Lecture 1 RL Overview

Simple Task ⟶ Complex Task

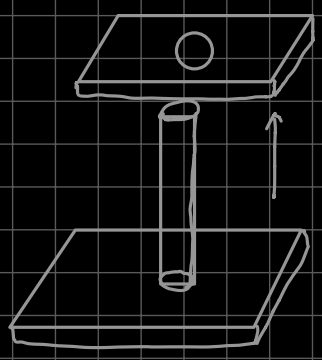
↑ ↑

Reward design ⟶ Curriculum learning

(举例：reward design 已经不能让 Agent 去完成任务. e.g: robot arms 将一带孔的板子套入直杆中)

(e.g. ① 先让 robot arm 从杆中取出板

② 让 robot arm 按照反向操作将板放回，记录此步骤作为 ground truth. 此后设计 reward 可参考 predicted position 与 ground truth 之间 distance.



Lecture 2 Markov Decision Process

Concepts && Definition:

Framework:

Markov Process → Markov Reward Process →

Markov Decision Process → Optimal Value Function

Markov Property:

- $P[S_{t+1} | S_t] = P[S_{t+1} | S_1, \dots, S_t]$
- The state captures all relevant information from the history.

Markov Process:

$\langle S, P \rangle$, S is a finite set of states ($S_1, S_2 \dots$), each state has Markov Property.

P is a state transition prob matrix:

$$P_{ss'} = P[S_{t+1} = s' | S_t = s]$$

Random Process:

$X(t, \xi)$ ⇒ 对于每个时刻 t_i, $X(t_i, \xi)$ 为随机变量

随机过程 $X(t,\xi)$ 为一组依赖于时间 t 的随机变量 $X(t_1,\xi)$, $X(t_2,\xi)$ ... $X(t_n,\xi)$ 组成。

Goal:

- Goal of RL: Find optimal policy to maximize the total amount of reward.
- Goal of MRP: Maximize the total amount of reward.

MRP:

- $<S, P, R, \gamma>$ ⇒ Markov Process with values.
- return G_t:

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

⇒ total discount reward from t.

why introduce discount γ:

(i) Avoid infinite returns

(ii) Represent uncertainty about the future

⇒ empirical design

- value function $V(s)$

$$v(s) = E[G_t | S_t = s] \Rightarrow$$

expected return starting from state s.

- Episodic task:

Special MRP that all sequences terminate.

MDP:

$$\langle S, A, P, R, \gamma \rangle \Rightarrow \text{MRP with decisions}$$

It's an environment that all states are Markov

has Markov property

- policy $\pi(a|s)$:

$$\pi(a|s) = P[A_t = a | S_t = s] \Rightarrow$$

(i) Policy is a distribution of actions given states.

(ii) Policy fully defined the behaviour of an agent

(iii) Policy is time-independent.

- state-value function $V_\pi(s)$:

$$v_\pi(s) = E_\pi[G_t | S_t = s] \Rightarrow$$

value function following policy π:

- action-value function: $q_\pi(s,a)$:

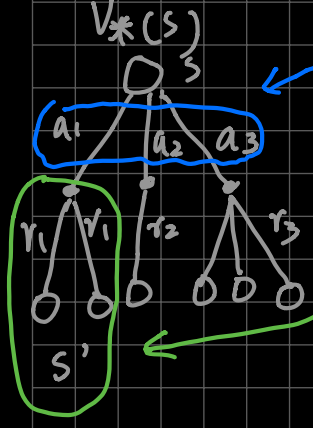
$$q_\pi(s,a) = \mathbb{E}_\pi[G_t \mid s_t = s,\ a_t = a] \Rightarrow$$

expected return starting from s, a and follows policy π.

Optimal Value Function: (specifies the best possible performance)

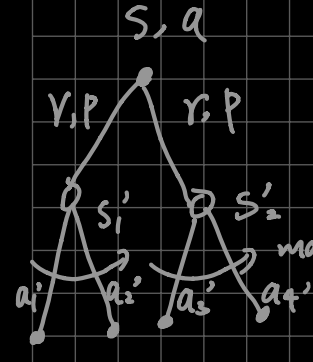
- state-value function:

$$V_*(s) = \max_\pi V_\pi(s) \Rightarrow \text{maximize value function over all policies (take different action)}$$



$$\Rightarrow = \max_a \left\{ R_s^a + \gamma \cdot \sum_{s' \in S} P_{ss'}^a \cdot V_*(s') \right\}$$

- action-value function:



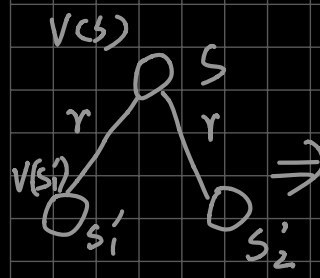
$$q_*(s,a) = \max_\pi q_\pi(s,a)$$

recursive form

$$= R_s^a + \gamma \cdot \sum_{s' \in S} P_{ss'}^a \max_{a'} q_*(s',a')$$

无法由 agent 决定的 take 一个 action 后 transition prob 由环境决定

Bellman Equation: (or Bellman Expectation Equation)



- MRP:

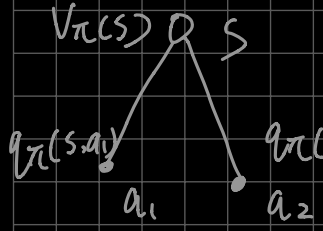
$$\Rightarrow v(s) = E[G_t \mid S_t = s] = E[\underbrace{R_{t+1} + \gamma \cdot v(S_{t+1})}_{\text{recursive form}} \mid S_t = s]$$

- MDP:

V_π:

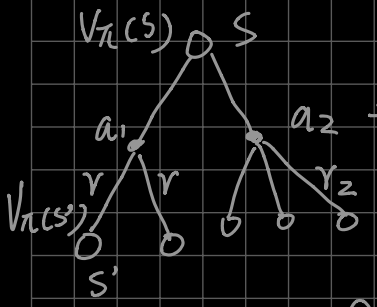
① $v_\pi(s) = E_\pi[G_t \mid S_t = s] = E_\pi[R_{t+1} + \gamma \cdot V_\pi(S_{t+1}) \mid S_t = s]$

regressive definition



$\Rightarrow$ ② $V_\pi(s) = \sum_{a \in A} \pi(a \mid s) \cdot q_\pi(s,a)$

1 layer definition



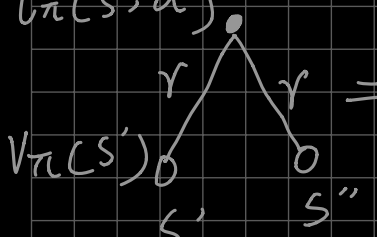
$\Rightarrow$ ③ $V_\pi(s) = \sum_{a \in A} \pi(a \mid s) \left(R_s^a + \gamma \cdot \sum_{s' \in S} P_{ss'}^a V_\pi(s')\right)$

2 layer definition

Q_π:

① $q_\pi(s,a) = E_\pi[G_t \mid S_t = s, A_t = a] = E_\pi[R_{t+1} + \gamma \cdot q_\pi(S_{t+1}, A_{t+1}) \mid S_t = s, A_t = a]$

regressive definition



$\Rightarrow$ ② $q_\pi(s,a) = R_s^a + \gamma \cdot \sum_{s' \in S} P_{ss'}^a V_\pi(s')$

1 layer definition

- Optimality Equation:

It is non-linear, so can't directly be solved. Need to use iterative solution methods. (Q-learning, SARSA)

Matrix Form: (Can be directly solved in this form)

- MRP:

following policy π

$$V = R + \gamma \cdot P \cdot V \xrightarrow{\text{per state form}}$$

$$\begin{bmatrix} V(1) \\ \vdots \\ V(n) \end{bmatrix} = \begin{bmatrix} R_1 \\ \vdots \\ R_n \end{bmatrix} + \gamma \cdot \begin{bmatrix} P_{11} & \cdots & P_{1n} \\ \vdots & & \\ P_{n1} & \cdots & P_{nn} \end{bmatrix} \begin{bmatrix} V(1) \\ \vdots \\ V(n) \end{bmatrix}$$

state → (columns) ; state ↓ (rows)

Solve: $V = (I - \gamma \cdot P)^{-1} \cdot R$, $O(n^3) \Rightarrow$

only for small MRPs, large MRPs cost is too high. Use iterative methods (DP · MC · TD) instead.

- MDP:

$$V_\pi = R^\pi + \gamma \cdot P^\pi \cdot V_\pi$$

Solve: $V_\pi = (I - \gamma \cdot P^\pi)^{-1} \cdot R^\pi$