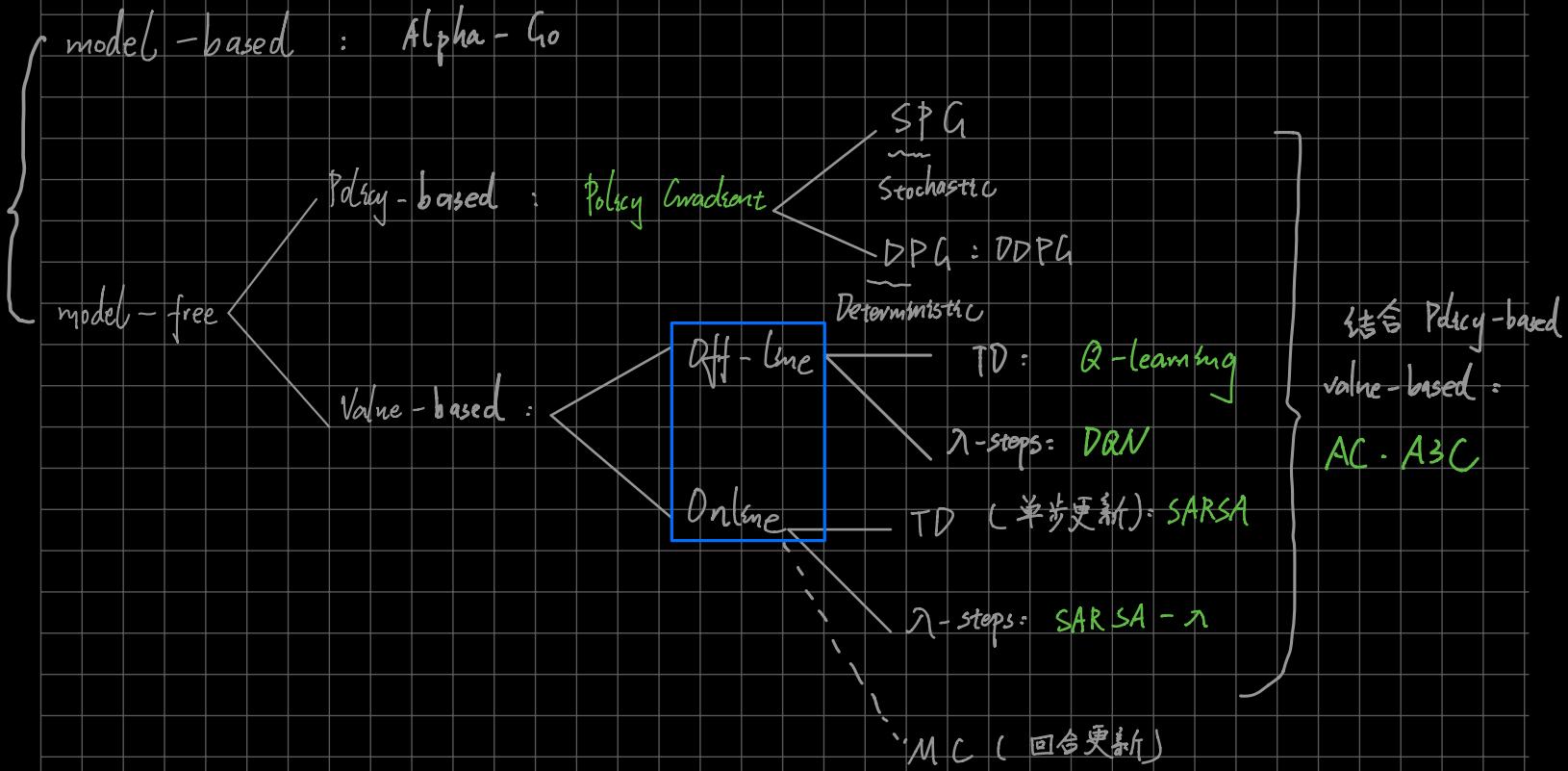


RL Methods classification



Brief Explanation:

Model-based method: Use a model to simulate real environment, then agent interact with the environment (model) to learn.

Model-free method: Agent 直接从真实的环境中得到反馈，并根据反馈进行学习。
采取行动

→ Model-based method 的优势为不需要真实环境反馈，避免了等待时间。

Model-free method 只能按步就班，在环境给出反馈后再进行决策。

Value based: 先求 action 的 Q value，再根据 Q value 得 optimal policy (Implicit policy)
(Indirect) e.g. \Rightarrow DQN, Q-learning, SARSA } RL 的 goal

Policy based: Using Policy gradient descent to find optimal policy.
(Explicit Policy) e.g. \Rightarrow DDPG } 为找到
task 的 optimal
policy

二者区别：

Policy-based method 输出的为动作的 概率分布，根据 概率来采取行动。

Value-based method 输出为动作的 取值，根据 最高 value 来采取行动

⇒ Action Space 是
—— 高维、连续的空间时，用 Policy-based method。
—— 低维、离散的空间时，用 Value-based method 中的 Q-table。
—— 高维、离散的空间时，用 Value-based method 中的 DQN。

MC 增量

$$MC : Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \underbrace{\alpha(R - Q(s_t, a_t))}_{MC \text{ error}} = (1-\alpha)Q(s_t, a_t) + \underbrace{\alpha \cdot R}_{MC \text{ error}}$$

MC 我们更倾向于用 G 。

$$G = R_{t+1} + \gamma \cdot R_{t+2} + \dots + \gamma^{T-t} R_{T+1}$$

Method of
Updating Q

1-Step TD:

$$TD(0) : Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \underbrace{\alpha [r_{t+1} + \gamma \cdot Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]}_{1\text{-Step TD error}}$$

(Temporal Difference)

不是 Time

n Steps TD (λ) (λ 一定指的 n steps, 入代表参数, 而非 steps)

$$TD(\lambda) : Q(s_t, a_t) \leftarrow (1-\lambda)Q(s_t, a_t) + \underbrace{\lambda \cdot R^{(n)}}_{n\text{-step TD error}}$$

$$R^{(n)} = \gamma^0 r_{t+1} + \gamma^1 r_{t+2} + \dots + \gamma^{n-2} r_{t+n-1} + \gamma^{n-1} Q(s_{t+n}, a_{t+n})$$

TD error:

$$R^\lambda = (1-\lambda) \sum_{i=1}^n \lambda^i \underbrace{R^{(i)}}_{i\text{-step TD error}}$$

⇒ ① MC error 可看作 Max-Steps TD error, 也为入=1的 TD(入)

② TD error 中, 入越大, 用到真实回报信息越多, 收敛越少。

③ MC: 只能用 情节式任务 (序列有终点) 情节式任务 & 连续型任务

Episodic MDP \Rightarrow All episodes must terminate.

Stochastic Policy Gradient $\Rightarrow \sum \pi(a|s) = 1$ ($\pi(a|s)$ 的值代表选择 action a 的概率)

Deterministic Policy Gradient $\Rightarrow \pi(s) = S \rightarrow A$ ($\pi(s)$ 的值代表选择的 action)

强化学习优势: Learning from interaction.

因此 RL 多应用于 游戏策略学习、博弈游戏学习、情景式对话等。

$\langle \underset{\sim}{A}, \underset{\sim}{S}, \underset{\sim}{R}, \underset{\sim}{P} \rangle$: 强化学习四元组

Action State Reward Transition

$$R: S_t \times A_t \times S_{t+1} \rightarrow R_t$$

见AI lecture slides.

$$P: S_t \times A_t \rightarrow S_{t+1}$$
, 称 model. 过程为 MDP.

MDP (Markov Decision Process) $\triangle RL \text{ is a MDP.}$

Basic Concepts.

Reward: 评判 Action 的立即回报好坏.

$$r_t = R(s_t, a_t)$$

G_t : 时间 t 下长期回报期望值

$$G_t = \sum_{n=0}^N \gamma^n r_{t+n}$$

\triangle Goal: 使平均累积回报量大

Discount Rate

$V_\pi(s)$ / Value: 策略 π 在状态 s 下长期期望收益
Value 是针对策略而言的

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

特定的 Action

Q : 策略 π 在状态 s , 动作 a 下长期期望收益.
 Q 是针对 action 而言的

$$Q_\pi(s, a) = E_\pi [G_t | S_t = s, A_t = a]$$

Bellman Equation: (贝尔曼期望方程, 又称动态规划方程)

与 action a , 状态 s 有关

$$V_{\pi}(s) = \sum_{a \in A} \pi(a|s) \cdot E [R_{t+1} + \gamma \cdot \sum_{s'} P_{ss'}^a V_{\pi}(s') | s_t = s]$$

各种 action 下总 value 值
对 action 和
不选择 action a 的
概率 (这是我们能控制的)
π 下选择 action a 的 value
→ 状态转移概率,
由环境决定

$$Q_{\pi}(s, a) = R(s, a) + \gamma \cdot \sum_{s'} P_{ss'}^a \sum_a \pi(a'|s') q_{\pi}(s', a')$$

reward 与 s' 无关, 只与
 s, a 有关

Bellman optimality equation: (Adapt from Bellman Equation, 所有
策略 (不同元的 Action
set A 不同) 的 value
之和为 1)

$$V_*(s) = E [R_{t+1} + \gamma \max_{\pi} V_{\pi}(s_{t+1}) | s_t = s]$$

Value 是针对策略而言的

$$Q_*(s, a) = E [R_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a') | s_t = s, a_t = a]$$

Q 是针对 action 而言的

RL 目前技术发展遇到的问题: 缺乏理论, 许多只是用实验证
实想法

本质来源: 控制论