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

5.5 基于重要度采样的 off-policy 策略

1. off-policy 策略

off-policy 策略包括两个策略：

- 目标策略：用来学习的策略，最终成为最优策略
- 行动策略：用于生成行动样本的策略，更具有试探性

假设希望预测 v_π 或 q_π，但是只有遵循另一个策略 $b\,(b \neq \pi)$ 所得到的若干幕样本。在这种情况下，π 是目标策略，b 是行动策略，两个策略都固定且已知。

2. 覆盖性假设

为了使用从 b 得到的多幕样本序列去预测 π，要求在 π 下发生的每个动作都至少偶尔能在 b 下发生。换句话说，对任意 $\pi(a|s) > 0$ ，要求 $b(a|s) > 0$。称其为覆盖性假设。

根据这个假设，在与 π 不同的状态中，b 必须是随机的。另一方面，目标策略 π 则可能是确定性的。在控制过程中，目标策略通常是一个确定性的贪心策略，它由动作价值函数的当前估计值所决定。而当行动策略是随机的且具有试探性时 (例如可使用 $\epsilon - greedy$ 策略)，这个策略会成为一个确定性的最优策略。

3. 重要度采样

重要度采样是一种在给定来自其他分布的样本的条件下，估计某种分布的期望值的通用方法。将重要度采样应用于 off-policy 策略学习，对回报值根据其轨迹在目标策略与行动策略中出现的相对概率进行加权，这个相对概率也被称为重要度采样比。给定起始状态 S_t ，后续的状态-动作轨迹 $A_t, S_{t+1}, A_{t+1}, \ldots, S_T$ 在策略 π 下发生的概率是

$$\begin{aligned}
&\Pr\{A_t, S_{t+1}, A_{t+1}, \ldots, S_T \mid S_t, A_{t:T-1} \sim \pi\} \\
&= \pi(A_t|S_t)p(S_{t+1}|S_t, A_t)\pi(A_{t+1}|S_{t+1}) \cdots p(S_T|S_{T-1}, A_{T-1}) \\
&= \prod_{k=t}^{T-1} \pi(A_k|S_k)p(S_{k+1}|S_k, A_k),
\end{aligned}$$

这里，p 是状态转移概率函数。因此，在目标策略和行动策略轨迹下的相对概率 (重要度采样比) 是

$$\rho_{t:T-1} \doteq \frac{\prod_{k=t}^{T-1} \pi(A_k|S_k)p(S_{k+1}|S_k, A_k)}{\prod_{k=t}^{T-1} b(A_k|S_k)p(S_{k+1}|S_k, A_k)} = \prod_{k=t}^{T-1} \frac{\pi(A_k|S_k)}{b(A_k|S_k)} \tag{1}$$

从行动策略中得到的回报的期望 $\mathbb{E}[G_t|S_t = s] = v_b(s)$ 是不准确的，所以不能用它们的平均来得到 v_π。使用比例系数 $\rho_{t:T-1}$ 可以调整回报使其有正确的期望值

$$\mathbb{E}[\rho_{t:T-1}G_t \mid S_t = s] = v_\pi(s). \tag{2}$$

4. 普通重要度采样和加权重要度采样

下面给出通过观察到的一批遵循策略 b 的多幕采样序列并将其回报进行平均来预测 $v_\pi(s)$ 的蒙特卡洛算法。为了方便起见，在这里对时刻进行编号时，即使时刻跨越幕的边界，编号也递增。也就是说，如果这批多幕采样序列中的第一幕在时刻 100 时在某个终止状态结束，下一幕就起始于 $t = 101$。这样就可以使用唯一的时刻编号来指代特定幕中的特定时刻。特别地，对于每次访问型方法，可以定义所有访问过状态 s 的时刻集合为 $\mathcal{T}(s)$。而对于首次访问型方法，$\mathcal{T}(s)$ 只包含在幕内首次访问状态 s 的时刻。用 $T(t)$ 来表示在时刻 t 后的首次终止，用 G_t 来表示在 t 之后到达 $T(t)$ 时的回报值。那么 $\{G_t\}_{t\in\mathcal{T}(s)}$ 就是状态 s 对应的回报值，$\{\rho_{t:T(t)-1}\}_{t\in\mathcal{T}(s)}$ 是相应的重要度采样比。为了预测 $v_\pi(s)$，只需要根据重要度采样比来调整回报值并对结果进行平均即可

$$V(s) \doteq \frac{\sum_{t\in\mathcal{T}(s)} \rho_{t:T(t)-1}G_t}{|\mathcal{T}(s)|} \tag{3}$$

通过这样一种简单平均实现的重要度采样被称为普通重要度采样。

另一个重要的方法是加权重要度采样，它采用一种加权平均的方法，其定义为

$$V(s) \doteq \frac{\sum_{t\in\mathcal{T}(s)} \rho_{t:T(t)-1}G_t}{\sum_{t\in\mathcal{T}(s)} \rho_{t:T(t)-1}} \tag{4}$$

两种重要度采样方法的区别：

两种重要度采样算法在首次访问型方法下的差异可以用偏差与方差来表示。普通重要度采样的估计是无偏的，而加权重要度采样的估计是有偏的（偏差值渐近收敛到零）。另一方面，普通重要度采样的方差一般是无界的，因为重要度采样比的方差是无界的。而在加权估计中任何回报的最大权值都是 1。如果假设回报值有界，那么即使重要度采样比的方差是无穷的，加权重要度采样估计的方差仍能收敛到零。

5.6 增量式实现

1. 普通重要度采样

假设有一个回报序列 $G_1, G_2, \cdots, G_{n-1}$，它们都从相同的状态开始，且每一个回报都对应一个随机权重 W_i（例如，$W_i = \rho_{t:T(t)-1}$）。希望得到如下的估计，并且在获得了一个额外的回报值 G_n 时能保持更新。

$$V_n \doteq \frac{\sum_{k=1}^{n-1} W_k G_k}{\sum_{k=1}^{n-1} W_k}, \quad n \geqslant 2 \tag{5}$$

V_n 的更新方法是

$$V_{n+1} \doteq V_n + \frac{W_n}{C_n}\Big[G_n - V_n\Big], \quad n \geqslant 1 \tag{6}$$

以及

$$C_{n+1} \doteq C_n + W_{n+1},$$

这里 $C_0 \doteq 0$（V_1 是任意的，所以不用特别指定）。下框给出了完整的用于蒙特卡洛评估的逐幕增量算法。

off-policy 策略 MC 预测算法（策略评估），用于估计 $Q \approx q\pi$

输入：一个任意的目标策略 π
初始化，对所有 $s \in \mathcal{S}, a \in \mathcal{A}(s)$：
  任意初始化 $Q(s,a) \in \mathbb{R}$
  $C(s,a) \leftarrow 0$
无限循环（对每幕）：
  $b \leftarrow$ 任何能包括 π 的策略　根据 b 生成一幕序列：$S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$
  $G \leftarrow 0$
  $W \leftarrow 1$
  对幕中的每一步循环，$t = T-1, T-2, \ldots, 0$, 当 $W \neq 0$ 时：
    $G \leftarrow \gamma G + R_{t+1}$
    $C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
    $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)} [G - Q(S_t, A_t)]$
    $W \leftarrow W \frac{\pi(A_t|S_t)}{b(A_t|S_t)}$
    如果 $W = 0$，则退出内层循环

5.7 off-policy 策略蒙特卡洛控制

off-policy 策略蒙特卡洛控制方法遵循行动策略并对目标策略进行学习和改进。为试探所有可能性，要求行为策略时软性的（在所有状态下选择动作的概率都非零）。下框展示了一个基于 GPI 和重要度采样的 off-policy 蒙特卡洛控制方法。目标策略 $\pi \approx \pi_*$ 是对应 Q 得到的贪心策略，这里 Q 是对 q_π 的估计。行动策略 b 可以是任何策略，但为了保证 π 能收敛到一个最优的策略，对每一个“状态-动作”二元组都需要取得无穷多次回报，这可以通过选择 $\epsilon - soft$ 的 b 来保证。

off-policy 策略 MC 控制算法，用于估计 $\pi \approx \pi_*$

初始化，对所有 $s \in \mathcal{S}$，　$a \in \mathcal{A}(s)$
  $Q(s,a) \in \mathbb{R}$ (任意值)
  $C(s,a) \leftarrow 0$
  $\pi(s) \leftarrow \arg\max_a Q(s,a)$ (出现平分的情况下选取方法应保持一致)
无限循环 (对每幕):
  $b \leftarrow$ 任意软性策略
  根据 b 生成一幕数据：$S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$
  $G \leftarrow 0$
  $W \leftarrow 1$
  对幕中的每一时刻循环，$t = T-1, T-2, \ldots, 0$:
    $G \leftarrow \gamma G + R_{t+1}$
    $C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
    $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)} [G - Q(S_t, A_t)]$
    $\pi(S_t) \leftarrow \text{argmax}_a Q(S_t, a)$ (出现平分的情况下选取方法应保持一致)
    如果 $A_t \neq \pi(S_t)$ 那么退出内层循环 (处理下一幕数据)
    $W \leftarrow W \frac{1}{b(A_t|S_t)}$

潜在问题：

当幕中某时刻后剩下的所有动作都是贪心的时候，这种方法也只会从幕的尾部进行学习。如果非贪心的行为较为普遍，则学习的速度会很慢，尤其是对于那些在很长的幕中较早出现的状态就更是如此。这可能会极大地降低学习速度。

解决：

时序差分学习 & 设置 $\gamma < 1$

5.8 折扣敏感的重要度采样

1. 普通重要性采样的问题

考虑一种幕很长且 γ 显著小于 1 的情况。假设幕持续 100 步并且 $\gamma = 0$。那么 0 时刻的回报就会是 $G_0 = R_1$, 但它的重要度采样比却会是 100 个因子之积，即 $\frac{\pi(A_0|S_0)}{b(A_0|S_0)}\frac{\pi(A_1|S_1)}{b(A_1|S_1)}\cdots\frac{\pi(A_{99}|S_{99})}{b(A_{99}|S_{99})}$。在普通重要度采样中会用整个乘积对回报进行缩放，但实际上只需要按第一个因子来衡量，即 $\frac{\pi(A_0|S_0)}{b(A_0|S_0)}$。另外 99 个因子：$\frac{\pi(A_1|S_1)}{b(A_1|S_1)}\cdots\frac{\pi(A_{99}|S_{99})}{b(A_{99}|S_{99})}$ 都是无关的，因为在得到首次收益之后，整幕回报就已经决定了。后面的这些因子与回报相独立且期望值为 1, 不会改变预期的更新，但它们会极大地增加其方差。在某些情况下，它们甚至可以使得方差无穷大。

2. 折扣敏感的重要度采样

这个思路的本质是把折扣看作幕终止的概率，或者说，部分终止的程度。对于任何 $\gamma \in [0\,1)$, 可以把回报 G_0 看作在一步内，以 $1-\gamma$ 的程度部分终止，产生的回报仅仅是首次收益 R_1，然后再在两步后以 $(1-\gamma)\gamma$ 的程度部分终止，产生 $R_1 + R_2$ 的回报，依此类推。后者的部分终止程度对应于第二步的终止程度 $1-\gamma$，以及在第一步尚未终止的 γ。因此，第三步的终止程度是 $(1-\gamma)\gamma^2$, 其中 γ^2 对应的是在前两步都不终止。这里的部分回报被称为平价部分回报

$$\bar{G}_{t:h} \doteq R_{t+1} + R_{t+2} + \cdots + R_h, \quad 0 \leqslant t < h \leqslant T,$$

其中“平价”表示没有折扣，“部分”表示这些回报不会一直延续到终止，而在 h 处停止，h 被称为视界 (T 是幕终止的时间)。传统的全回报 G_t 可被看作上述平价部分回报的总和

$$\begin{aligned}
\mathrm{Gt} \doteq\ & R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots + \gamma^{T-t-1} R_T \\
=\ & (1-\gamma) R_{t+1} \\
& + (1-\gamma)\gamma\,(R_{t+1} + R_{t+2}) \\
& + (1-\gamma)\gamma^2\,(R_{t+1} + R_{t+2} + R_{t+3}) \\
& + (1-\gamma)\gamma^{T-t-2}\,(R_{t+1} + R_{t+2} + \cdots + R_{T-1}) \\
& + \gamma^{T-t-1}\,(R_{t+1} + R_{t+2} + \cdots + R_T) \\
=\ & (1-\gamma)\sum_{h=t+1}^{T-1} \gamma^{h-t-1}\bar{G}_{t:h} + \gamma^{T-t-1}\bar{G}_{t:T}.
\end{aligned}$$

由于 $\bar{G}_{t:h}$ 只涉及到视界 h 为止的收益，因此只需要使用到 h 为止的概率值。定义了如下普通重要度采样估计器，它是式3的推广

$$V(s) \doteq \frac{\sum_{t\in\mathcal{T}(s)}\left((1-\gamma)\sum_{h=t+1}^{T(t)-1}\gamma^{h-t-1}\rho_{t:h-1}\bar{G}_{t:h} + \gamma^{T(t)-t-1}\rho_{t:T(t)-1}\bar{G}_{t:T(t)}\right)}{|\mathcal{T}(s)|} \tag{7}$$

以及如下的加权重要度采样估计器，它是式4的推广

$$V(s) \doteq \frac{\sum_{t\in\mathcal{T}(s)}\left((1-\gamma)\sum_{h=t+1}^{T(t)-1}\gamma^{h-t-1}\rho_{t:h-1}\bar{G}_{t:h}+\gamma^{T(t)-t-1}\rho_{t:T(t)-1}\bar{G}_{t:T(t)}\right)}{\sum_{t\in\mathcal{T}(s)}\left((1-\gamma)\sum_{h=t+1}^{T(t)-1}\gamma^{h-t-1}\rho_{t:h-1}+\gamma^{T(t)-t-1}\rho_{t:T(t)-1}\right)} \tag{8}$$

这两个估计器都考虑了折扣率，但当 $\gamma=1$ 时没有任何影响。

5.9 每次决策型重要度采样

在 off-policy 策略估计器3和4中，分子中求和计算中的每一项是一个求和式

$$\begin{aligned}\rho_{t:T-1}G_t &= \rho_{t:T-1}\left(R_{t+1}+\gamma R_{t+2}+\cdots+\gamma^{T-t-1}R_T\right)\\ &= \rho_{t:T-1}R_{t+1}+\gamma\rho_{t:T-1}R_{t+2}+\cdots+\gamma^{T-t-1}\rho_{t:T-1}R_T\end{aligned} \tag{9}$$

式9中的每个子项是一个随机收益和一个随机重要度采样比的乘积，例如可以将第一个子项写为

$$q\rho_{t:T-1}R_{t+1}=\frac{\pi(A_t|S_t)}{b(A_t|S_t)}\frac{\pi(A_{t+1}|S_{t+1})}{b(A_{t+1}|S_{t+1})}\frac{\pi(A_{t+2}|S_{t+2})}{b(A_{t+2}|S_{t+2})}\cdots\frac{\pi(A_{T-1}|S_{T-1})}{b(A_{T-1}|S_{T-1})}R_{t+1}$$

可以发现，在所有因子中，只有第一个和最后一个是相关的，其他所有比率均为期望值为 1 的独立随机变量

$$\mathbb{E}\left[\frac{\pi(A_k|S_k)}{b(A_k|S_k)}\right]\doteq\sum_a b(a|S_k)\frac{\pi(a|S_k)}{b(a|S_k)}=\sum_a\pi(a|S_k)=1$$

因此，由于独立随机变量乘积的期望是变量期望值的乘积，所以就可以把除了第一项以外的所有比率移出期望，只剩下

$$\mathbb{E}[\rho_{t:T-1}R_{t+1}]=\mathbb{E}[\rho_{t:t}R_{t+1}]$$

如果对式9的第 k 项重复这样的分析，就会得到

$$\mathbb{E}[\rho_{t:T-1}R_{t+k}]=\mathbb{E}[\rho_{t:t+k-1}R_{t+k}]$$

这样，式9的期望就可以写成

$$\mathbb{E}[\rho_{t:T-1}G_t]=\mathbb{E}\left[\tilde{G}_t\right]$$

其中

$$\tilde{G}_t=\rho_{t:t}R_{t+1}+\gamma\rho_{t:t+1}R_{t+2}+\gamma^2\rho_{t:t+2}R_{t+3}+\cdots+\gamma^{T-t-1}\rho_{t:T-1}R_T$$

这种思想称为每次决策型重要度采样，使用 $\tilde{G}_t$ 来计算与普通重要采样估计器相同的无偏期望，以减小估计器的方差

$$V(s)\doteq\frac{\sum_{t\in\mathcal{T}(s)}\tilde{G}_t}{|\mathcal{T}(s)|}$$

5.10 本章总结

蒙特卡洛方法是从多幕采样数据中学习价值函数和最优策略的方法蒙特卡洛方法相比于动态规划方法的优点：

1. MC 方法不需要描述环境动态特性的模型，而可以直接通过与环境的交互来学习最优的决策行为。

2. MC 方法可以使用数据仿真或采样模型。在非常多的应用中，虽然很难构建 DP 方法所需要的显式状态概率转移模型，但是通过仿真采样得到多幕序列数据却是很简单的。

3. MC 方法可以很简单和高效地聚焦于状态的一个小的子集，它可以只评估关注的区域而不评估其余的状态。

4. MC 方法在马尔可夫性不成立时性能损失较小。这是因为它不用后继状态的估值来更新当前的估值。换句话说，这是因为它不需要自举。

蒙特卡洛方法不需要建立一个模型计算每一个状态的价值，而只需简单地将从这个状态开始得到的多个回报平均即可。因为一个状态的价值就是回报的期望，因此这个平均值就是状态的一个很好的近似。在控制方法中，没有环境转移模型的情况下，动作价值函数也可以用于改进策略。蒙特卡洛方法逐幕地交替进行策略评估和策略改进，并可以逐幕地增量式实现。

保证足够多的试探是蒙特卡洛控制方法中的一个重要问题。贪心地选择在当前时刻的最优动作是不够的，因为这样其他的动作就不会得到任何回报，并且可能永远不会学到实际上可能更好的其他动作。一种避免这个问题的方法是随机选择每幕开始的“状态-动作二元组，使其能够覆盖所有的可能性。这种试探性出发方法有时可以在具备仿真采样数据的应用中使用，但不太可能应用在具有真实经验的学习中。另外一种方法是使用温和的采样策略。

- 在 on-policy 策略方法中，智能体一直保持试探并尝试寻找也能继续保持试探的最优策略。
- 在 off-policy 策略方法中，虽然智能体也在试探，但它实际学习的可能是与它试探时使用的策略无关的另一个确定性最优策略。

off-policy 策略预测指的是在学习目标策略的价值函数时，使用的是另一个不同的行动策略所生成的数据。这种学习方法基于某种形式的重要度采样，即用在两种策略下观察到的动作的概率的比值对回报进行加权，从而把行动策略下的期望值转化为目标策略下的期望值。

- 普通重要度采样将加权后的回报按照采样幕的总数直接平均，采样得到的是无偏估计，但具有更大甚至可能是无限的方差。
- 加权重要度采样则进行加权平均，它的方差总是有限的，因此在实践中更受青睐。

MC 方法与 DP 方法的不同之处在于以下两个方面。

1. 蒙特卡洛方法是用样本经验计算的，因此可以无需环境的概率转移模型，直接学习。

2. 蒙特卡洛方法不自举。也就是说，它们不通过其他价值的估计来更新自己的价值估计。

6 时序差分学习

时序差分学习（TD）结合了蒙特卡洛方法和动态规划方法的思想。

- 与 MC 方法一致的是：时序差分方法也可以直接从与环境互动的经验中学习策略，而不需要构建关于环境动态特性的模型。
- 与 DP 方法一致的是：时序差分方法无须等待交互的最终结果 (使用了自举思想)，而可以基于已得到的其他状态的估计值来更新当前状态的价值函数。

6.1 时序差分预测

1. 时序差分与蒙特卡洛间的差别

TD 和蒙特卡洛方法都利用经验来解决预测问题。给定策略 π 的一些经验，以及这些经验中的非终止状态 S_t，这两种方法都会更新它们对于 v_π 的估计 V。大致来说，蒙特卡洛方法需要一直等到一次访问后的回报知道之后，再用这个回报作为 $V(S_t)$ 的目标进行估计。一个适用于非平稳环境的简单的每次访问型蒙特卡洛方法可以表示成

$$V(S_t) \leftarrow V(S_t) + \alpha\Big[G_t - V(S_t)\Big] \tag{10}$$

在这里，G_t 是时刻 t 真实的回报，α 是常量步长参数。这个方法称作常量 αMC。蒙特卡洛方法必须等到一幕的末尾才能确定对 $V(S_t)$ 的增量，而 TD 方法只需要等到下一个时刻即可。在 $t+1$ 时刻，TD 方法立刻就能构造出目标，并使用观察到的收益 R_{t+1} 和估计值 $V(S_{t+1})$ 来进行一次有效更新。最简单的 TD 方法在状态转移到 S_{t+1} 并收到 R_{t+1} 的收益时会立刻做如下更新

$$V(S_t) \leftarrow V(S_t) + \alpha\Big[R_{t+1} + \gamma V(S_{t+1}) - V(S_t)\Big] \tag{11}$$

蒙特卡洛更新的目标是 G_t，而 TD 更新的目标是 $R_{t+1} + \gamma V(S_{t+1})$。这种 TD 方法被称为 TD(0)，或单步 TD。

2. TD(0) 算法

下框中的算法完整地描述了 TD(0) 的过程。

表格型 TD(0) 算法，用于估计 v_π

输入：待评估的策略 π
算法参数：步长 $\alpha \in (0, 1]$
对于所有 $s \in \mathcal{S}^+$，任意初始化 $V(s)$，其中 V(终止状态)=0
对每幕循环：
  初始化 S
  对幕中的每一步循环：
    $A \leftarrow$ 策略 π 在状态 S 下做出的决策动作执行动作 A，观察到 R，S'
    $V(S) \leftarrow V(S) + \alpha\Big[R + \gamma V(S') - V(S)\Big]$
    $S + S'$
  直到 S 为终止状态

$$
\begin{aligned}
v_\pi(s) &\doteq \mathbb{E}_\pi[G_t \mid S_t = s] && (12)\\
&= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s] \\
&= \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s] && (13)
\end{aligned}
$$

蒙特卡洛方法把对式12的估计值作为目标，而 DP 方法则把对13式的估计值作为目标。TD 的目标也是一个“估计值”，理由有两个：它采样得到对式13的期望值，并且使用当前的估计值 V 来代替真实值 v_π。因此，TD 算法结合了蒙特卡洛采样方法和 DP 自举法，可以很好地结合蒙特卡洛方法和 DP 方法的优势。

图6.1是表格型 TD(0) 的回溯图。回溯图顶部状态节点的价值的估计值是根据它到一个直接后继状态节点的单次样本转移来更新的。将 TD 和蒙特卡洛更新称为采样更新，因为它们都会通过采样得到一个后继状态 (或“状态-动作”二元组), 使用后继状态的价值和沿途得到的收益来计算回溯值，然后相应地改变原始状态 (或“状态-动作”二元组) 价值的估计值。采样更新和 DP 方法使用的期望更新不同，它的计算基于采样得到的单个后继节点的样本数据，而不是所有可能后继节点的完整分布。

图 6.1: 回溯图

在 TD(0) 的更新中，括号里的数值是一种误差，它衡量的是 S_t 的估计值与更好的估计 $R_{t+1} + \gamma V(S_{t+1})$ 之间的差异。这个数值，如下面公式所示，被称为 TD 误差

$$
\delta_t \doteq R_{t+1} + \gamma V(S_{t+1}) - V(S_t) \tag{14}
$$

每个时刻的 TD 误差是当前时刻估计的误差。由于 TD 误差取决于下一个状态和下一个收益，所以要到一个时刻步长之后才可获得。也就是说，$V(S_t)$ 中的误差 δ_t 在 $t+1$ 时刻才可获得。如果价值函数数组 V 在一幕内没有改变 (例如，蒙特卡洛方法), 则蒙特卡洛误差可写为 TD 误差之和

$$
\begin{aligned}
G_t - V(S_t) &= R_{t+1} + \gamma G_{t+1} - V(S_t) + \gamma V(S_{t+1}) - \gamma V(S_{t+1}) \\
&= \delta_t + \gamma\big(G_{t+1} - V(S_{t+1})\big) \\
&= \delta_t + \gamma\delta_{t+1} + \gamma^2\big(G_{t+2} - V(S_{t+2})\big) \\
&= \delta_t + \gamma\delta_{t+1} + \gamma^2\delta_{t+2} + \cdots + \gamma^{T-t-1}\delta_{T-1} + \gamma^{T-t}\big(G_T - V(S_T)\big) \\
&= \delta_t + \gamma\delta_{t+1} + \gamma^2\delta_{t+2} + \cdots + \gamma^{T-t-1}\delta_{T-1} + \gamma^{T-t}\,(0-0) \\
&= \sum_{k=t}^{T-1} \gamma^{k-t}\delta_k
\end{aligned} \tag{15}
$$

如果 V 在该幕中变化了 (例如 TD(0) 的情况，V 不断被更新), 那么这个等式就不准确，但如果时刻步长较小，则等式仍能近似成立。

6.2 时序差分预测方法的优势

1. 与 DP 方法对比

相比 DP 方法，TD 的优势在于它不需要环境模型，即描述收益和下一状态联合分布的模型。

2. 与 MC 方法对比

相比 MC 方法，TD 运用了一种在线的、完全递增的方法来实现。蒙特卡洛方法必须等到一幕的结束，而 TD 方法只需等到下一时刻即可。在一些应用场景中，幕非常长，所以把学习推迟到整幕结束之后就太晚了。在另一些应用场景中可能是持续性任务，无法划分出“幕”的概念。最后，有一些 MC 方法必须对那些采用实验性动作的幕进行打折或者干脆忽略掉，这可能会大大减慢学习速度。而 TD 方法则不太容易受到这些问题的影响，因为它们从每次状态转移中学习，与采取什么后续动作无关。

对于任何固定的策略 π, TD(0) 都已经被证明能够收敛到 v_π。如果步长参数是一个足够小的常数，那么它的均值能收敛到 v_π。目前虽然没有数学上证明出 TD 和 MC 的收敛速度哪一个更快，但是在实践中，TD 方法在随机任务上通常比常量 α MC 方法收敛得更快。

6.3 TD(0) 的最优性

1. 批量更新

假设只有有限的经验，使用增量学习方法的一般方式是反复地呈现这些经验，直到方法最后收敛到一个答案为止。给定近似价值函数 V, 在访问非终止状态的每个时刻 t, 使用式10或式11计算相应的增量，但是价值函数仅根据所有增量的和改变一次。然后，利用新的值函数再次处理所有可用的经验，产生新的总增量，依此类推，直到价值函数收敛。称这种方法为批量更新，因为只有在处理了整批的训练数据后才进行更新。

2. TD 与 MC 的收敛结果

在批量更新下，只要选择足够小的步长参数 α, TD(0) 就能确定地收敛到与 α 无关的唯一结果。常数 α MC 方法在相同条件下也能确定地收敛，但是会收敛到不同的结果。

批量 MC 总是找出最小化训练集上均方误差的估计，而批量 TD(0) 总是找出完全符合马尔可夫过程模型的最大似然估计参数。通常，一个参数的最大似然估计是使得生成训练数据的概率最大的参数值。马尔可夫过程模型参数的最大似然估计可以很直观地从观察到的多幕序列中得到。从 i 到 j 的转移概率估计值，就是观察数据中从 i 出发转移到了的次数占从 i 出发的所有转移次数的比例。而相应的期望收益则是在这些转移中观察到的收益的平均值。可以据此来估计价值函数，并且如果模型是正确的，估计也就完全正确。这种估计被称为确定性等价估计，因为它等价于假设潜在过程参数的估计是确定性的而不是近似的。批量 TD(0) 通常收敛到的就是确定性等价估计。

这一点也有助于解释为什么 TD 方法比 MC 方法更快地收敛。在以批量的形式学习的时候，TD(0) 比 MC 方法更快是因为它计算的是真正的确定性等价估计。

虽然确定性等价估计从某种角度上来说是一个最优答案，但直接计算它几乎是不可能的。如果 $n = |\mathcal{S}|$ 是状态数，那么仅仅建立过程的最大似然估计就可能需要 n^2 的内存，如果按传统方法，计算相应的价值函数则需要 n^3 数量级的步骤。相比之下，TD 方法则可以使用不超过 n 的内存，并且通过在训练集上反复计算来逼近同样的答案。对于状态空间巨大的任务，TD 方法可能是唯一可行的逼近确定性等价解的方法。

6.4 Sarsa：on-policy 策略下的时序差分控制

与 MC 方法一样，TD 方法同样需要在探索与利用做出权衡，策略也同样能分为 on-policy 和 off-policy。以下为 on-policy 策略下的 TD 控制方法。

第一步学习的是行为值函数，对所有状态 s 及动作 a 估计出当前行动策略下对应的 $q_\pi(s, a)$。

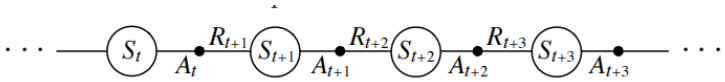

图 6.2: 状态和动作交替出现的序列

“状态-动作”二元组之间的转移并学习“状态-动作”二元组的价值与 6.3 节的方法一致，都是是带有收益过程的马尔可夫链。确保状态值在 TD(0) 下收敛的定理同样也适用于对应的关于动作值的算法上

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\Big[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)\Big] \tag{16}$$

每当从非终止状态的 S_t 出现一次转移之后，就进行上面的一次更新。如果 S_{t+1} 是终止状态，那么 $Q(S_{t+1}, A_{t+1})$ 则定义为 0。这个更新规则用到了描述这个事件的五元组 $(S_t, A_t, R_{t+1}, S_{t+1}, A_{t+1})$ 中的所有元素。根据这个五元组把这个算法命名为 $Sarsa$。Sarsa 的回溯图如图6.3所示。

图 6.3

基于 Sarsa 预测方法设计一个同轨策略下的控制算法，和所有其他的同轨策略方法一样，持续地为行动策略 π 估计其动作价值函数 q_π，同时以 q_π 为基础，朝着贪心优化的方向改变 π。下框中给出了 Sarsa 控制算法的一般形式。

Sarsa 算法，用于估计 $Q \approx q\pi$

算法参数：步长 $\alpha \in (0, 1]$, 很小的 $\varepsilon, \varepsilon > 0$
对所有 $s \in S^+, a \in \mathcal{A}(s)$, 任意初始化 $Q(s, a)$, 其中 $Q($ 终止状态，$\cdot) = 0$
对每幕循环：
  初始化 S
  使用从 Q 得到的策略 (例如 ε-贪心), 在 S 处选择 A
    对幕中的每一步循环：
    执行动作 A, 观察到 R, S'
    使用从 Q 得到的策略 (例如 -贪心), 在 S' 处选择 A'
  $Q(S, A) \leftarrow Q(S, A) + \alpha\Big[R + \gamma Q(S', A') - Q(S, A)\Big]$
    $S \leftarrow S'$; $A \leftarrow A'$;
  直到 S 是终止状态

6.5 Q 学习：off-policy 策略下的时序差分控制

off-polcy 策略下的时序差分控制算法被称为 Q 学习, 其定义为

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\Big[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)\Big] \tag{17}$$

在这里，待学习的动作价值函数 Q 采用了对最优动作价值函数 q_* 的直接近似作为学习目标，而与用于生成智能体决策序列轨迹的行动策略是什么无关 (作为对比，Sarsa 的学习目标中使用的是待学习的动作价值函数本身，由于它的计算需要知道下一时刻的动作 A_{t+1}, 因此与生成数据的行动策略是相关的)。这大大简化了算法的分析，也很早就给出收敛性证明。只需要所有的“状态-动作”二元组可以持续更新，整个学习过程就能够正确地收敛。基于这种假设以及步长参数序列的某个常用的随机近似条件，可以证明 Q 能以 1 的概率收敛到 q_*。Q 学习算法的流程如下面框中所示。

Q 学习算法，用于预测 $\pi \approx \pi_*$

算法参数：步长 $\alpha \in (0,1]$, 很小的 $\varepsilon, \varepsilon > 0$

对所有 $s \in S^+, a \in \mathcal{A}(s)$, 任意初始化 $Q(s,a)$, 其中 $Q($ 终止状态，$\cdot)$=0

对每幕：

初始化 S 对幕中的每一步循环：

使用从 Q 得到的策略 (例如 ε-贪心), 在 S 处选择 A

执行 A, 观察到 R, S'

$Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma \max_a Q(S',a) - Q(S,A)\big]$

$S + S'$

直到 S 是终止状态

Q 学习回溯图的顶部节点，也即更新过程的根节点，必须是一个代表动作的小实心圆点。更新也都来自于动作节点，即在下一个状态的所有可能的动作中找到价值最大的那个。因此回溯图底部的节点就是所有这些可能的动作节点。