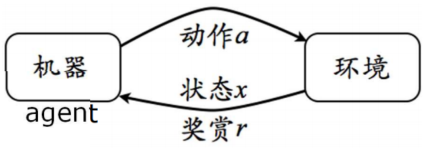
一、基本概念

通常用马尔可夫决策过程（MDP）描述

四元组

关键要素：<A, X, R, P>

action space: A
state space: X
reward: $R: X \times A \times X \to \mathbb{R}$
transition: $P: X \times A \times X \to \mathbb{R}$



E=<X,A,P,R>

- 机器在环境E中
- 状态空间X，每个状态x是机器感知到的环境的描述
- 动作空间A，机器可以操作的动作
- 潜在转移函数P，执行动作a时，由P把当前状态以某种概率转移到另一个状态
- 潜在奖励函数R，转移到另一个状态时进行奖赏或惩罚

机器要学习一个（长期累积奖励最大化的）策略π，在状态x下知道要执行的动作或π（x，a）：x状态下执行a的概率

策略评价：累积回报

$$\text{T-step: } \frac{1}{T}\sum_{t=1}^{T} r_t \qquad \text{discounted: } \sum_{t=1}^{\infty} \gamma^t r_t$$

二、探索和利用

探索-利用窘境

尝试次数有限，两者矛盾

仅探索能更好的估计出奖赏，但失去很多获得奖励机会

仅利用没有很好估计奖赏，错失最优摇臂

需要平衡方法

ε-贪心

每次尝试以ε概率探索，其余概率选择当前平均奖赏最高的摇臂

每尝试一次就更新平均奖赏Q（k）

一段时间后，可以不再探索，即逐步缩小ε的值

Softmax算法

平均奖赏高的，被选取摇臂的概率也更高

三、有模型学习

四元组已知，机器内部能模拟出与环境近似的情况，在已知模型的环境中学习

状态值函数V：从x出发执行策略π的累计奖赏

状态-动作值函数Q：从x出发执行动作啊后再策略π的累计奖赏

状态值函数 state value function

$$V^{\pi}(x) = E[\sum_{t=1}^{T} r_t | x]$$

状态动作值函数 state-action value function

$$Q^{\pi}(x,a) = E[\sum_{t=1}^{T} r_t | x, a] = \sum_{x' \in X} P(x'|x,a)\Big(R(x,a,x') + V^{\pi}(x')\Big)$$

两者关系

$$V^{\pi}(x) = \sum_{a \in A} \pi(a|x) Q^{\pi}(x,a)$$

根据V和Q的关系

$$Q^{*}(x,a) = \sum_{x' \in X} P(x'|x,a)(R(x,a,x') + \gamma V^{*}(x'))$$

由于 MDP具有马尔可夫性质，即系统下一时刻的状态仅由当前时刻的状态决定，不依赖于以往任何状态，于是值函数有很简单的递归形式

由于最优值函数的累积奖赏值已达最大，因此可对前面的 Bellman 等式(16.7)和(16.8)做一个改动，即将对动作的求和改为取最优：

$$\begin{cases} V_T^*(x) = \max_{a \in A} \sum_{x' \in X} P_{x \to x'}^a \left(\frac{1}{T} R_{x \to x'}^a + \frac{T-1}{T} V_{T-1}^*(x')\right); \\ V_\gamma^*(x) = \max_{a \in A} \sum_{x' \in X} P_{x \to x'}^a \left(R_{x \to x'}^a + \gamma V_\gamma^*(x')\right). \end{cases} \tag{16.13}$$

换言之，

$$V^*(x) = \max_{a \in A} Q^{\pi^*}(x,a). \tag{16.14}$$

代入式(16.10)可得最优状态-动作值函数

$$\begin{cases} Q_T^*(x,a) = \sum_{x' \in X} P_{x \to x'}^a \left(\frac{1}{T} R_{x \to x'}^a + \frac{T-1}{T} \max_{a' \in A} Q_{T-1}^*(x',a')\right); \\ Q_\gamma^*(x,a) = \sum_{x' \in X} P_{x \to x'}^a \left(R_{x \to x'}^a + \gamma \max_{a' \in A} Q_\gamma^*(x',a')\right). \end{cases} \tag{16.15}$$

上述关于最优值函数的等式，称为最优 Bellman 等式，其唯一解是最优值函数.

- 强化学习
  - 基本概念、四元组。
- 多摇臂赌博机
  - 探索和利用、解决方法
- 有模型强化学习
  - 状态值函数Q和状态-动作值函数V
  - Bellman等式，最优Bellman等式（关于Q和V的）
  - 策略迭代和值迭代
- 免模型强化学习
  - 蒙特卡洛学习，时序差分学习
  - on-policy，off-policy
- 值函数近似
- 模仿学习
- 逆强化学习

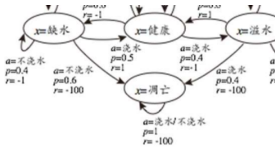
MDP是什么

是序贯决策（sequential decision）的数学模型，用于在系统状态具有马尔可夫性质的环境中模拟智能体可实现的随机性策略与回报。MDP基于一组交互对象，即智能体和环境进行构建，所具有的要素包括状态、动作、策略和奖励。在MDP的模拟中，智能体会感知当前的系统状态，按策略对环境实施动作，从而改变环境的状态并得到奖励，奖励随时间的积累被称为回报

QV互相推

MRP:

$$V(x) = \sum_{x' \in X} P(x'|x)\Big(R(x') + V(x')\Big)$$



MDP:

$$V^{\pi}(x) = \sum_{a \in A} \pi(a|x) \sum_{x' \in X} P(x'|x,a)\Big(R(x,a,x') + V^{\pi}(x')\Big)$$

状态值函数 state value function

$$V^{\pi}(x) = E[\sum_{t=1}^{T} r_t | x]$$

状态动作值函数 state-action value function

$$Q^{\pi}(x,a) = E[\sum_{t=1}^{T} r_t | x, a] = \sum_{x' \in X} P(x'|x,a)\Big(R(x,a,x') + V^{\pi}(x')\Big)$$

两者关系

$$V^{\pi}(x)=\sum_{a\in A}\pi(a|x)Q^{\pi}(x,a)$$

根据V和Q的关系

$$Q^{*}(x,a)=\sum_{x'\in X}P(x'|x,a)(R(x,a,x')+\gamma V^{*}(x'))$$

其中，使用了动作改变条件

$$Q^{\pi}(x,\pi'(x))\geq V^{\pi}(x)$$

以及状态-动作值函数

$$Q^{\pi}(x',\pi'(x'))=\sum_{x'\in X}P_{x'\to x'}^{\pi'(x')}(R_{x'\to x'}^{\pi'(x')}+\gamma V^{\pi}(x'))$$

策略迭代和值迭代收敛，怎么从bellman推出来的

$$Q^{\pi_{t+1}}(x,a)=\sum_{s'}P(x'|x,a)\big(R(x,a,x')+\gamma\max_{a}Q^{\pi_t}(x',a)\big)$$

$$V_{t+1}(x)=\max_{a}\sum P(x'|x,a)\big(R(x,a,x')+\gamma V_t(x')\big)$$

策略迭代算法估计的是状态值函数V，而最终的策略是通过状态-动作值函数Q来获得. 当模型已知时，从 V 到 Q有很简单的转换方法，

分清楚有模型方法和无模型方法

有：还原R和P，求解MDP最优策略。

预测时DP需要更新所有轨迹，需要知道环境；

通过求V计算Q，通过使用Q来对策略进行max改善。

无：直接逼近最优策略

使用计算出的Q函数直接改善，这是因为没有环境所以求不出状态价值函数

蒙特卡洛和时序差分各有优缺点，区别

蒙：通过考虑采样轨迹，解决了策略迭代无法应用于模型未知情况的问题

缺点：算法效率低，采用非深度优先策略，需要采样整条轨迹后再更新策略的值估计，没有充分利用MDP结构。

时序：结合了动态规划和蒙的思想，算法在每执行一步策略后就进行值函数更新

两个policy什么意思，采样策略和优化策略不一样怎么解决问题

on：被评估的和被改进的是同一个策略。更新策略自己的网络，得靠策略自己生成的数据去更新，不能靠其他时刻的策略。

off：选择动作的策略与即将更新的策略网络是不一致的，比如在原始策略上加一个ϵ-greedy

Sarsa算法和Q-learning算法在迭代过程中的区别是什么

Sarsa是同策略算法，选择动作时使用ϵ贪心策略

Q-Learning是异策略算法，选择动作是使用确定性策略

模仿，逆是要学什么，要解决什么问题

模仿：不只是获得多步决策后的累积奖励，还有人类专家的决策过程范例

解决问题：多步决策搜索空间巨大，基于累积奖励学习多步之前的合适决策很困难。可直接模仿专家的”状态动作对“

逆解决问题：设计奖赏很难，从专家范例中反推出奖赏是有利的

附加题难度：

需要脱离课本的东西

已知Bellman最优方程：

$$U^*(s) = \max_a \left(R(s,a) + \gamma \sum_{s'} T(s' \mid s,a) U^*(s') \right)$$

和值迭代的更新公式：

$$U_{k+1}(s) \leftarrow \max_a \left[R(s,a) + \gamma \sum_{s'} T(s' \mid s,a) U_k(s') \right]$$

试证明：当$\|U_k - U_{k-1}\|_\infty < \delta, \delta = \frac{\epsilon(1-\gamma)}{\gamma}$时，$\|U^* - U_k\|_\infty < \epsilon$，其中$\|U_k - U_{k-1}\|_\infty = \max_{s'} |U_k(s') - U_{k-1}(s')|$