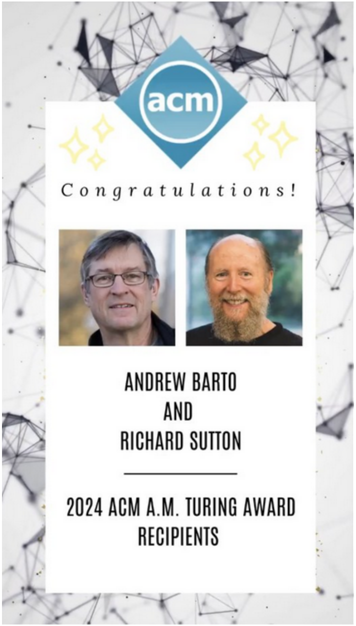
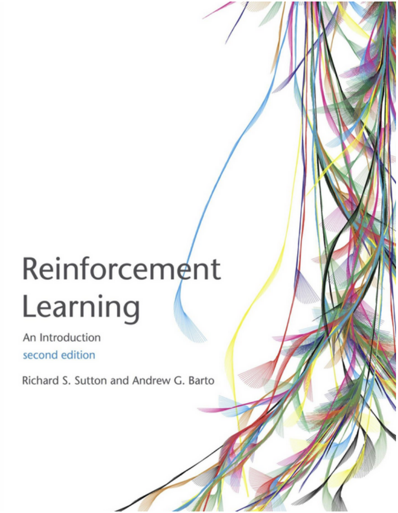
Lecture 15 - Markov Decision Process & Reinforcement Learning

赵尉辰

南开大学 统计与数据科学学院


acm
Congratulations!
ANDREW BARTO
AND
RICHARD SUTTON
2024 ACM A.M. TURING AWARD
RECIPIENTS


Reinforcement
Learning
An Introduction
second edition
Richard S. Sutton and Andrew G. Barto

一些学习资料[2]

- 王树森：深度强化学习；
- 张伟楠：强化学习[1]；
- 赵世钰：强化学习的数学原理。

[1] https://wnzhang.net/teaching/sjtu-rl-2024/
[2] https://mp.weixin.qq.com/s/0TaPFikmz83oVoMhQZPsDg

目录

目录

两种人工智能任务

学习：机器(算法)通过经验(数据)去提升具体任务指标(优化目标)的过程

- 预测型任务 (拟合数据)
    - 有监督学习：根据数据X预测所需输出/标签Y；
      例如：指纹识别、人脸识别；
    - 无监督学习：根据数据X预测其分布，并生成数据实例；
      例如：文本生成DeepSeek、图像生成；
- 决策型任务
    - 强化学习：在动态环境中根据一定策略采取行动；
      例如：下围棋AlphaGo、打游戏AlphaStar；

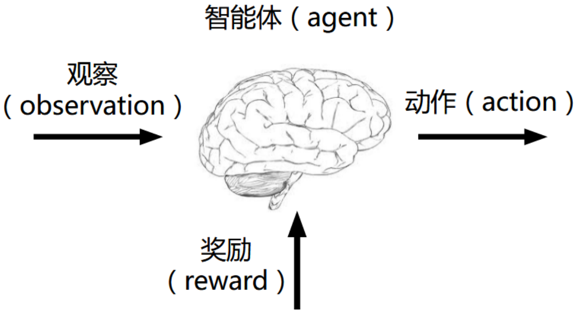
区别：预测型任务不改变环境，决策型任务影响动态环境变化。

决策型任务的分类

环境特性	**白盒环境** • 变量和目标之间的关系可以用具体公式表示	**黑盒环境** • 变量和目标之间的关系无法用具体公式表示
静态环境 • 环境没有转移的状态 • 单步决策	运筹优化 • （混合整数）线性规划 • 非线形优化	黑盒优化 • 神经网络替代模型优化 • 贝叶斯优化
动态环境 • 环境有可转移的状态 • 多步决策	动态规划 • MDP直接求解 • 树、图搜索	强化学习 • 策略优化 • Bandits、序贯黑盒

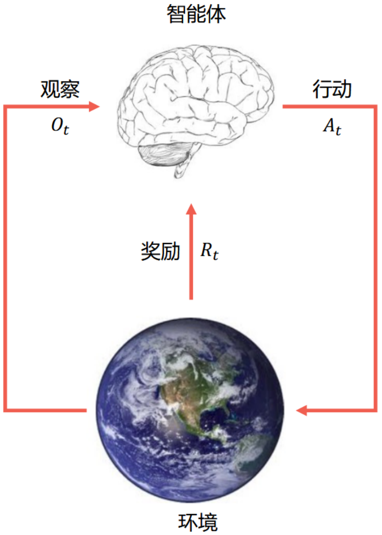
强化学习

强化学习是从与环境交互中学习来实现目标的计算方法。



Agent 与 Model 的区别：动作会直接影响/改变环境。

强化学习交互过程



- 在每一步t，智能体：
  - 获得观察O_t
  - 执行动作A_t
  - 获得奖励R_t
- 环境：
  - 获得动作A_t
  - 给出奖励R_t
  - 给出观察O_{t+1}

强化学习的要素

- 历史(History)是观察、动作和奖励的序列。

$$H_t = O_1, A_1, R_1, O_2, A_2, R_2, ..., O_{t-1}, A_{t-1}, R_{t-1}, O_t$$

  根据这个历史可以决定接下来会发生什么
  - 智能体给出动作A_t;
  - 环境给出奖励R_t，以及下一步观察O_{t+1}.

- 状态(State)是用来确定当前时间步t发生的事情(动作、奖励、观察)的信息。

  状态是关于历史的函数

$$S_t = f(H_t).$$

强化学习的要素

- 策略(Policy)是智能体在状态下的行为方式。
  - 确定性策略(Deterministic Policy) π是从状态集合S到动作集合A的映射
    $$a = \pi(s)$$
  - 随机策略(Stochastic Policy) π是如下条件概率
    $$\pi(a|s) = P(A_t = a|S_t = s)$$
- 奖励(Reward)：随机变量 R_t 和确定观测值 $r(a, s)$

目录

Markov决策过程

一个Markov决策过程 (Markov Decision Process, MDP) 可以表示为五元组：

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, P_{s,a}, \gamma, R)$$

其中：

- 状态空间 (State Space) $\mathcal{S}$：所有可能状态的集合，记为 $s \in \mathcal{S}$。
- 动作空间 (Action Space) $\mathcal{A}$：所有可能动作的集合，记为 $a \in \mathcal{A}$。
- 状态转移概率 (Transition Probability)： 在状态 s 执行动作 a 后转移到状态 s' 的概率，记为 $P_{s,a}(s') \triangleq P(S_{t+1} = s' | S_t = s, A_t = a)$.
- 折扣因子 (Discount Factor) $\gamma \in [0, 1)$：用于权衡当前奖励与未来奖励的重要性。
- 奖励函数 (Reward Function)： $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$, 也可以只和状态有关。

Markov决策过程

- Markov决策过程数学建模了结果部分随机、部分在决策者的控制下的决策过程

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, S_2, ..., S_t]$$

$$P[S_{t+1}|S_t, A_t] = P[S_{t+1}|S_1, A_1 ..., S_t, A_t]$$

- Markov决策过程数学形式化地描述了一种强化学习的环境
  - 环境完全可观测；
  - 当前状态可以完全表征之后的过程 (Markov 性)。

Markov决策过程

1. 从初始状态 $S_0 = s_0$ 开始;
2. 对于每一步 $t = 0, 1, 2, \ldots / t = 0, 1, \ldots, T$
   - 智能体选择某个动作$A_t = a_t \in \mathcal{A}$
   - 智能体得到奖励$R(s_t, a_t)$ / MDP以概率$P_{s_t,a_t}(s_{t+1})$转移到下一个状态s_{t+1}
   - MDP转移到下一个状态$S_{t+1} \sim P_{s_t,a_t}$ / 智能体得到奖励$r(s_t, a_t, s_{t+1})$
3. 一直进行或到终止时刻T，得到一回合(Episode) 的完整轨迹(Trajectory)

$$s_0 \xrightarrow{a_0, R(s_0, a_0)} s_1 \xrightarrow{a_1, R(s_1, a_1)} s_2 \xrightarrow{a_2, R(s_2, a_2)} s_3 \cdots$$

4. 计算智能体的回报 (Return)/累计奖励:

$$R(s_0, a_0) + \gamma R(s_1, a_1) + \gamma^2 R(s_2, a_2) + \cdots$$

$$r(s_0, a_0, s_1) + \gamma r(s_1, a_1, s_2) + \gamma^2 r(s_2, a_2, s_3) + \cdots$$

回报的随机性

随机性来源：

- 状态的随机性：来源于状态转移

$$p(s'|s,a) \triangleq P(S_{t+1}=s'|S_t=s, A_t=a)$$

- 动作的随机性：来源于随机策略

$$\pi(a|s) \triangleq P(A_t=a|S_t=s)$$

价值函数/策略评估(Policy Evaluation)

- 动作价值函数 (action-value function)

$$Q^\pi(s,a) = \mathbb{E}[R(S_0,A_0)+\gamma R(S_1,A_1)+\gamma^2 R(S_2,A_2)+\cdots|S_0=s,A_0=a,\pi]$$

$$Q^\pi(s_t,a_t) = \mathbb{E}[R(S_t,A_t)+\gamma R(S_{t+1},A_{t+1})+\cdots|S_t=s_t,A_t=a_t,\pi]$$

- 状态价值函数 (state-value function)

$$\begin{aligned} V^\pi(s) &= \mathbb{E}[R(S_0,A_0)+\gamma R(S_1,A_1)+\gamma^2 R(S_2,A_2)+\cdots|S_0=s,\pi] \\ &= \mathbb{E}_{A\sim\pi(\cdot|s)}\left[R(s,A)+\gamma\sum_{s'\in S} P_{s,A}(s')V^\pi(s')\right] \\ &= \mathbb{E}_{A\sim\pi(\cdot|s)}[Q^\pi(s,A)] \end{aligned}$$

$$V^\pi(s_t) = \mathbb{E}_{A_t\sim\pi(\cdot|s_t)}[Q^\pi(s_t,A_t)]$$

MDP的目标

目标：选择能够最大化累积奖励的动作

- 最优动作价值函数 (optimal action-value function)

$$Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a)$$

- 最优状态价值函数 (optimal state-value function)

$$V^*(s) = \max_{\pi} V^{\pi}(s)$$

问题1：价值函数如何计算？

问题2：在状态s下，策略改变了动作的选择后，策略整体是否变得更好？

贝尔曼方程(Bellman Equations)[3]

- 状态价值V^π - 状态价值V^π

$$V^\pi(s_t) = \mathbb{E}_{A_t, S_{t+1}}\left[R_t(S_t, A_t) + \gamma \cdot V^\pi(S_{t+1}) | S_t = s_t\right].$$

- 动作价值Q^π - 状态价值V^π

$$Q^\pi(s_t, a_t) = \mathbb{E}_{S_{t+1}}\left[R_t(S_t, A_t) + \gamma \cdot V^\pi(S_{t+1}) | S_t = s_t, A_t = a_t\right].$$

- 动作价值Q^π - 动作价值Q^π

$$Q^\pi(s_t, a_t) = \mathbb{E}_{S_{t+1}, A_{t+1}}\left[R_t(S_t, A_t) + \gamma \cdot Q^\pi(S_{t+1}, A_{t+1}) | S_t = s_t, A_t = a_t\right]$$

[3] 证明参考：《深度强化学习》，王树森、黎彧君、张志华，附录A

策略提升

定义 1 (策略提升)

对于两个策略π, π'，如果满足对于任何状态s，有

$$Q^\pi(s, \pi'(s)) \triangleq \mathbb{E}_{A \sim \pi'(\cdot|s)}[Q^\pi(s, A)] \geq V^\pi(s)$$

则称π'是π的策略提升*(Policy Improvement)*。

例. 给定MDP，π'是π的策略提升，如果：

- 在某个状态s_1下，两策略的输出不同，并且有

$$\pi'(\cdot|s_1) \neq \pi(\cdot|s_1), \quad Q^\pi(s_1, \pi'(s_1)) > Q^\pi(s_1, \pi(s_1)) = V^\pi(s_1)$$

- 在其他所有状态s下，两策略输出相同，即

$$\pi'(\cdot|s) = \pi(\cdot|s), \quad Q^\pi(s, \pi'(s)) = Q^\pi(s, \pi(s)) = V^\pi(s)$$

策略提升定理(Policy Improvement Theorem)

定理 1 (策略提升定理)

对于两个策略π, π'，如果π'是π的策略提升，则对于任何状态s，有

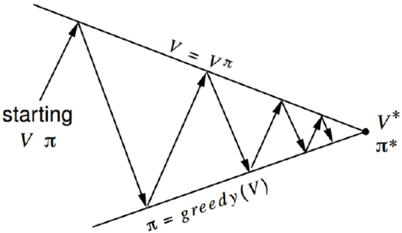
$$V^{\pi'}(s) \geq V^{\pi}(s)$$

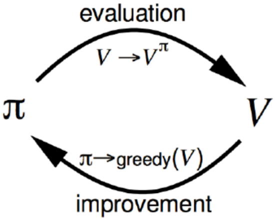
即是π'的期望回报超过π, 策略π'比策略π更好。

证明参考：Reinforcement Learning An Introduction, second edition by Richard S. Sutton and Andrew G. Barto, 4.2

策略提升定理

策略 π $\longrightarrow$ 评估策略价值 $V^{\pi}(\cdot)$ $\longrightarrow$ 寻找到新策略π'使得 $Q^{\pi}\big(s, \pi'(s)\big) \geq V^{\pi}(s)$





ϵ-Greedy 策略提升定理

定义 2

设动作空间的大小为m，ϵ-*Greedy*策略π定义为

$$\pi(a|s) = \begin{cases} \epsilon/m + 1 - \epsilon & \text{if } a^* = \arg\max_{a\in A} Q(s,a) \\ \epsilon/m & \text{otherwise} \end{cases}$$

定理 2

设π为一个ϵ-*Greedy*策略，如果另一个ϵ-*Greedy* 策略π'是基于Q^π的提升，即对于任何状态s，有

$$Q^\pi(s, \pi'(s)) \geq V^\pi(s)$$

那么有

$$V^{\pi'}(s) \geq V^\pi(s).$$

目录

动态规划

环境特性	**白盒环境** • 变量和目标之间的关系可以用具体公式表示	**黑盒环境** • 变量和目标之间的关系无法用具体公式表示
静态环境 • 环境没有转移的状态 • 单步决策	运筹优化 • （混合整数）线性规划 • 非线形优化	黑盒优化 • 神经网络替代模型优化 • 贝叶斯优化
动态环境 • 环境有可转移的状态 • 多步决策	动态规划 • MDP直接求解 • 树、图搜索	强化学习 • 策略优化 • Bandits、序贯黑盒

动态规划

回顾Bellman方程

$$V^\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V^\pi(s') \right]$$

$$\begin{aligned} Q^\pi(s,a) &= R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V^\pi(s') \\ &= R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) \sum_{a' \in \mathcal{A}} \pi(a'|s') Q^\pi(s',a') \end{aligned}$$

Bellman最优方程

$$V^*(s) = \max_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V^*(s') \right]$$

$$Q^*(s,a) = R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) \max_{a' \in \mathcal{A}} Q^*(s',a')$$

动态规划

最优策略

$$\pi^*(s) = \arg\max_{a\in\mathcal{A}} \left[\underbrace{R(s,a)}_{\text{需要 } R} + \gamma \underbrace{\sum_{s'\in\mathcal{S}} P(s'|s,a)}_{\text{需要 } P} \underbrace{V^*(s')}_{\text{已知}} \right]$$

直接比较$\{Q^*(s,a)\}_{a\in\mathcal{A}}$可得最优策略[4]

$$\pi^*(s) = \arg\max_{a\in\mathcal{A}} Q^*(s,a)$$

可以对最优策略和最优价值函数执行迭代更新:

- 策略迭代
- 价值迭代

[4]这两种方式定义的最优策略π^*是等价的。

策略迭代

基于状态价值函数V^π的策略迭代

1. 随机初始化策略π_0;
2. 对$k = 0, 1, 2, \ldots$重复以下过程直到收敛：
   - 策略评估：固定当前策略π_k，计算其状态价值函数V^{π_k}

     $$V_{n+1}^{\pi_k}(s) \leftarrow \sum_a \pi_k(a|s) \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V_n^{\pi_k}(s') \right]$$

     需迭代至$V_n^{\pi_k}$收敛，该步骤比较耗费计算资源。
   - 策略更新：基于V^{π_k}，对每个状态$s \in \mathcal{S}$选择贪心动作：

     $$\pi_{k+1}(s) \leftarrow \arg\max_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V^{\pi_k}(s') \right]$$

策略迭代

基于动作价值函数Q^π的策略迭代

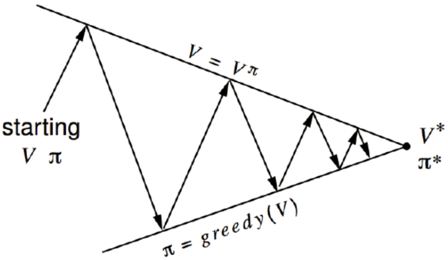
1. 随机初始化策略π_0;
2. 对$k = 0, 1, 2, \ldots$重复以下过程直到收敛：
   - 策略评估：固定当前策略π_k，计算其状态价值函数Q^{π_k}

     $$Q^{\pi_k}_{n+1}(s,a) \leftarrow R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) \sum_{a' \in \mathcal{A}} \pi_k(a'|s') Q^{\pi_k}_n(s',a')$$

     需迭代至$V^{\pi_k}_n$收敛，该步骤同样耗费计算资源。
   - 策略更新：基于Q^{π_k}，对每个状态$s \in \mathcal{S}$选择贪心动作：

     $$\pi_{k+1}(s) \leftarrow \arg\max_{a \in \mathcal{A}} Q^{\pi_k}(s,a)$$

加速策略迭代



价值迭代

1. 对每个状态$s \in \mathcal{S}$，初始化$V_0(s) = 0$;
2. 对$k = 0, 1, 2, \ldots$重复以下过程直到收敛至V^*：
   - 对所有状态$s \in \mathcal{S}$，基于Bellman最优方程同步更新：

$$V_{k+1}(s) \leftarrow \max_{a \in \mathcal{A}} \left[R(s, a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s, a) V_k(s') \right]$$

3. 收敛后，一次性计算最优策略：

$$\pi^*(s) \leftarrow \arg\max_{a \in \mathcal{A}} \left[R(s, a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s, a) V^*(s') \right]$$

注. 在以上的计算中没有明确的策略。

同步 vs. 异步价值迭代

- 同步(Synchronous)价值迭代会储存两份价值函数
  - 对所有状态$s \in \mathcal{S}$,

$$V_{new}(s) \leftarrow \max_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V_{old}(s') \right]$$

  - 更新：$V_{old} \leftarrow V_{new}$
- 异步(Asynchronous)价值迭代只储存一份价值函数
  - 对所有状态$s \in \mathcal{S}$,

$$V(s) \leftarrow \max_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V(s') \right]$$

策略迭代 & 价值迭代

特性	策略迭代 (Policy Iteration)	价值迭代 (Value Iteration)
核心目标	优化策略序列 $\pi_0, \pi_1, \ldots, \pi^*$	直接优化价值函数 $V_0, V_1, \ldots, V^*$
迭代结构	双重循环: 1. 策略评估：计算 V^{π_k} 2. 策略更新：$\pi_{k+1} \leftarrow \text{greedy}(V^{\pi_k})$	单循环: 直接更新 $V_{k+1} \leftarrow \mathcal{T}^* V_k$
中间策略	显式生成策略序列 π_k	无显式中间策略，最后一步提取 π^*
计算开销	高（每次策略评估需多次扫描状态）	低（每步单次更新）
内存需求	中等：需存储 V 和 π	低：只需存储 V
适用场景	策略空间小/需要中间策略序列	状态空间大/只关心最终策略
收敛保证	$\gamma < 1$ 时均收敛到唯一最优解 (V^*, π^*)	

目录

强化学习

环境特性	**白盒环境** • 变量和目标之间的关系可以用具体公式表示	**黑盒环境** • 变量和目标之间的关系无法用具体公式表示
静态环境 • 环境没有转移的状态 • 单步决策	运筹优化 •（混合整数）线性规划 • 非线形优化	黑盒优化 • 神经网络替代模型优化 • 贝叶斯优化
动态环境 • 环境有可转移的状态 • 多步决策	动态规划 • MDP直接求解 • 树、图搜索	强化学习 • 策略优化 • Bandits、序贯黑盒

强化学习

动态规划中，我们关注在给出一个已知MDP模型后，即，状态转移和奖励函数明确给定后：

- 计算最优价值函数；
- 学习最优策略；

然而在实际问题中，状态转移和奖励函数一般是无法明确给出的，我们只有一些数据：

Episode 1: $s_0^{(1)} \xrightarrow{a_0^{(1)}, r(s_0)^{(1)}} s_1^{(1)} \xrightarrow{a_1^{(1)}, r(s_1)^{(1)}} s_2^{(1)} \xrightarrow{a_2^{(1)}, r(s_2)^{(1)}} s_3^{(1)} \cdots s_T^{(1)}$

Episode 2: $s_0^{(2)} \xrightarrow{a_0^{(2)}, r(s_0)^{(2)}} s_1^{(2)} \xrightarrow{a_1^{(2)}, r(s_1)^{(2)}} s_2^{(2)} \xrightarrow{a_2^{(2)}, r(s_2)^{(2)}} s_3^{(2)} \cdots s_T^{(2)}$

...

Episode N: $s_0^{(N)} \xrightarrow{a_0^{(N)}, r(s_0)^{(N)}} s_1^{(N)} \xrightarrow{a_1^{(N)}, r(s_1)^{(N)}} s_2^{(N)} \xrightarrow{a_2^{(N)}, r(s_2)^{(N)}} s_3^{(N)} \cdots s_T^{(N)}$

无模型的强化学习

无模型的强化学习(Model-free Reinforcement Learning) 直接从经验中学习价值和策略，而无需构建 Markov 决策过程模型。

问题1：如何计算价值函数/策略评估？

问题2：如何策略更新/无模型策略控制？

无模型的强化学习

问题1：如何计算价值函数/策略评估？

问题2：如何策略更新/无模型策略控制？

Monte Carlo 价值函数估计

目标：从采取策略π交互出的经验片段中“学习”价值函数V^π.

记智能体的回报/累计奖励为

$$G_t \triangleq R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots$$

则对状态$s \in \mathcal{S}$，价值函数为

$$\begin{aligned} V^\pi(s) &\triangleq \mathbb{E}[R_0 + \gamma R_1 + \gamma^2 R_2 + \cdots | s_0 = s, \pi] \\ &= \mathbb{E}[R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots | s_t = s, \pi] \\ &= \mathbb{E}[G_t | s_t = s, \pi] \approx \frac{1}{N} \sum_{i=1}^{N} G_t^{(i)} \end{aligned}$$

其中$\{G_t^{(i)}\}_{i \in \{1,2,\ldots,N\}}$为从采取策略$\pi$交互出的$N$个经验片段中计算的累计奖励。

注. 只能对有限长度的MDP应用Monte Carlo方法，即片段有终止时刻。

Monte Carlo 价值函数估计

具体实现算法

1. 使用策略 π 采样片段

$$s_0^{(i)} \xrightarrow{a_0^{(i)}} s_1^{(i)} \xrightarrow{a_1^{(i)}} s_2^{(i)} \xrightarrow{a_2^{(i)}} s_3^{(i)} \ldots s_T^{(i)}, \quad i = 1, 2, \ldots, N$$

2. 在一个片段中的每个时间步长 t 的状态 s 都被访问
   - 计数增量更新

$$N(s) \leftarrow N(s) + 1$$

   - 计算回报G_t，总累计奖励增量更新

$$C(s) \leftarrow C(s) + G_t$$

   - Monte Carlo 估计价值为累计奖励的均值

$$V(s) \approx \frac{C(s)}{N(s)}, \quad V(s) \leftarrow V(s) + \frac{1}{N(s)}(G_t - V(s))$$

时序差分(Temporal Difference, TD)方法

智能体的回报/累计奖励为

$$G_t \triangleq R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots = R_t + \gamma G_{t+1}$$

那么价值函数也可以递归地定义为

$$V^\pi(s) = \mathbb{E}[G_t|s_t = s, \pi] = \mathbb{E}[R_t + \gamma G_{t+1}|s_t = s, \pi]$$

回顾Bellman方程

$$V^\pi(s) = \mathbb{E}\left[R_t + \gamma \cdot V^\pi(s_{t+1})|s_t = s\right].$$

启发我们利用$R_t + \gamma \cdot V^\pi(s_{t+1})$更新价值函数，即

$$V(s_t) \leftarrow V(s_t) + \alpha\Big(\underbrace{r_t}_{\text{观测值}} + \gamma \underbrace{V(s_{t+1})}_{\text{未来的猜测}} - V(s_t)\Big)$$

其中α调整更新的量。

MC方法 & TD方法

MC方法

- 必须等待片段结束，直到累计奖励已知；
- 只能从完整序列中学习；
- 只能在片段化的(有终止的)环境下工作。

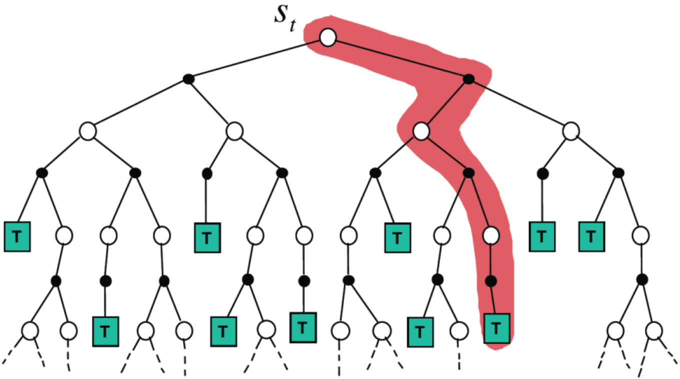
TD方法

- 能够在每一步之后进行在线学习；
- 能够从不完整的序列中学习；
- 能够在连续的(无终止的)环境下工作。

形象地说：MC像是一个必须等电影完全结束才能写影评的评论家，如果电影永不结束，他就永远写不出评论；而TD则像边看边写的评论家，每看一点新内容就能更新他的看法。

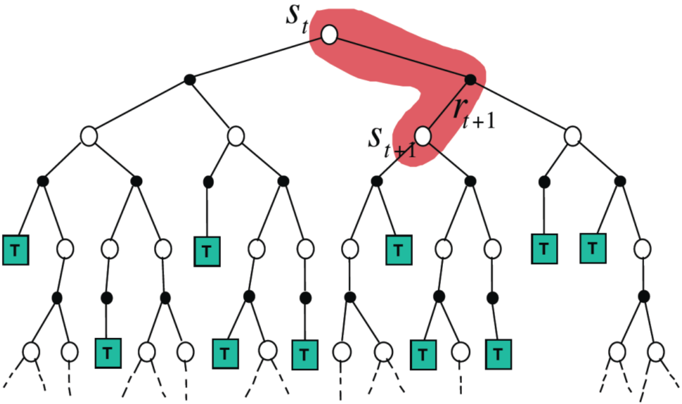
蒙特卡洛反向传播

$$V(S_t) \leftarrow V(S_t) + \alpha\big(G_t - V(S_t)\big)$$



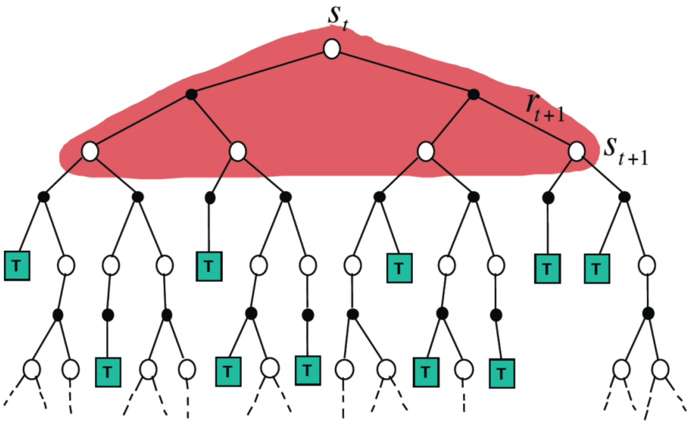
时序差分反向传播

$$V(S_t) \leftarrow V(S_t) + \alpha\left(R_{t+1} + \gamma V(S_{t+1}) - V(S_t)\right)$$



动态规划反向传播

$$V(S_t) \leftarrow \mathbb{E}[R_{t+1} + \gamma V(S_{t+1})]$$



无模型的强化学习

问题1：如何计算价值函数/策略评估？

问题2：如何策略更新/无模型策略控制？

无模型的强化学习

回顾最优策略的定义

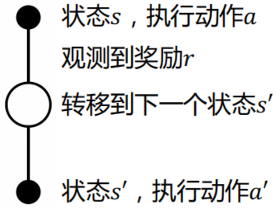
$$\pi^*(s) = \arg\max_{a\in\mathcal{A}} \left[\underbrace{R(s,a)}_{\text{需要 } R} + \underbrace{\gamma \sum_{s'\in\mathcal{S}} P(s'|s,a)}_{\text{需要 } P} \underbrace{V^*(s')}_{\text{已知}} \right]$$

$$\pi^*(s) = \arg\max_{a\in\mathcal{A}} Q^*(s,a)$$

因此，我们考虑对Q函数做策略控制。

SARSA

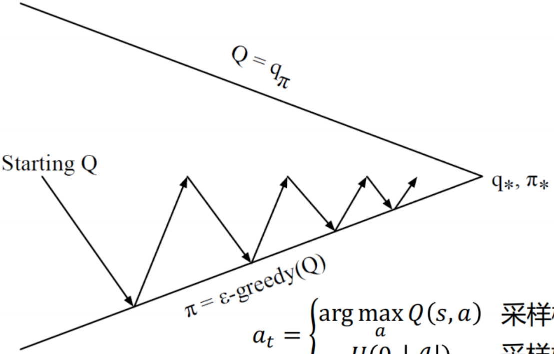
对于当前策略执行的每个(状态s-动作a-奖励r-状态s'-动作a')五元组



SARSA 更新动作价值函数

$$Q(s,a) \leftarrow Q(s,a) + \alpha(r + \gamma Q(s',a') - Q(s,a))$$

SARSA



$$a_t = \begin{cases} \arg\max_a Q(s,a) & \text{采样概率：} 1-\epsilon \\ U(0, |\mathcal{A}|) & \text{采样概率：} \epsilon \end{cases}$$

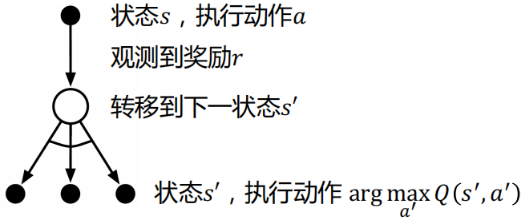
同策略与异策略

- 行为策略是指生成数据的策略，负责探索(exploration)方向；
- 目标策略是指待优化的策略，负责利用(exploitation)方向；
- 同策略(On-policy)：行为策略和目标策略是同一个策略。
  - 使用当前正在优化的策略与环境交互；
  - 直接学习该策略的值函数；
  - 探索与利用的平衡：策略本身需包含探索机制(如ϵ-贪婪)。
- 异策略(Off-policy)：行为策略和目标策略是不同策略。
  - 使用一个探索性行为策略(如ϵ-贪婪)生成数据；
  - 用这些数据优化另一个目标策略；
  - 数据复用：可用历史数据或不同策略生成的数据优化目标策略。

问题：如何实现异策略学习，即用别人的经验来更新我的策略？

Q-learning

对于Q函数来说，四元组：s-a-r-s' 是与策略无关的。对于同策略方法来说，a'是通过目标策略$\pi(\cdot|s')$得到的，但异策略方法允许行为策略与目标策略不同，那么干脆贪心的选择 $\arg\max_{a'\in\mathcal{A}} Q(s', a')$



Q-learning 更新动作价值函数

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_{t+1} + \gamma \cdot \max_{a'_{t+1}} Q(s_{t+1}, a'_{t+1}) - Q(s_t, a_t))$$

参数化价值函数近似

- 特征化状态$x(s)$
- 构建参数化（可学习的）函数来近似价值函数

$$V_\theta(s) \approx V^\pi(s)$$

例如：线性模型 $V_\theta(s) = \theta^\mathrm{T} x(s)$.

- 建立均方误差损失

$$J(\theta) \triangleq \mathbb{E}_\pi[\frac{1}{2}(V^\pi(s) - V_\theta(s))^2]$$

基于随机梯度下降优化

$$\theta \leftarrow \theta - \alpha \frac{\partial J(\theta)}{\partial \theta}$$

状态价值函数近似

目标：$\theta \leftarrow \theta + \alpha(V^{\pi}(s) - V_{\theta}(s))x(s)$

- MC方法：对于训练数据 $(s_1, G_1), (s_2, G_2), \ldots, (s_T, G_T)$

$$\theta \leftarrow \theta + \alpha(G_t - V_{\theta}(s))x(s_t)$$

- TD方法：对于训练数据 $(s_1, r_2 + \gamma V_{\theta}(s_2)), (s_2, r_3 + \gamma V_{\theta}(s_3)), \ldots, (s_T, r_T)$

$$\theta \leftarrow \theta + \alpha\left(r_{t+1} + \gamma V_{\theta}(s_{t+1}) - V_{\theta}(s)\right) x(s_t)$$

注. 对于Q函数来说同理。

策略梯度方法

- 参数化策略

$$\pi_\theta(a|s)$$

- 优化目标函数

$$\max_\theta J(\theta) \triangleq \max_\theta \mathbb{E}_{\pi_\theta}[R] = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T}\gamma^t r(s_t, a_t)\right]$$

其中$\tau = (s_0, a_0, s_1, a_1, \ldots)$是按照策略$\pi_\theta$得到的一个轨迹。

定理 3 (策略梯度定理)

对任意可微的策略$\pi_\theta(a|s)$，有

$$\nabla_\theta J(\theta) = \mathbb{E}_{(s_t,a_t)\sim\pi_\theta}\left[\nabla_\theta \log \pi_\theta(a_t \mid s_t) \cdot Q^{\pi_\theta}(s_t, a_t)\right]$$

总结

一个Markov决策过程可以表示为五元组：

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, P_{s,a}, \gamma, R)$$

其中：

- 状态空间 (State Space) $\mathcal{S}$：所有可能状态的集合，记为 $s \in \mathcal{S}$。
- 动作空间 (Action Space) $\mathcal{A}$：所有可能动作的集合，记为 $a \in \mathcal{A}$。
- 状态转移概率 (Transition Probability)： 在状态 s 执行动作 a 后转移到状态 s' 的概率，记为 $P_{s,a}(s') \triangleq P(S_{t+1} = s' | S_t = s, A_t = a)$.
- 折扣因子 (Discount Factor) $\gamma \in [0, 1)$：用于权衡当前奖励与未来奖励的重要性。
- 奖励函数 (Reward Function)： $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$, 也可以只和状态有关。

总结

- 动作价值函数

$$Q^\pi(s_t, a_t) = \mathbb{E}[R(S_t, A_t) + \gamma R(S_{t+1}, A_{t+1}) + \cdots | S_t = s_t, A_t = a_t, \pi]$$

- 状态价值函数

$$V^\pi(s_t) = \mathbb{E}_{A_t \sim \pi(\cdot|s_t)}[Q^\pi(s_t, A_t)]$$

- Bellman方程

$$V^\pi(s_t) = \mathbb{E}_{A_t, S_{t+1}}\left[R_t(S_t, A_t) + \gamma \cdot V^\pi(S_{t+1}) | S_t = s_t\right].$$

定理 4 (策略提升定理)

对于两个策略π, π'，如果π'是π的策略提升，则对于任何状态s，有

$$V^{\pi'}(s) \geq V^\pi(s)$$

即是π'的期望回报超过π，策略π'比策略π更好。

总结

策略迭代

- 优化策略序列 $\pi_0, \pi_1, \ldots, \pi^*$；
- 双重循环：
  1. 策略评估：计算 V^{π_k}
  2. 策略更新：

  $\pi_{k+1} \leftarrow \text{greedy}(V^{\pi_k})$；
- 显式生成策略序列 π_k.

价值迭代

- 直接优化价值函数$V_0, V_1, \ldots, V^*$；
- 单循环：直接更新$V_{k+1} \leftarrow \mathcal{T}^* V_k$；
- 无显式中间策略，最后一步提取 π^*.

前沿

- 深度强化学习；
- 多智能体强化学习；
- 强化学习在大语言模型的应用。