法律声明

- 本课件包括：演示文稿，示例，代码，题库，视频和声音等，小象学院拥有完全知识产权的权利；只限于善意学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或机构不得盗版、复制、仿造其中的创意，我们将保留一切通过法律手段追究违反者的权利。

关注 **小象学院**

第4讲　蒙特卡罗法

强化学习

主讲人：叶梓

上海交通大学博士

主要研究方向：机器学习、深度学习、人工智能

本章内容

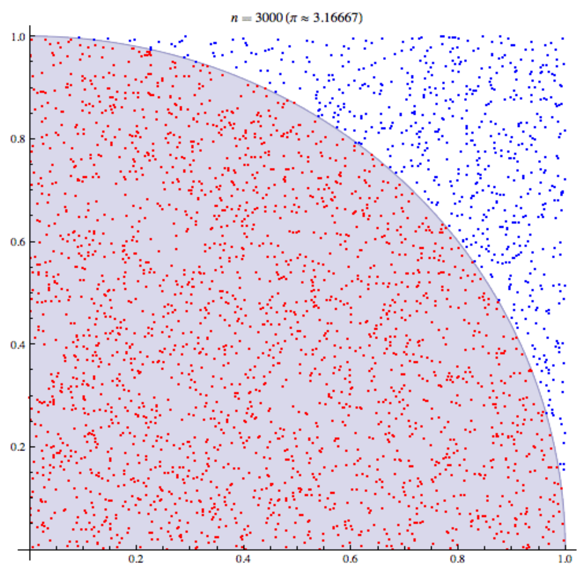
- 蒙特卡罗法的起源概述
- “经验”与“一幕”
- 首次访问与每次访问
- 蒙特卡罗控制
- 同策略与异策略
- 重要性采样
- MC与DP的差异
- 案例演示

蒙特卡罗法——起源

- 蒙特卡罗方法，是用随机数来解决计算的问题，是一类重要的数值计算方法。
- 该方法的名字来源于世界著名的赌城蒙特卡罗，是一种以概率为基础的方法。


蒙特卡罗法——起源

如何近似计算圆周率π?

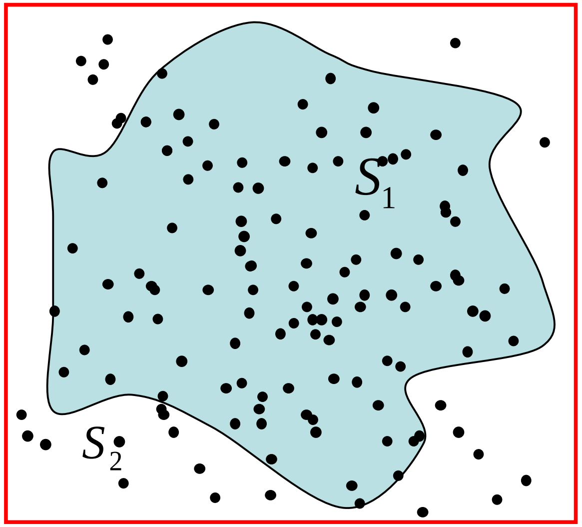


$$\frac{S_1}{S_2}=\frac{\pi/4}{1}=\frac{\pi}{4}\approx\frac{N_1}{N_2} \qquad \therefore \pi \approx \frac{4N_1}{N_2}$$

当N_2=30000时，π的估计值与真实值只相差0.07%

1、随机样本
2、试验多次
3、总结经验

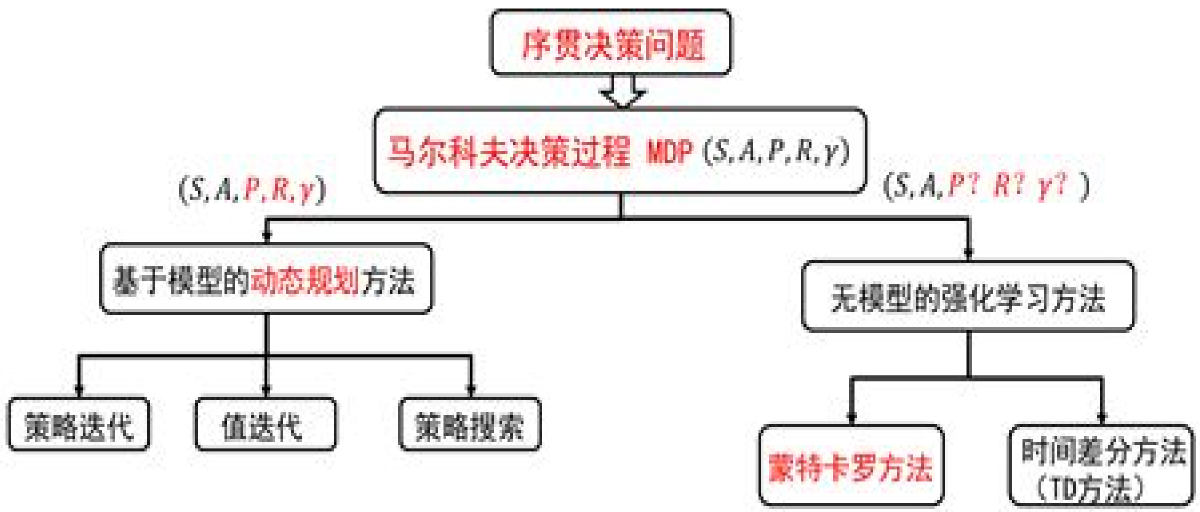
如何测量不规则图形的面积?



$$\frac{S_1}{S_2}\approx\frac{N_1}{N_2}$$

蒙特卡罗法——与DP的差异

- 若模型已知时，MDP可以利用动态规划的方法来解决。
- 动态规划方法计算状态处的值函数时利用了模型$P_{ss'}^{a}$,而在无模型强化学习中，模型$P_{ss'}^{a}$是未知的。



无模型的强化学习

- 无模型的强化学习算法要想利用策略评估和策略改善的框架，必须采用其他的方法对当前策略进行评估（计算值函数）。
- 只能回到值函数最原始的定义公式：

当前时刻，回报的期望

$$v_\pi(s) = E_\pi\left[G_t \mid S_t = s\right] = E_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s\right]$$

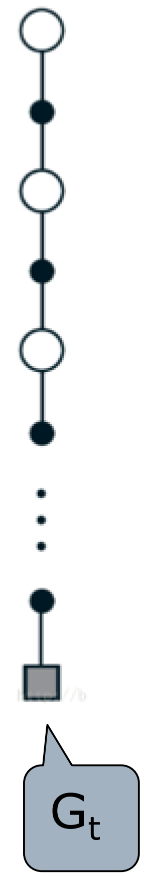
$$q_\pi(s) = E_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a\right]$$

蒙特卡罗法——概述

- MC法直接根据“经验”(experience)中的“一幕幕”(episodes)进行学习。
- MC法是model-free的（不需要理解环境），即MDP中的状态转移矩阵P（知道Reward吗？），对于Agent来说是未知的。
- MC法从完整的一幕中学习，而无需“自举”（bootstrapping）。
  - 每一幕都必须到终点。
- MC的基本思想：value就是return（回报）的平均值。

蒙特卡罗法——Episode

- MC法：在不清楚MDP状态转移概率及即时奖励的情况下，直接从经历完整的Episode来学习状态价值。
- Episode的含义是“一幕”或“回合”，完整的Episode不要求起始状态一定是某一个特定的状态，但是要求Agent最终进入环境的某一个终止状态。
  - 由于**不知道状态转移概率**，每一幕都要经历到终点。
  - 因为到终点才能得到总回报。



蒙特卡罗法——经验

- 什么是MC中的经验？经验是一组episodes的集合，其实就是训练样本。
- 基本思想：在完整的Episode中，用平均回报值(return)代替价值v。
- 比如在某一状态s，遵循策略π，最终获得了总回报G，这就是一个样本。如果有许多这样的样本，就可以估计在状态s下，遵循策略π的期望回报，也就是状态值函数$V_{\pi}(s)$了。
- 根据统计的原理，Episode越多，结果越准确。

状态行动值评估

- 当有模型时，只需要state values 就可以确定一个规则，这时只需要选择会引向最好的 reward 和下一个状态的 action 即可；
- 当模型未知时，获得“状态-行动值”比“状态值”更重要，因为仅仅具有 state values 不足确定一个规则，需要明确地知道每个 action values。
- 因此，蒙特卡罗法的一个重要目标就是评估 q^*。

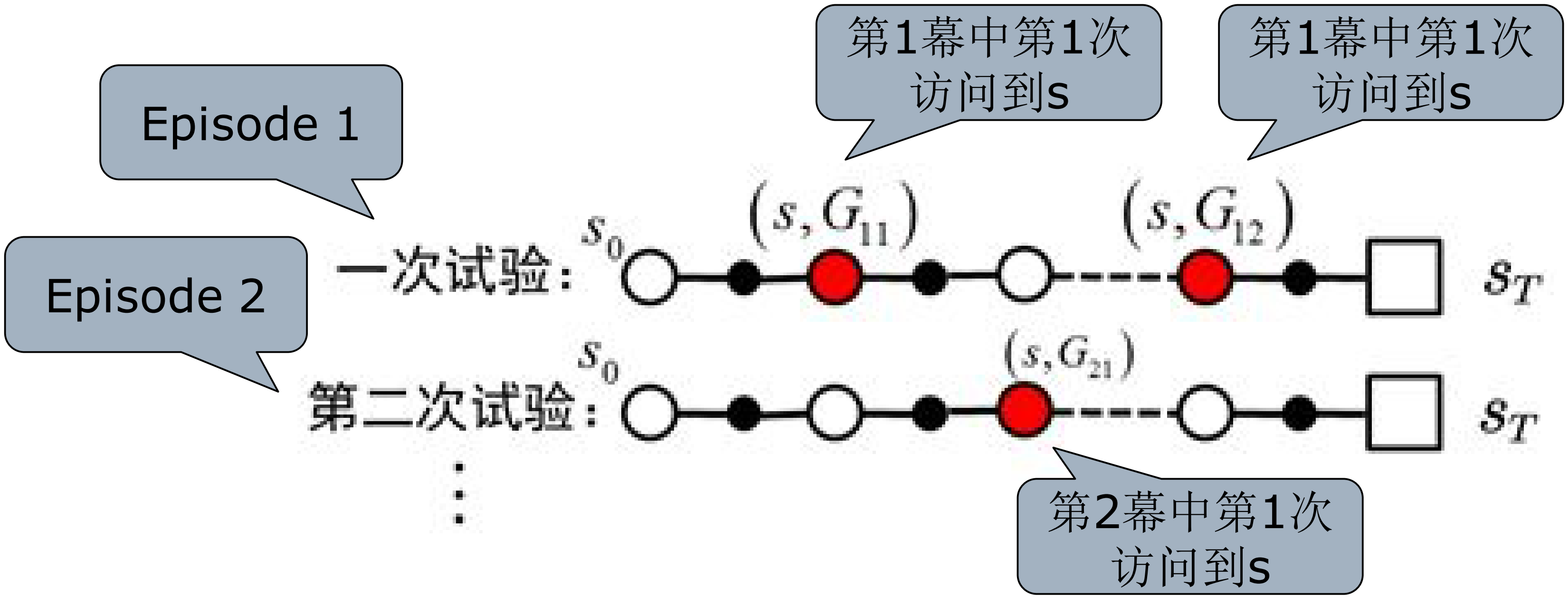
状态行动值评估

- 要评估 q*，首先应该考虑的状态-行动值的评价问题，也就是估计 $q_\pi(s,a)$（在状态 s 下执行行为 a 的期望 return）。
- MC 方法估计 q* 的思想与估计状态值v是一样的。
- 在一个episode 中，一对 s,a 称为是被visit过，是指过在该 episode 中，agent 经历过s 状态且选择执行了行为 a。

首次访问与每次访问

- 考虑一个问题，如果在某一个episode中，状态s出现了两次，分别在t_1时刻和t_2时刻，计算状态s的值时是仅用第一个还是两个都用呢？
- 引出了两种方法：首次访问（first-visit）MC 与每次访问（every-visit）MC：
- 其中 first-visit MC 方法是在各幕中，对 (s,a) 的所有 first visits 进行平均；
- 而 every-visit MC 方法是在各幕中，对 (s,a) 的所有 visits 进行平均。

首次访问与每次访问



- 目标：在给定策略下，从一系列的完整Episode经历中学习得到该策略下的状态价值函数。

首次访问蒙特卡洛策略评估

- 首次访问蒙特卡罗方法是指，在计算状态s处值函数时，只利用每次试验中第一次访问到状态s时的返回值。
- 如上一页中的图，计算状态s处的均值时只利用$G_{i1}(s)$。因此第一次访问蒙特卡罗方法的计算公式为：

$$v(s) = \frac{G_{11}(s) + G_{21}(s) + \cdots}{N(s)}$$

每次访问蒙特卡洛策略评估

- 每次访问蒙特卡罗方法是指，在计算状态S处的值函数时，利用所有访问到状态S时的回报返回值：

$$v(s) = \frac{G_{11}(s) + G_{12}(s) + \cdots + G_{21}(s) + \cdots}{N(s)}$$

- 根据大数定理
- 当：$N(s) \to \infty$ 则有，$v(s) \to v_\pi(s)$

增量计算均值

- 不论是First-Visit还是Every-Visit，在计算回报均值时，都是利用总回报除以状态s的总访问次数。
- 能否对均值进行增量式的求取?

是可以的！

$$
\begin{aligned}
\mu_k &= \frac{1}{k}\sum_{j=1}^{k} x_j \\
&= \frac{1}{k}\left(x_k + \sum_{j=1}^{k-1} x_j\right) \\
&= \frac{1}{k}\left(x_k + (k-1)\mu_{k-1}\right) \\
&= \mu_{k-1} + \frac{1}{k}\left(x_k - \mu_{k-1}\right)
\end{aligned}
$$

这个推导很重要！

增量计算均值

- 将一般的均值求取变为增量式均值求取的过程，我们可以很快地将其应用于MC方法中

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

$$V(S_t) \leftarrow V(S_t) + \frac{1}{N(S_t)}(G_t - V(S_t))$$

- 这里将MC方法变为增量式，便可以使得算法不关心计数值N(t)，而换为类似于学习速率的参数：

$$V(S_t) \leftarrow \underbrace{V(S_t)}_{\text{均值}} + \alpha(\underbrace{G_t}_{\text{新值}} - \underbrace{V(S_t)}_{\text{均值}}))$$

探索的必要性

- 蒙特卡罗方法是利用经验平均来估计值函数，能否得到正确的值函数，完全取决于经验。
- **获得充足的经验是无模型强化学习的关键。**
  - 在动态规划方法中，为了保证值函数的收敛性，算法会对状态空间中的状态进行逐个扫描。
- 因此，在蒙特卡洛方法中必须采用一定的方法保证每个状态都能被访问到。

探索的必要性

- 应聘者先打开了左边的门，得到的奖励为R=0，则V(left)=0，然后打开了右边的门，得到的奖励为R=1，所以V(right)=+1，如果按照贪婪法则：因V(right)>V(left)，所以应聘者会继续打开右边的门，
- 如果右边的门获取到的奖励始终使得V(right)>V(left)成立，则应聘者再也不会打开左边的门，哪怕左边的门中有很大的概率R=100。

"Behind one door is tenure - behind the other is flipping burgers at McDonald's."

- There are two doors in front of you.
- You open the left door and get reward 0
  $V(left) = 0$
- You open the right door and get reward +1
  $V(right) = +1$
- You open the right door and get reward +3
  $V(right) = +2$
- You open the right door and get reward +2
  $V(right) = +2$

$\vdots$

探索性初始化

- 必须采用一定的方法保证每个状态都能被访问到。第一种方法是：探索性初始化。
- 所谓探索性初始化是指每个状态都有一定的几率作为初始状态。
- 这里还需要介绍策略改进方法，以及便于进行迭代计算的平均方法。
- 蒙特卡罗方法利用经验平均对策略值函数进行估计。当值函数被估计出来后，对于每个状态s，通过最大化动作值函数，来进行策略的改进。

蒙特卡罗控制

- 在 DP 方法中我们知道，将策略评估与策略改进相结合就构成了 policy iteration 过程，先回顾一下 DP 的 policy iteration 过程：

$$\pi_0 \xrightarrow{E} v_{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} v_{\pi_1} \xrightarrow{I} \pi_2 \xrightarrow{E} \cdots \xrightarrow{I} \pi_* \xrightarrow{E} v_{\pi_*}$$

状态值

- 而在 MC 中， policy iteration 过程是这样的：
- 其中，$\xrightarrow{E}$代表的是一个完整的策略评估的过程，$\xrightarrow{I}$代表的是一个完整的 策略改进的过程。

状态-行为值

$$\pi_0 \xrightarrow{E} q_{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} q_{\pi_1} \xrightarrow{I} \pi_2 \xrightarrow{E} \cdots \xrightarrow{I} \pi_* \xrightarrow{E} q_{\pi_*}$$

这里为什么要用q值？

- 策略迭代包括两个部分，策略估计和策略改进，其中策略估计用的是贝尔曼期望方程，策略改进用的是greedy方法。
- 如果要将其变为模型无关的方法，那么在策略估计中就不能使用贝尔曼期望方程，而是变为sample方法，比如MC方法（或TD方法）。
- 假如用MC方法对值函数进行估计，得到策略对应的状态值函数V，然后利用greedy方法得到新策略：

  - Greedy policy improvement over $V(s)$ requires model of MDP

$$\pi'(s) = \underset{a \in \mathcal{A}}{\operatorname{argmax}}\ \mathcal{R}_s^a + \mathcal{P}_{ss'}^a V(s')$$

未知

这里为什么要用q值？

- 基于状态值函数的策略改进中需要理解模型的知识，而MC方法不了解模型，即**转移概率矩阵P**。
- 因此不能使用状态值函数作为估计的对象。
- 而对于状态-行为值函数，如果已经完成了策略估计，那么只需要求一个argmax即可对策略进行改进，所以可以使用q函数作为估计对象。

- Greedy policy improvement over $Q(s, a)$ is model-free

$$\pi'(s) = \underset{a \in \mathcal{A}}{\operatorname{argmax}}\ Q(s, a)$$

探索性初始化蒙特卡罗控制

- 采用了 exploring starts 的MC 方法称为 Monte Carlo ES，算法的伪代码如下所示：

Monte Carlo ES (Exploring Starts)

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:
  $Q(s, a) \leftarrow$ arbitrary
  $\pi(s) \leftarrow$ arbitrary
  $Returns(s, a) \leftarrow$ empty list

Repeat forever:
  Choose $S_0 \in \mathcal{S}$ and $A_0 \in \mathcal{A}(S_0)$ s.t. all pairs have probability > 0
  Generate an episode starting from S_0, A_0, following π
  For each pair s, a appearing in the episode:
    $G \leftarrow$ return following the first occurrence of s, a
    Append G to $Returns(s, a)$
    $Q(s, a) \leftarrow \text{average}(Returns(s, a))$
  For each s in the episode:
    $\pi(s) \leftarrow \arg\max_a Q(s, a)$

每次试验的初始状态和动作都是随机选择的

策略评估

策略改进

无探索性初始化的MC控制

- 如何保证初始状态不变的同时（即取消exploring starts假设）访问到所有状态呢?
- 有两种方法：on–policy（同策略）方法和off–policy（异策略）方法。
- on-policy：是指产生数据的策略与评估和要改善的策略是同一个策略。
- off-policy：评估和改善的不是用来产生数据的policy，在 off-policy 方法中一般使用 2 个policy，一个学习并成为 optimal policy，另外一个则更偏向于探索，用来产生行为。

同策略与异策略

- 举个简单的例子：
  - 如果想要学习到一个确定性策略，比如利用greedy方法得到的策略，若直接用该策略进行采样并不会有探索机制，
  - 但是我们想要对各个动作和状态进行探索，那就可以使用一个随机策略进行采样，比如epsilon-greedy得到的策略。

同策略（on-policy）

- 在 on-policy 控制方法中，规则常常是 soft（柔性）的，即对所有的 $s \in S$ 与 $a \in A(s)$，均有 $\pi(a|s)>0$。
  - 即在任一状态下，所有的行为都有可能被选中。
- ε−greedy policies：
  - 在 1−ε 的概率下会选择当前具有最大 action value 估计值的行为，而在 1−ε 的概率下会随机从所有行为中随机选择一种 action，
  - 因此，选择 greedy action 的概率是 $1-\varepsilon+\varepsilon/|A(s)|$，而选择非 greedy action 的概率是 $\varepsilon/|A(s)|$。

$$\pi(a|s) \leftarrow \begin{cases} 1-\varepsilon+\varepsilon/|\mathcal{A}(s)| & \text{if } a = A^* \\ \varepsilon/|\mathcal{A}(s)| & \text{if } a \neq A^* \end{cases}$$

- ε-greedy policies 属于 ε-soft policies 中的一种，ε-greedy policies 是最接近 greedy 的ε-soft policies。

同策略首次访问的伪码

- On-policy first-visit MC的伪代码如下所示：

On-policy first-visit MC control (for ε-soft policies)

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:

  $Q(s,a) \leftarrow$ arbitrary

  $Returns(s,a) \leftarrow$ empty list

  $\pi(a|s) \leftarrow$ an arbitrary $\boxed{\varepsilon\text{-soft policy}}$

ε-soft策略

Repeat forever:

  (a) Generate an episode using π

  (b) For each pair s, a appearing in the episode:

    $G \leftarrow$ return following the first occurrence of s, a

    Append G to $Returns(s,a)$

    $Q(s,a) \leftarrow \text{average}(Returns(s,a))$

First-visit

策略评估

  (c) For each s in the episode:

    $A^* \leftarrow \arg\max_a Q(s,a)$

策略改进

    For all $a \in \mathcal{A}(s)$:

$$\pi(a|s) \leftarrow \begin{cases} 1 - \varepsilon + \varepsilon/|\mathcal{A}(s)| & \text{if } a = A^* \\ \varepsilon/|\mathcal{A}(s)| & \text{if } a \neq A^* \end{cases}$$

ε-soft策略

异策略

- 前面提到了探索的重要性。
- 强化学习中控制方式的目标是学习一系列的优化行为的action values，然而为了寻找优化的行为，它们不能总选择“当前”最优的行为，而需要探索所有的动作。
- 因此需要采用具有“探索精神”的 policy，并在探索的同时得到优化的 policy。
- off-policy 方法中一般使用 2 个policy，一个学习并成为 optimal policy，记作π，另外一个则更偏向于探索，用来产生行为，记作μ。

异策略

- 用来学习（评估并改进）的策略π(a|s)称为是target policy，
- 而用来产生行为的策略μ(a|s)称为是 behavior policy。
- 在这种机制下，学习来源于数据而“偏离”（off）了 target policy，因此将整个过程称为是off–policy learning。
- on-policy 相比之下更简单，off-policy 方法则需要更多的概念来表达，它常常有较大的方差且收敛较慢，但 off-policy 往往更强大，更具有一般性。

偏差与方差

- 在监督学习中，偏差(bias)和方差(variance)的可理解为欠拟合和过拟合。
  - 偏差大（欠拟合），预测值与样本之间的偏差较大，说明精度低；
  - 方差大（过拟合），样本值之间的方差大，说明样本的置信度较差，学出的模型适用性差。
- 不同的机器学习方法会在两者之间权衡。

异策略

- 用于异策略的目标策略π(a|s)和行动策略μ(a|s)并非任意选择的，而是必须满足一定的条件。
- 这个条件是**覆盖性条件**，即：为了利用从规则μ产生的episodes来评估π的value，则需要规则π下的所有行为在规则μ下被执行过。
- 也就是说：行动策略μ产生的行为覆盖或包含目标策略π产生的行为。
- 用式子表示即为：满足π(a|s)>0的任何(s,a)均满足μ(a|s)>0。

插播一点数学：重要性采样

- 重要性采样（Importance Sampling）是统计学中的一种采样方法。它主要用于一些难以直接采样的数据分布上。
- 假设有一个很复杂的概率密度函数p(x)，求解随机变量基于此概率下的某个函数期望，即：

$$E_{x\sim p(x)}[f(x)] \Rightarrow E_{x\sim p(x)}[f(x)] = \int_x p(x)f(x)\mathrm{d}x$$

- 如果 f(x) 的形式非常复杂，直接求解变得非常困难。当采样的数量足够大时，采样的样本就可以无限接近原分布，基于此思想，求解公式变为：

$$E_{x\sim p(x)}[f(x)] = \frac{1}{N}\sum_{x_i\sim p(x),i=1}^{N} f(x_i)$$

插播一点数学：重要性采样

- 当采样一些非常奇怪的概率分布，采样可能会遇到困难。这时重要性采样就派上用场了。
- 令待采样的分布为p(x)，另一个简单的定义域与p(x)相同的分布为$\tilde{p}$(x)，则可以得到：

$$\begin{aligned}
E_{x\sim p(x)}[f(x)] &= \int_x p(x)f(x)\mathrm{d}x \\
&= \int_x \tilde{p}(x)\frac{p(x)}{\tilde{p}(x)}f(x)\mathrm{d}x \\
&= E_{x\sim\tilde{p}(x)}[\frac{p(x)}{\tilde{p}(x)}f(x)] \\
&\simeq \frac{1}{N}\sum_{x_i\sim\tilde{p}(x),i=1}^{N}\frac{p(x)}{\tilde{p}(x)}f(x)
\end{aligned}$$

$$\Rightarrow \quad E[f]=\frac{1}{N}\sum_n \omega^n f\left(z^n\right)$$

插播一点数学：重要性采样

- 在重要性采样中，使用的采样概率分布与原概率分布越接近，方差越小。
- 但被积函数的概率分布往往很难求得，或很奇怪，没有简单地采样概率分布能与之相似，如果使用分布差别很大的采样概率对原概率分布进行采样，方差会趋近于无穷大。
- 一种减小重要性采样积分方差的方法是采用加权重要性采样：

$$E[f] \approx \sum_{n=1}^{N} \frac{\omega^n}{\sum_{m=1}^{N} \omega^m} f(z^n)$$

异策略与重要性采样

- 几乎所有的 off-policy 方法都使用重要性采样，
- 这里根据轨迹在 target policies 和 behavior policies 下发生的概率比来对 returns 赋予权重，称为 importance sampling ratio（重要性权重）。
- 给定初始状态 S_t，state-action 轨迹 $A_t, S_{t+1}, A_{t+1},\ldots,S_T$在规则 π下发生的概率为：

$$\prod_{k=t}^{T-1} \pi(A_t|S_t)p(S_{k+1}|S_k, A_k)$$

异策略与重要性采样

- 在异策略方法中，行动策略μ即用来产生样本的策略，所产生的轨迹概率分布相当于重要性采样中的$\tilde{p}(x)$，用来评估和改进的策略π所对应的轨迹概率分布为$p(x)$，
- 因此利用行动策略μ所产生的累积函数返回值来评估策略π时，需要在累积函数返回值前面乘以重要性权重。

策略π下，轨迹的概率

$$\rho_t^T = \frac{\prod_{k=t}^{T-1} \pi(A_k|S_k)\, p(S_{k+1}|S_k, A_k)}{\prod_{k=t}^{T-1} \mu(A_k|S_k)\, p(S_{k+1}|S_k, A_k)} = \prod_{k=t}^{T-1} \frac{\pi(A_k|S_k)}{\mu(A_k|S_k)}$$

策略μ下，轨迹的概率

普通重要性采样与加权重要性采样

无偏估计，方差无界

有偏估计，方差较小

- 普通重要性采样的值函数估计：

t之后的第一个终止时刻

从t到T(t)的回报值

$$V(s)=\frac{\sum_{t\in\mathcal{T}(s)}\rho_t^{T(t)}G_t}{|\mathcal{T}(s)|}$$

- 加权重要性采样的值函数估计：

$$V(s)=\frac{\Sigma_{t\in\mathcal{T}(s)}\rho_t^{T(t)}G_t}{\Sigma_{t\in\mathcal{T}(s)}\rho_t^{T(t)}}$$

$\rho_t^{T(t)}$：重要性权重 $=\prod_{k=t}^{T-1}\frac{\pi(A_k|S_k)}{\mu(A_k|S_k)}$

MC的增量式实现

- Monte Carlo方法的增量式实现分为两种：
- 对 on-policy MC方法，其增量式实现即为对returns求平均；
- 对增量实现的 off-policy MC方法还要再分为两类：
  - 普通重要性采样（ordinary importance sampling）中只是对 returns 赋予了权重$\rho_t^{T(t)}$，采用的还是简单平均的方法。
  - 因此，只需要考虑 off-policy Monte Carlo 方法中的加权重要性采样（weighted importance sampling）
  - 形式即可。

加权重要性采样的增量式实现

- 假设我们有 returns 序列 $G_1, G_2, \cdots, G_{n-1}$，它们均以同样的状态开始，并且每个具有的随机权重为 W_i（例如 $W_i = \rho_t^{T(t)}$），则希望估计的值为：

$$V_n \doteq \frac{\sum_{k=1}^{n-1} W_k G_k}{\sum_{k=1}^{n-1} W_k}, \quad n \geq 2$$

- 假设已经有了估计值 V_n，以及当前获得的 return 值 G_n，则下一步就是利用 V_n 与 G_n 来估计 V_{n+1}，此时还需要记录的值是前 n 个 returns 的 weights 的和，即分母的和，这样就可以得到更新方程：

$$V_{n+1} \doteq V_n + \frac{W_n}{C_n}\left[G_n - V_n\right], \quad n \geq 1 \qquad C_{n+1} \doteq C_n + W_{n+1}$$

异策略每次访问的MC预测

- 异策略每次访问的Mc预测方法如下：

Incremental off-policy every-visit MC policy evaluation

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:
　　$Q(s,a) \leftarrow$ arbitrary
　　$C(s,a) \leftarrow 0$
　　$\mu(a|s) \leftarrow$ an arbitrary soft behavior policy
　　$\pi(a|s) \leftarrow$ an arbitrary target policy

Repeat forever:
　　Generate an episode using μ:
　　　　$S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T, S_T$
　　$G \leftarrow 0$
　　$W \leftarrow 1$
　　For $t = T-1, T-2, \ldots$ downto 0:
　　　　$G \leftarrow \gamma G + R_{t+1}$
　　　　$C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
　　　　$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t,A_t)}[G - Q(S_t, A_t)]$
　　　　$W \leftarrow W \frac{\pi(A_t|S_t)}{\mu(A_t|S_t)}$
　　　　If $W = 0$ then ExitForLoop

注意：这里可以观察到R

累加W

加权重要性采样

异策略每次访问的MC控制

- 异策略每次访问的MC控制方法如下：

Off-policy every-visit MC control (returns $\pi \approx \pi_*$)

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:
  $Q(s,a) \leftarrow$ arbitrary
  $C(s,a) \leftarrow 0$
  $\pi(s) \leftarrow$ a deterministic policy that is greedy with respect to Q

确定性策略，因此π为1

Repeat forever:
  Generate an episode using any soft policy μ:
    $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T, S_T$
  $G \leftarrow 0$
  $W \leftarrow 1$
  For $t = T-1, T-2, \ldots$ downto 0:
    $G \leftarrow \gamma G + R_{t+1}$
    $C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
    $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t,A_t)}[G - Q(S_t, A_t)]$
    $\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$ (with ties broken consistently)
    If $A_t \neq \pi(S_t)$ then ExitForLoop
    $W \leftarrow W \frac{1}{\mu(A_t|S_t)}$

对每对(s,a)都需要获得很多次的 returns，这里策略μ为ε-soft的。

策略评估

策略改善

MC与DP的差异

- 蒙特卡罗夫方法的每一次experience，都是从一个初始状态（即根节点）开始，沿着某个特定episode 的转变轨迹遍历每一个经历过的节点，最终以终止状态结束。
- 针对某一个节点，MC 方法只包含该特定episode 选择的 action 的转换，而 DP 方法会将所有可能的转换都包含在内。从全局上来看，MC 方法包含了一个episode 经历的所有转换，而 DP 方法只包含一步转换过程。
- MP 方法的一个重要属性在于，它对每一个状态的估计是独立的，不依赖于对其他状态的估计。并且，MP 方法估计每一单一状态的 value 的计算成本是与状态的数量无关的。

MC法的优点

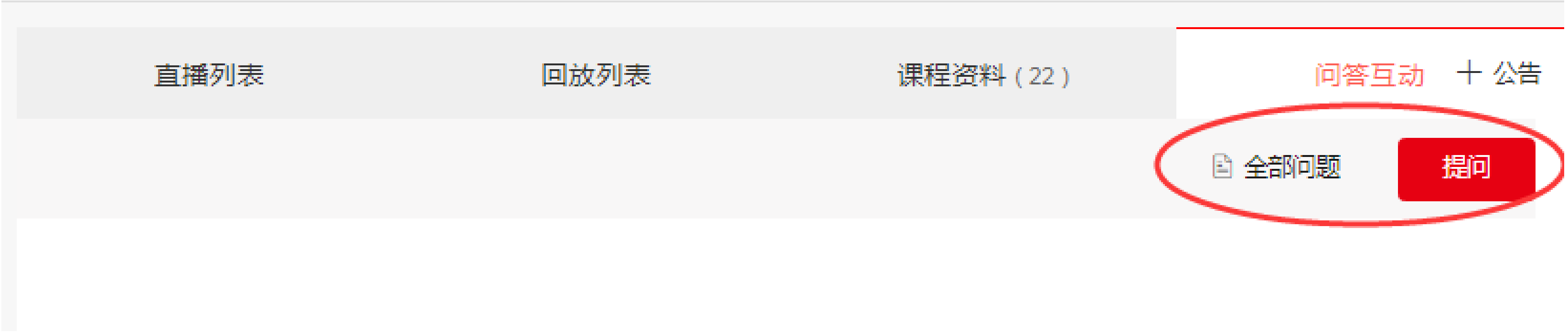
- MC直接从与环境的交互中进行学习，不需要环境的动态模型。
- MC可以利用仿真或者采样模型，在很多实际应用中，仿真构建 sample episodes 很容易，而构建 DP 所需要的转换概率的确切模型很难。
- 如果只希望得到一部分 states 的估计值，Monte Carlo 方法可以很容易的聚焦到这些 states 上，它只要不计算不关注的 states 即可，而 DP 方法中，对某个 state 的估计需要干涉到其他所有相关的 states，不如 Monte Carlo 方法简便。
- 当违背 Markov 属性时，Monte Carlo 方法受到的影响较小，因为它不需要基于后续状态的 value 估计值来更新 value 估计值，也就是说，它们不进行“自举”(bootstrap)。

参考资料

- https://blog.csdn.net/coffee_cream/article/details/66972281
- https://zhuanlan.zhihu.com/p/28107168
- https://zhuanlan.zhihu.com/p/25743759

问答互动

在所报课的课程页面，
1、点击“全部问题”显示本课程所有学员提问的问题。
2、点击“提问”即可向该课程的老师和助教提问问题。



联系我们

小象学院：互联网新技术在线教育领航者

- 微信公众号：**小象学院**