UCS2424 PRINCIPLES OF REINFORCEMENT LEARNING

Unit - I

Introduction

Reinforcement learning - Examples - Elements of Reinforcement Learning - Tic Tac Toe - Limitations and Scope - Multi-armed Bandits; Finite Markov Decision Processes

* Reinforcement Learning?

→ a method in which an agent learns to take actions in an environment to maximize a reward signal.

→ The learner is not told what actions to take, but instead must discover which actions yield the most reward by trying them.

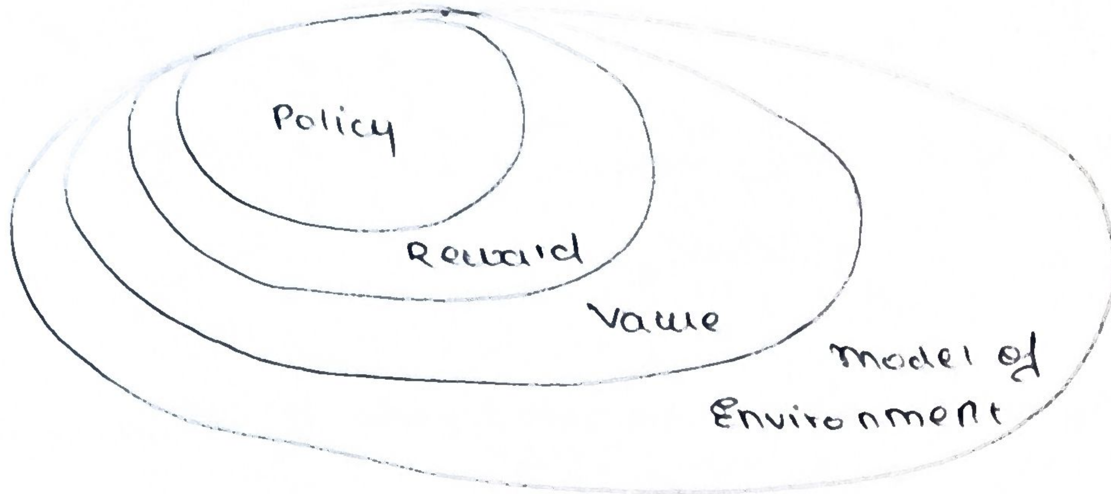
→ Two important characters - (i) trial & error
(ii) delayed reward

* Features of RL

1. Exploration vs. Exploitation - agent must balance between exploiting known actions for rewards and exploring new actions to improve future decisions

2. Goal - Directed Agent - RL agents are goal-driven, interact with uncertain environments, and adjust based on experiences.

* Elements of RL



Policy: what to do

Reward: what is good

Value: what is good because it predicts reward

Model: what values what

* Limitations of RL

1. Complexity and Computation

→ RL is well suited for complex problems, not simple ones.

→ requires substantial computational resources due to its trial-and-error learning process

→ can have high maintenance cost

2. Data Dependency

→ needs a significant amount of data to learn effectively

→ no labelled data, RL creates its own data through interactions with the environment

3. Reward Function Quality

→ success of RL heavily depends on the quality of the reward function

→ Designing an appropriate reward function can be challenging

4. Balancing Exploration vs. Exploitation

→ Striking the right balance between exploration and exploitation is difficult.

→ Too much exploration = inefficiency

Too much exploitation = may cause agent to miss better options

* Scope of RL

1. Gaming
2. Robotics
3. Finance
4. Health care
5. NLP
6. Energy Management

* TicTacToe

* Games vs. Search Problems

(i) Unpredictability - In games, opponents are unpredictable, requiring strategies for all possible moves.

(ii) Time Constraints - Limited time restricts search depth, so approximations are often needed

* Game Playing Strategy

1. Maximize winning possibility assuming that opponent will try to minimize (Minimax Algorithm)

2. Ignore the unwanted portion of the search tree (α-β Pruning)

3. Use an evaluation (utility) function to measure the winning possibility of the player.

★ <u>Multi - armed Bandit Problem</u>

→ choose btwn multiple options (arms) over a series of trials to maximize its rewards

→ depicts exploration vs. exploitation

exploration ↓ try new arms.

exploitation ↓ exploit known arms

<u>Problem Setup</u>

K : Total no. of arms

T : no. of trials

A_t : arm selected at time t

R_t : corresponding reward

μ_k : expected reward

<u>Objective</u>: maximize cumulative reward

$$\sum_{t=1}^{T} R_t$$

write about exploration vs. exploitation

<u>Expected Reward and Regret</u> - regret = diff. between reward we would have received if we had always chosen the best arm vs. the actual reward

$$\text{Regret} = \underbrace{T \cdot \max_{t=1,2..K} \mu_k}_{\text{exp}} - \underbrace{\sum_{t=1}^{T} R_t}_{\text{actual}}$$

* Approaches to solving MAB Problem

ϵ-greedy

Upper confidence bound (UCB)

$\boxed{\epsilon\text{-greedy}}$

explore → ϵ

exploit → $1-\epsilon$

$$A_t = \begin{cases} \text{random arm} & \epsilon \\ \arg\max \mu_k & 1-\epsilon \end{cases}$$

$\boxed{\text{UCB}}$

→ construct an upper confidence interval around the estimated reward.

→ select arm with highest UB

→ upper confidence bound is calculated as:

$$UCB_k = \underbrace{\hat{\mu}_k}_{\text{find argmax}} + c\sqrt{\frac{\ln t}{\underbrace{n_k}_{\text{no. of times arm has been pulled}}}}$$

→ exhibits sub-linear regret

i.e regret ↓ as T ↑

⇒ converges well

* Markov Decision Process (MDP)

MDP- mathematically idealised form of RL problems

		All states observable?	
		Yes	No
Action	No	chain	HMM
	Yes	MDP	Part obser MDP

Def

Markov-chain - give transition probabilities

$$P_{xx'} = P[X_{u+1} = x' \mid X_k = x]$$

go to x' given that $X_u = x$ now

Chain

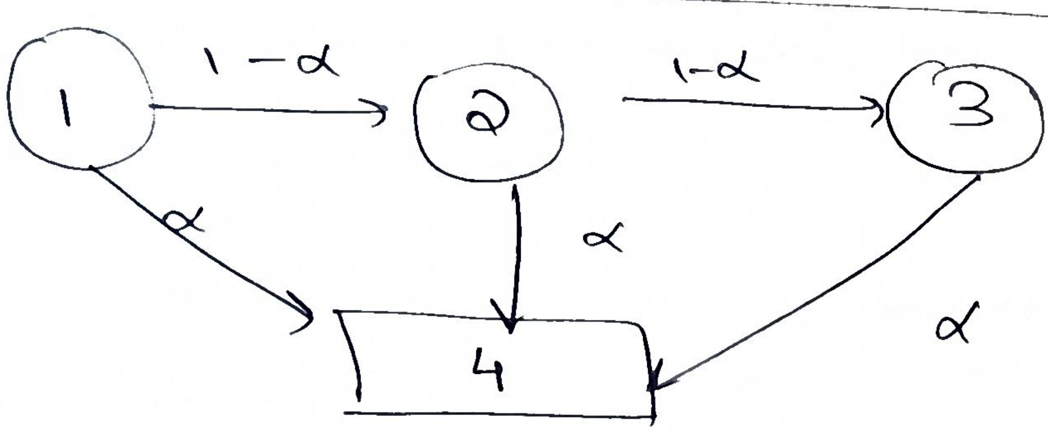
can be represented as state transition probabilities

$$P_{xx'} = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1n} \\ p_{21} & & & \\ \vdots & & & \\ p_{n1} & & & p_{nn} \end{bmatrix}$$

Matrix

Example of Markov chain as figure

Diagram



$$\begin{array}{c|c} & \begin{matrix} 1 & 2 & 3 & 4 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \end{matrix} & \begin{bmatrix} 0 & 1-\alpha & 0 & \alpha \\ & \vdots & & \\ & \vdots & & \\ & & & \end{bmatrix} \end{array}$$

Markov Reward Process (MRP)

→ ads rewards n extending Markov chain

→ Reward R_{k+1} only depends on state X_k

Reward

$$(X, P, R, \gamma)$$

reward

↳ discount factor

Return

→ discounted reward from state k onwards

$$G_k = R_{k+1} + \gamma R_{k+2} + \gamma^2 R_{k+3} \dots$$

Return

Value Function in MRP

→ expected return from being in state x_k

$$v(x_k) = E[G_k \mid x = x_k]$$

Value Function