# 强化学习原理

## 第五章读书笔记

姓名：石若川 学号：2111381 专业：智能科学与技术

# 5 蒙特卡洛方法

蒙特卡洛方法仅仅需要经验，即从真实或者模拟的环境交互中采样得到的状态、动作、收益的序列。蒙特卡洛方法不需要关于环境动态变化规律的先验知识，却依然能够达到最优的行为。模型只需要能够生成状态转移的一些样本，而不需要像动态规划 (DP) 算法那样生成所有可能转移的概率分布。

蒙特卡洛算法通过平均样本的回报来解决强化学习问题。在分幕式任务中，假设一段经验可以被分为若干个幕，并且无论选取怎样的动作整个幕一定会终止。价值估计以及策略改进在整个幕结束时才进行。因此蒙特卡洛算法是逐幕做出改进的，而非在每一步 (在线) 都有改进。

蒙特卡洛算法与多臂赌博机类似，采样并平均每一个“状态-动作”二元组的回报。其主要的区别在于，每一个状态都类似于一个不同的赌博机问题，并且这些不同的赌博机问题是相互关联的。换言之，在某个状态采取动作之后的回报取决于在同一个幕内后来的状态中采取的动作。由于所有动作的选择都在随时刻演进而不断地学习，所以这个问题是非平稳的。

为了处理其非平稳性，采用了广义策略迭代 (GPI) 算法的思想，从马尔可夫决策过程采样样本的经验回报中学习价值函数。价值函数和其对应的策略依然以同样的方式 (GPI) 保证其最优性。

## 5.1 蒙特卡洛预测

**1. 蒙特卡洛方法的基础**

一个状态的价值是从该状态开始的期望回报，即未来的折扣收益累积值的期望。那么一个方法是根据经验进行估计，即对所有经过这个状态之后产生的回报进行平均。随着越来越多的回报被观察到，平均值就会收敛于期望值。这一想法是所有蒙特卡洛算法的基础。

**2. first-visit 型和 every-visit 型 MC 算法**

假设给定在策略 $\pi$ 下途经状态 $s$ 的多幕数据，我们想估计策略 $\pi$ 下状态 $s$ 的价值函数 $v_\pi(s)$。在给定的某一幕中，每次状态 $s$ 的出现都称为对 $s$ 的一次访问。我们称第一次访问为 $s$ 的首次访问。首次访问 (first-visit) 型 MC 算法用 $s$ 的所有首次访问的回报的平均值估计 $v_\pi(s)$，而每次访问 (every-visit) 型 MC 算法则使用所有访问的回报的平均值。

当 s 的访问次数 (或首次访问次数) 趋向无穷时，first-visit 型 MC 和每次访问型 MC 均会收敛到 $v_\pi(s)$。算法中的每个回报值都是对 $v_\pi(s)$ 的一个独立同分布的估计，且估计的方差是有限的。根据大数定理，这一平均值的序列会收敛到它们的期望值。每次平均都是一个无偏估计，其误差的标准差以 $1/\sqrt{n}$ 衰减，这里的 $n$ 是被平均的回报值的个数。

first-visit 型 MC 算法，用于估计 $V \approx v_\pi$

输入：待评估的策略 $\pi$
初始化：
对所有 $s \in \mathcal{S}$, 任意初始化 $V(s) \in \mathbb{R}$
　对所有 $s \in \mathcal{S}$, $Returns(s) \leftarrow$ 空列表
无限循环 (对每幕):
　根据 $\pi$ 生成一幕序列：$S_0, A_0, R_1, S_1, A_1, R_2, \ldots, S_{T-1}, A_{T-1}, R_T$
　$G \leftarrow 0$
　对本幕中的每一步进行循环，$t = T-1, T-2, \ldots, 0$ :
　　$G \leftarrow \gamma G + R_{t+1}$
　　除非 $S_t$ 在 $S_0, S_1, \ldots, S_{t-1}$ 中已出现过：
　　将 G 加入 $Returns(S_t)$
　　$V(S_t) \leftarrow \text{average}(Returns(S_t))$

**3. 回溯图思想用于蒙特卡洛方法**

如图5.1所示，在蒙特卡洛算法中估计时，根为一个状态节点，然后往下是某幕样本序列一直到终止状态为止的完整轨迹，其中包含该幕中的全部状态转移。DP 的回溯图显示了所有可能的转移，而蒙特卡洛算法则仅仅显示在当前幕中采样到的那些转移。

蒙特卡洛算法对于每个状态的估计是独立的。它对于一个状态的估计完全不依赖于对其他状态的估计，这与 DP 完全不同。换言之蒙特卡洛算法并没有使用自举思想。

图 5.1: 回溯图

## 5.2 动作价值的蒙特卡洛估计

如果无法得到环境的模型，那么计算动作的行为值函数比起计算状态的值函数更加有用一些。必须通过显式地确定每个动作的价值函数来确定一个策略。

行为值函数的策略评估问题的目标就是估计 $q_\pi(s,a)$ ，即在策略 $\pi$ 下从状态 $s$ 采取动作 $a$ 的期望回报。只需将对状态的访问改为对“状态-动作”二元组的访问，蒙特卡洛算法就可以用几乎和之前完全相同的方式来解决此问题。

**1. first-visit 型和 every-visit 型 MC 算法**

如果在某一幕中状态 $s$ 被访问并在这个状态中采取了动作 $a$, 我们就称“状态-动作”二元组 $(s,a)$ 在这一幕中被访问到。每次访问（every-visit）型 MC 算法将所有“状态-动作”二元组得到的回报的平均值作为价值函数的近似；而首次访问（first-visit）型 MC 算法则将每幕第一次在这个状态下采取这个动作得到的回报的平均值作为价值函数的近似。和之前一样，在对每个“状态-动作”二元组的访问次数趋向无穷时，这些方法都会二次收敛到行为值函数的真实期望值。

**2. 探索性初始化和非零概率的策略**

问题：一些“状态-动作”二元组可能永远不会被访问到。如果 $\pi$ 是一个确定性的策略，那么遵循 $\pi$ 意味着在每一个状态中只会观测到一个动作的回报。在无法获取回报进行平均的情况下，蒙特卡洛算法将无法根据经验改善动作价值函数的估计。

解决：

- 探索性初始化：将指定的“状态-动作”二元组作为起点开始一幕采样，同时保证所有“状态-动作”二元组都有非零的概率可以被选为起点。这样就保证了在采样的幕个数趋向于无穷的时候，每一个“状态-动作”二元组都会被访问到无数次。

- 非零概率策略：只考虑那些在每个状态下所有动作都有非零概率被选中的随机策略。

## 5.3 蒙特卡洛控制

**1. 蒙特卡洛方法下的策略迭代**

蒙特卡洛方法方法从任意的策略 $\pi_0$ 开始交替进行完整的策略评估和策略改进，最终得到最优的策略和动作价值函数

$$\pi_0 \xrightarrow{E} q_{\pi_0} \xrightarrow{1} \pi_1 \xrightarrow{E} q_{\pi_1} \xrightarrow{1} \pi_2 \xrightarrow{E} \cdots \xrightarrow{1} \pi_* \xrightarrow{E} q_*,$$

$\xrightarrow{E}$ 表示策略评估，而 $\xrightarrow{1}$ 表示策略改进。

策略评估完全按照前一节所述的方法进行。假设观测到了无限多幕的序列，并且这些幕保证了探索性初始化假设。在这种情况下，对于任意的 $\pi_k$，蒙特卡洛算法都能精确地计算对应的 $q_{\pi_k}$。

策略改进的方法是在当前价值函数上贪心地选择动作。对于任意的一个动作价值函数 $q$，对应的贪心策略为：对于任意一个状态 $s \in \mathcal{S}$，选择对应动作价值函数最大的动作

$$\pi(s) \doteq \arg\max_a q(s,a)$$

策略改进可以通过将 $q_{\pi_k}$ 对应的贪心策略作为 $\pi_{k+1}$ 来进行。这样的 $\pi_k$ 和 $\pi_{k+1}$ 满足策略改进定理，因为对所有的状态 $s \in \mathcal{S}$

$$\begin{aligned} q_{\pi_k}\left(s, \pi_{k+1}(s)\right) &= q_{\pi_k}(s, \operatorname*{argmax}_a q_{\pi_k}(s,a)) \\ &= \max_a q_{\pi_k}(s,a) \\ &\geqslant q_{\pi_k}(s, \pi_k(s)) \\ &\geqslant v_{\pi_k}(s) \end{aligned}$$

**2. 确保蒙特卡洛方法收敛的假设**

上文提出了两个很强的假设来保证蒙特卡洛算法的收敛。一个是探索型初始化假设，另一个是在进行策略评估的时候有无限多幕的样本序列进行试探。为了得到一个实际可用的算法，必须去除这两个假设。

首先先讨论在进行策略评估时可以观测到无限多幕样本序列这一假设。事实上，经典 DP 算法的结果也仅仅是渐近地收敛于真实的价值函数。无论是 DP 还是蒙特卡洛算法，有两个方法可以解决这一问题。

- 一种方法是想方设法在每次策略评估中对 $q_{\pi_k}$ 做出尽量好的逼近这就需要做一些假设并定义一些测度，来分析逼近误差的幅度和出现概率的上下界，然后采取足够多的步数来保证这些界足够小。这种方法可以保证收敛到令人满意的近似水平。

- 第二种避免无限多幕样本序列假设的方法是不再要求在策略改进前就完成策略评估。在每一个评估步骤中，我们让动作价值函数逼近 $q_{\pi_k}$，但我们并不期望它在经过很多步之前非常接近真实的值。这种思想的一种极端实现形式就是价值迭代，即在相邻的两步策略改进中只进行一次策略评

估，而不要求多次迭代后的收敛。价值迭代的“就地更新”版本则更加极端：在单个状态中交替进行策略的改进与评估。

**3. 基于探索性初始化的蒙特卡洛**

对于蒙特卡洛策略迭代，自然可以逐幕交替进行评估与改进。每一幕结束后，使用观测到的回报进行策略评估，然后在该幕序列访问到的每一个状态上进行策略的改进。使用这个思路的算法，称作基于探索性初始化的蒙特卡洛 (蒙特卡洛 ES)，下框中给出了这个算法。

蒙特卡洛 ES，用于估计 $V \approx v_\pi$

初始化：
  对所有 $s \in \mathcal{S}$, 任意初始化
  $\pi(s) \in \mathcal{A}(s)$
  对所有 $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$, 任意初始化 $Q(s, \alpha) \in \mathbb{R}$
  对所有 $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$, $Returns(s, a) \leftarrow$ 空列表
无限循环 (对每幕):
  选择 $S_0 \in \mathcal{S}$ 和 $A_0 \in \mathcal{A}(S_0)$ 以使得所有“状态-动作”二元组的概率都 $>0$
  从 $S_0, A_0$ 开始根据 $\pi$ 生成一幕序列：$S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$
  $G \leftarrow 0$
  对幕中的每一步循环，$t = T-1, T-2, \ldots, 0$ :
    $G \leftarrow \gamma G + R_{t+1}$
    除非二元组 $S_t, A_t$ 在 $S_0, A_0, S_1, A_1 \ldots, S_{t-1}, A_{t-1}$ 中已出现过：
      将 $G$ 加入 $Returns(S_t, A_t)$
      $Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$
      $\pi(S_t) \leftarrow \text{argmax}_a\, Q(S_t, a)$

在蒙特卡洛 ES 中，无论之前遵循的是哪一个策略，对于每一个“状态-动作”二元组的所有回报都被累加并平均。可以很容易发现，蒙特卡洛 ES 不会收敛到任何一个次优策略。因为如果真的收敛到次优策略，其价值函数一定会收敛到该策略对应的价值函数，而在这种情况下还会得到一个更优的策略。只有在策略和价值函数都达到最优的情况下，稳定性才能得到保证。因为动作价值函数的变化随着时间增加而不断变小，所以应该一定能收敛到这个不动点。

## 5.4 没有探索性初始化假设的蒙特卡洛控制

**1. on-policy 方法和 off-policy 方法**

能够避免探索性初始化假设的唯一一般性解决方案就是，智能体能够持续不断地选择所有可能的动作。有两种方法可以保证这一点，分别被称为同轨策略 (on-policy) 方法和离轨策略 (off-policy) 方法。

- on-policy：用于生成采样数据序列的策略和用于实际决策的待评估和改进的策略是相同的
- off-policy：用于评估或者改进的策略与生成采样数据的策略是不同的，即生成的数据“离开”了待优化的策略所决定的决策序列轨迹。

在 on-policy 策略方法中，策略一般是“软性”的，即对于任意 $s \in S$ 以及 $a \in \mathcal{A}(s)$ ，都有 $\pi(a|s) > 0$，但它们会逐渐地逼近一个确定性的策略。其中之一是 $\epsilon - greedy$ 策略，意思是在绝大多数时候都采取

获得最大估计值的动作价值函数所对应的动作，但同时以一个较小的 $\epsilon$ 概率随机选择一个动作。即对于所有的非贪心动作，都有 $\frac{\varepsilon}{|A(s)|}$ 的概率被选中，剩余的大部分的概率为 $1-\varepsilon+\frac{\varepsilon}{|A(s)|}$，这是选中贪心动作的概率。这种 $\epsilon-greedy$ 策略是 $\epsilon-soft$ 策略的一个例子，即对某个 $\varepsilon>0$，所有的状态和动作都有 $\pi(a|s)\geqslant\frac{\varepsilon}{|A(s)|}$。在所有 $\epsilon-soft$ 策略中，$\epsilon-greedy$ 策略在某种层面上是最接近贪心策略的。

**2. on-policy 策略算法**

on-policy 策略算法的蒙特卡洛控制的总体思想依然是 GPI。如同蒙特卡洛 ES 一样，使用 first-visit 型 MC 算法来估计当前策略的动作价值函数。然而，由于缺乏试探性出发假设，不能简单地通过对当前价值函数进行贪心优化来改进策略，否则就无法进一步试探非贪心的动作。但是，GPI 并不要求优化过程中所遵循的策略一定是贪心的，只需要它逐渐逼近贪心策略即可。在 on-policy 策略算法中，仅仅改为遵循 $\epsilon-greedy$ 策咯。对于任意一个 $\epsilon-soft$ 策略 $\pi$, 根据 $q_\pi$ 生成的任意一个 $\epsilon-greedy$ 策略保证优于或等于 $\pi$ 。

on-policy 策略的 first-visit 型 MC 控制算法（对于 $\epsilon-soft$ 策略），用于估计 $V\approx v_\pi$

算法参数：很小的 $\varepsilon>0$

初始化：

$\pi\leftarrow$ 一个任意的 $\epsilon-soft$ 策略

对所有 $s\in\mathcal{S},a\in\mathcal{A}(s)$, 任意初始化 $Q(s,a)\in\mathbb{R}$

对所有 $s\in\mathcal{S},\ a\in\mathcal{A}(s),\ Returns(s,a)\leftarrow$ 空列表

无限循环 (对每幕):

根据 $\pi$ 生成一幕序列：$S_0,A_0,R_1,\ldots,S_{T-1},A_{T-1},R_T$

$G\leftarrow 0$

对幕中的每一步循环，$t=T-1,T-2,\ldots,0$

$G\leftarrow\gamma G+R_{t+1}$

除非“状态-动作”二元组 $S_t,A_t$ 在 $S_0,A_0,S_1,A_1\ldots,S_{t-1},A_{t-1}$ 已出现过：

把 G 加入 $Returns(S_t,A_t)$

$Q(S_t,A_t)\leftarrow$average$(Returns(S_t,A_t))$

$A^*\leftarrow\text{argmax}_a\,Q(S_t,a)$（有多个最大值时任意选取）

对所有 $a\in\mathcal{A}(S_t)$:

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

**3. $\epsilon-greedy$ 策略的策略改进定理证明**

根据策略改进定理，对于一个 $\epsilon-soft$ 策略 $\pi$，任何一个根据 $q_\pi$ 生成的 $\epsilon-greedy$ 策略都是对其的一个改进。假设 $\pi'$ 是一个 $\epsilon-greedy$ 策略。策略改进定理成立，因为对任意的 $s\in\mathcal{S}$

$$
\begin{aligned}
q_\pi(s,\pi'(s)) &= \sum_a \pi'(a|s) q_\pi(s,a) \\
&= \frac{\varepsilon}{|\mathcal{A}(s)|}\sum_a q_\pi(s,a) + (1-\varepsilon)\max_a q_\pi(s,a) \\
&\geqslant \frac{\varepsilon}{|\mathcal{A}(s)|}\sum_a q_\pi(s,a) + (1-\varepsilon)\sum_a \frac{\pi(a|s) - \frac{\varepsilon}{|\mathcal{A}(s)|}}{1-\varepsilon} q_\pi(s,a) \\
&= \frac{\varepsilon}{|\mathcal{A}(s)|}\sum_a q_\pi(s,a) - \frac{\varepsilon}{|\mathcal{A}(s)|}\sum_a q_\pi(s,a) + \sum_a \pi(a|s) q_\pi(s,a) \\
&= v_\pi(s)
\end{aligned} \tag{1}
$$

所以，根据策略改进定理，$\pi' \geqslant \pi$，即对于任意的 $s \in S$，满足 $v_{\pi'}(s) \geqslant v_\pi(s)$。

下面证明该式的等号成立的条件是：当且仅当 $\pi'$ 和 $\pi$ 都为最优的 $\epsilon - soft$ 策略，即它们都比所有其他的 $\epsilon - soft$ 策略更优或相同。

设想一个与原先环境几乎相同的新环境，唯一的区别是我们将策略是 $\epsilon - soft$ 这一要求“移入”环境中。新的环境与旧的环境有相同的动作与状态集，并表现出如下的行为。在状态 $s$ 采取动作 $a$ 时，新环境有 $1-\varepsilon$ 的概率与旧环境的表现完全相同。然后以 $\varepsilon$ 的概率重新等概率选择一个动作，然后有和旧环境采取这一新的随机动作一样的表现。在这个新环境中的最优策略的表现和旧环境中最优的 $\epsilon - soft$ 策略的表现是相同的。令 $\widetilde{v}_*$ 和 $\widetilde{q}_*$ 为新环境的最优价值函数，则当且仅当 $v_\pi = \widetilde{v}_*$ 时，策略 $\pi$ 是一个最优的 $\epsilon - soft$ 策略。根据 $\widetilde{v}_*$ 的定义，它是下式的唯一解

$$
\begin{aligned}
\widetilde{v}_*(s) &= (1-\varepsilon)\max_a \widetilde{q}_*(s,a) + \frac{\varepsilon}{|\mathcal{A}(s)|}\sum_a \widetilde{q}_*(s,a) \\
&= (1-\varepsilon)\max_a \sum_{s',r} p(s',r|s,a)\Big[r + \gamma \widetilde{v}_*(s')\Big] + \frac{\varepsilon}{|\mathcal{A}(s)|}\sum_a\sum_{s',r} p(s',r|s,a)\Big[r + \gamma \widetilde{v}_*(s')\Big]
\end{aligned} \tag{2}
$$

当等式成立且 $\epsilon - soft$ 策略 $\pi$ 无法再改进的时候，由公式1我们有

$$
\begin{aligned}
v_\pi(s) &= (1-\varepsilon)\max_a q_\pi(s,a) + \frac{\varepsilon}{|\mathcal{A}(s)|}\sum_a q_\pi(s,a) \\
&= (1-\varepsilon)\max_a \sum_{s',r} p(s',r|s,a)\Big[r + \gamma v_\pi(s')\Big] \\
&+ \frac{\varepsilon}{|\mathcal{A}(s)|}\sum_a\sum_{s',r} p(s',r|s,a)\Big[r + \gamma v_\pi(s')\Big]
\end{aligned} \tag{3}
$$

然而，除了下标为 $v_\pi$ 而不是 $\widetilde{v}$。以外，这个等式和上一个完全相同。由于 $\widetilde{v}_*$ 是唯一解，因此一定有 $v_\pi = v_*$。