Q(s',a')$: Maximum Q-value of the next state s', assuming the best future action.

Advantages :

1. work well for discrete state-action space
2. Can handle delayed rewards by maximizing long-term rewards
3. Does not require a model for the environment.

Application of Q-Learning

Game AI
Robotics
Autonomous
Finance.

Q3 Describe different action-value estimation methods.

1 Greedy Action Selection Method.

It is a straightforward strategy where an agent always chooses the action that currently has the highest estimated value, meaning the action that is believed to give the best reward based on previous experience.

2 ε-Greedy Action Selection Method

It is an extension of the greedy action selection strategy. It introduce an element of random exploration by selecting a random action with a small probability ε and selecting the action with the highest estimated value (greedy choice) with a probability of 1-ε.

How it works :

Most of the time - you pick the best option you know (with probability 1 - ε1).

Occassionally, you pick a random option to check if there's something better (with probability ε).

ε is a small number, like 0.1 (or 10%). So, 90% of the time you pick the best option, & 10% of the time you pick randomly.

3 Upper Confidence Bound Action Selection Method

It is more sophisticated approach than ε-greedy for balancing exploration & exploitation in decision making problems, like the multi-armed bandit problem.

How it works :

The key idea behind UCB is that uncertainity plays a crucial role in choosing which option (or arm) to explore next. When you're uncertain about how good a particular option is, you should explore it more; As you gather more information, the uncertainity decreases, and you can start exploiting the options that perform well.

What is the Upper-Confidence-Bound (UCB) action selection in RL?

UCB is a Strategy ~~action selection~~ in Reinforcement Learning used in the multi-armed bandit problem to ~~create~~ balance exploration & exploitation.

It select action based on:

$$Q(a) + c\sqrt{\frac{\ln t}{N(a)}}$$

where

$Q(a)$ = estimated action value

c = exploration factor

$N(a)$ = number of times action a has been chosen

t = total number of actions taken.

UCB ensures Exploration of less-sampled actions while favouring high-reward actions.

UCB ensures it reduced randomness compared to ε|epsilon ε-greedy methods

Advantage:

1. No need to set a learning rate.
2. Naturally balances exploration vs exploitation.
3. Theoretical guarantees.

Limitations :

1. Assumes rewards are bounded and stochastic.
2. Require maintaining counts and averages for each action
3. Not easily Scalable to complex environment.

Q5 Explain the concept of Goals and Rewards in the Agent-Environment Interface. How do they influence the behaviour of an agent?

a In the agent-environment interface of reinforcement learning, goals and rewards are fundamental concept that drive the learning and decision-making process of an agent. A goal represent the desired outcome or objective that the agent aims to achieve through its interaction with environment.

b The reward serve as feedback, guiding the agent to understand which actions are beneficial and which are not.

c Over the time agent learns to choose action that maximize the cumulative rewards, which measure a long-term success.

d In this way, goal define what the agent should achieve & reward influence the agent's behavior by reinforcing actions that lead to accomplishment of those goals.

e This dynamic interaction helps the agent develop an optimal strategy or policy for achieving the best possible outcome in a given environment.

f Example:

a] In chess, the goal is to win game, not just capture pieces (sub-goals).

b] A robotic arm should prioritige precision over just speed.

c] Self Driving Car goal is to reach the destination safely and quickly.

d] Robot vaccum cleaner goal is to clean an entire room.

e] Game playing agent is to win the game.

f] warehouse Robot goal is to pick & deliver packages efficiently

g] Healthcare Treatment Agent goal is to improve patients health outcomes.

Why are Optimal Value Function important in RL, and how they are different from regular value Function.

Optimal Value Function are crucial in reinforcement learning because they represent best possible performance an agent can achieve from any given state or state-action pair, assuming it follows the optimal policy thereafter. These function serve as a benchmark for evaluating other policies.

Unlike regular value function, which estimate the expected return a specific policy π, optimal value function ~~for state and~~ Q^* represent the maximum expected return achievable from a state or action, over all possible policies.

According to the presentation, optimal value function satisfy the Bellman Optimality Equations, which are non-linear equation involving maximization over possible actions. Solving these equation is crucial central to finding optimal policies. While regular value function depend on the current policy, optimal value functions are policy-independent, they define the goal of learning in RL.

Regular Value Function : Value under a specific policy $\pi \rightarrow V^{\pi}(s), Q^{\pi}(s,a)$.

Optimal Value Function : Best possible value across ~~value function~~ all policies $\rightarrow$ $V^*(s), Q^*(s,a)$.

Q7 Consider the Markov Decision Process (MDP) shown in figure 1, with discount factor $\gamma = 0.5$. Upper case letter A, B, C represent ~~action~~ states; arcs represent state transitions; lower case letters ab, ba, bc, ca, cb represent actions; signed integers represent rewards; and fraction represent transition probabilities. Consider the ~~random~~ uniform random policy $\pi(s,a)$ that takes all actions from state 's' with equal probability. The initial value function ~~$V3(s)$~~ of $V1(A) = V1(B) = V1(C) = 2$. Apply two iteration to compute a new value function $V3(s)$.

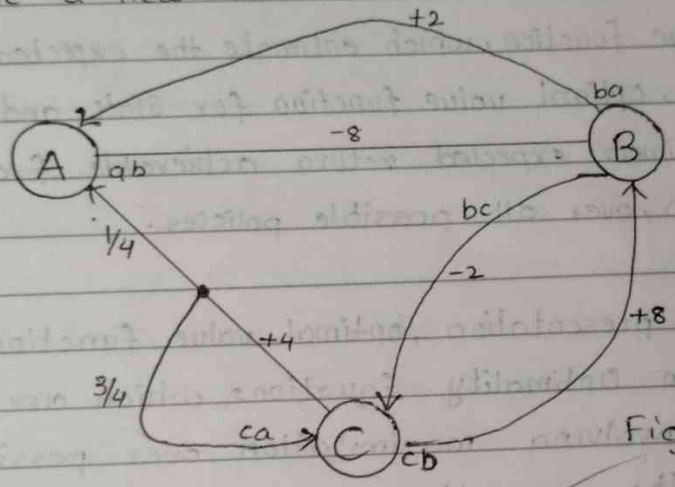
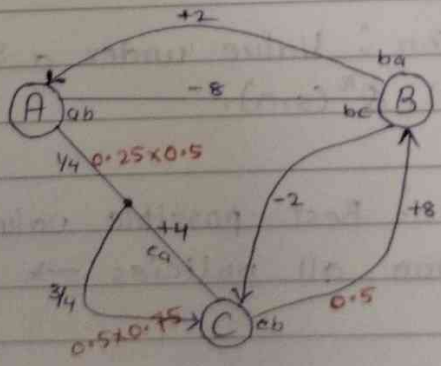


Fig 1

Sol: $V1(A) = V1(B) = V1(C) = 2$

$$V^{\pi}(s) = \sum_{a} \pi(s,a) \sum_{s'} P^{a}_{ss'} \left[R^{a}_{ss'} + \gamma V^{\pi}(s') \right]$$



Iteration 1 :

$V_2(A) = 1\,[-8 + (0.5)\,V_1(B)]$

$= [-8 + 0.5 \times 2]$

$= -8 + 0.5 \times 2$

$= -7$

$V_2(B) = 0.5\,[2 + (0.5)\,V_1(A)] + 0.5\,[-2 + (0.5)\,V_1(C)]$

$= 0.5\,[2+1] + 0.5\,[-2+1]$

$= 0.5 \times 3 + 0.5\,[-1]$

$= 1.5 + (-0.5)$

$= 1$

$V_2(C) = 0.125\,[4 + 0.5\,V_1(A)] + 0.5\,[8 + 0.5\,V_1(B)] + 0.375\,[4 + 0.5\,V_1(C)]$

$= 0.125\,[4+1] + 0.5\,[8+1] + 0.375\,[4+1]$

$= 0.125\,[5] + 0.5\,[9] + 0.375\,[5]$

$= 0.625 + 4.5 + 1.875$

$= 7$

Iteration 2 :

$V(S) = P_{ss}^{a}\,[R_{t+1} + \gamma\, V(S')]$

$V_3(A) = 1\,[-8 + 0.5\,V_2(B)]$

$= 1\,[-8 + 0.5\,(1)]$

$= -7.5$

$$V_3(B) = 0.5\,[2 + (0.5)V_2(A)] + 0.5\,[-2 + (0.5)V_2(C)]$$
$$= 0.5\,[2 + (0.5)(-7)] + 0.5\,[-2 + (0.5)(7)]$$
$$= 0.5\,[2 + (-3.5)] + 0.5\,[-2 + 3.5]$$
$$= 0.5\,[-1.5] + 0.5\,[1.5]$$
$$= -0.75 + 0.75$$
$$= 0$$

$$V_3(C) = 0.125\,[4 + 0.5\,V_2(A)] + 0.5\,[8 + 0.5\,V_2(B)] + 0.375\,[4 + 0.5\,V_1(C)]$$
$$= 0.125\,[4 + 0.5(-7)] + 0.5\,[8 + 0.5(1)] + 0.375\,[4 + 0.5(7)]$$
$$= 0.125\,[4 + (-3.5)] + 0.5\,[8 + 0.5] + 0.375\,[4 + 3.5]$$
$$= 0.125\,[0.5] + 0.5\,[8.5] + 0.375\,[7.5]$$
$$= 0.0625 + 4.25 + 2.8125$$
$$= 7.125$$

Anjuman - I -Islam's

M.H SABOO SIDDIK COLLEGE OF ENGINEERING

8, Saboo Siddik Polytechnic Rd., Byculla, Mumbai – 400 008.

Department of Computer Science & Engineering (AI & ML)

(2024-2025)

For Assignment [5 Marks]

Class : CSE(AIML)
Subject Name : RL
Subject Code :

Assignment No:	2
Title:	Module 4,5,6
Date of Performance:	15/4/25
Date of Submission:	
Roll No:	211714
Name of the Student:	Iram Mohd Ahmed Shaikh

Evaluation:

Sr. No	Rubric	Marks
1	On time Submission & Completion (2)	02
2	Technical Content (2)	02
3	Presentation & Organization (1)	01
4	Total (5)	05

Signature of the Teacher:

Date:

5, 14, 23, 32, 52

Assign-2 RL (1)

Q1 Explain key concept involved in Monte Carlo Prediction.

1 Generate Episodes

The agent follows a given policy (deterministic or stochastic) to interact with the environment and generate complete episodes (from start to terminal state), recording states, actions and rewards.

2 Sample Episodes Returns.

For each state encountered during the episode calculate the return G_t, which is the total-discounted sum of rewards from that that state to the end of the episode.

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots + \gamma^{T-t-1} R_T$$

3 Estimate Value Function

Update the value function $V(s)$ by averaging the returns observed for that state across multiple episodes. This can be done using:

First-visit MC: Use only the first occurrence of a state in each episode.

Every visit MC: Use all occurrence of the state.

4 Repeat Over Many Episodes

Continue generating episodes & updating estimates until the value function converges to a stable approximation of the true value function.

So, basically Monte Carlo prediction is a model free method that relies on complete episodes to estimate value functions based on actual returns, making it effective in environment with unknown dynamics. It is particularly useful when learning from experience with needing to know transition probabilities.

Q2 Describe the update rule used in TD prediction.

Temporal Difference prediction involves adjusting the value estimate of a state based on the observed reward and the estimated value of the next state. This allows learning to occur directly from raw experience, without needing a model of the environment.

Temporal Difference Updated Rule -

1 Temporal Difference Error :

At each time step, the TD prediction algorithm calculates a TD error, which represents the gap between the current prediction and the improved estimate.

δt is computed as :

$$\delta = R_{t+1} + \gamma \cdot V(S_{t+1}) - V(S_t)$$

R_{t+1} - Reward received after transitioning from state S_t to S_{t+1}.

γ - discount factor. $(0 < \gamma \leq 1)$ which determines the importance of future reward.

$V(S_t)$ - Current value estimate for the current state.

$V(S_{t+1})$ - Value estimate for the next state.

2 Update Equation :

Once the TD error is computed, the value of the current state is updated using :

$$V(S_t) \leftarrow V(S_t) + \alpha \cdot \delta$$

α : Learning rate - controls how much new information overrides the old.

3 Effect of Learning Rate (α)

A high learning rate (α close to 1) causes fast learning but may lead to instability.

A low learning rate provides more stable updates but require more data to converge.

4 Convergence :

Under appropriate condition (e.g the environment follows the Markov property), TD(0) prediction converges to the true value function over time.

That means the estimated values will become more accurate as more transitions are observed and processed.

Assign 2 RL (3)

Q3 Discuss a case study where dynamic channel allocation technique have been employed to improve the performance of wireless network.

Case Study : Dynamic Channel Allocation in IEEE 802.11 Wireless LANs.

Background : Wireless Local Area Network (WLANs), especially those based on the IEEE 802.11 standard (Wi-Fi), often suffer from performance issues in high-density environment (eg. campuses, offices, malls) due to co-channel interference and limited spectrum availability. Traditional Static channel assignment leads to suboptimal performance because it cannot adapt to varying traffic loads and interference levels.

Problem : In a university campus scenerio, multiple Access Points (APs) were deployed across different department. However, due to overlapping channels and fixed allocations, student & staff experienced.

High packet loss
Increased latency
Poor througput during peak usage.

Solution : Implementation of Dynamic Channel Allocation (DCA) -

The network administration implemented a DCA algorithm that :

Continuously monitored network conditions (eg. signal to-noise ratio, interference).

Automatically reassigned channels to APs based on current network load and interference patterns.

Prioritized critical application (eg. online exam, video lecture) by dynamically reducing congestion on their channels.

Technique Used :

A centralized DCA algorithm was employed which used real-time measurement from all APs &:

- Detected congested or noisy channels.
- Reallocated APs to less congested frequencies.
- Balanced user load across APs and channels.

Results :

After deploying DCA :

- Throughout improved by 40%.
- Packet loss reduced by 25%.
- User experience became more consistent, especially during peak times.
- The network adapted better to dynamic usage pattern (eg. class changeovers, events).

Conclusion : This case study demonstrate how dynamic channel allocations technique can significantly enhance wireless network performance in real-world environment. By adapting to real-time condition, DCA provides better spectrum utilization, lower interference & improved Quality of Service (QoS) for users.

Assign 2 RK (4)

Q4 Compare and contrast the challenges faced in elevator dispatching dynamic channel allocation & job-shop scheduling applications.

Aspect	Elevator Dispatching	Dynamic Channel Allocation	Job-shop Scheduling.
Key Challenges	Managing unpredictable, human traffic.	Handling signal interference & dynamic usage.	Allocation jobs to machine with multiple Constraints.
Environment	Real-time, physical world with user behavior	Wireless communication system with fluctuating, demand.	Industrial, discrete system with job-specific rules.
objectives	Minimize wait/ travel time, save energy	Maximize bandwidth, reduce interference.	Minimize completion time, idle time & delays.
Constraints.	Limited number of elevators, floor requests.	Limited frequency bands, legal regulations.	Machine capacity, job sequence
Common Challenges	Multi-objective optimization	Need for real-time adaptability	Combinational complexity.