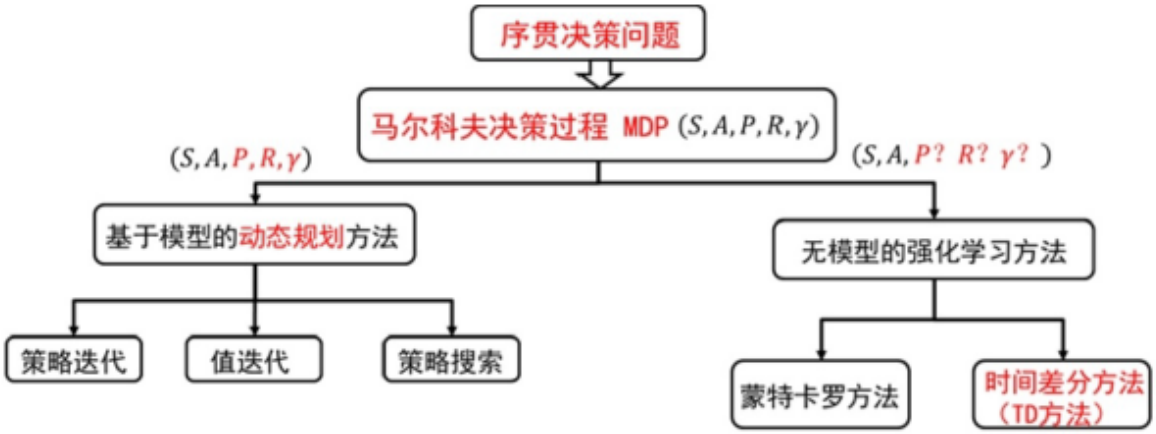
4 Temporal Difference-Based Reinforcement Learning Method

4.1 基于时间差分的强化学习算法理论

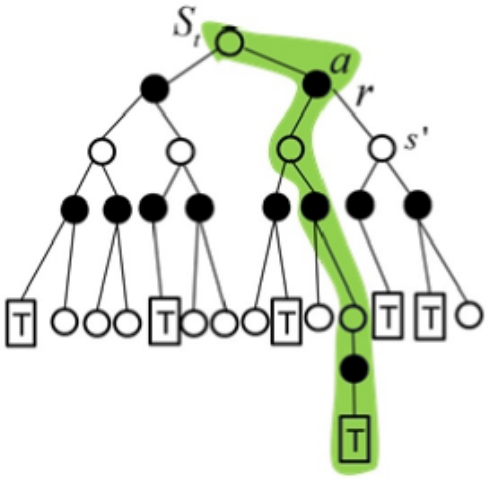
时间差分（Temporal-Difference，简称TD）方法是另一种无模型强化学习方法，也是强化学习理论中最核心的内容。与动态规划的方法和蒙特卡洛的方法相比，时间差分方法的主要不同在于值函数的估计。



用动态规划方法计算值函数时用到了当前状态 s 的所有后继状态 s' 处的值函数。值函数的计算用到了 bootstrapping 的方法。所谓 bootstrapping 本意是指自举，此处是指当前值函数的计算用到了后继状态的值函数。即用后继状态的值函数估计当前值函数。要注意的是，此处后继的状态是由模型公式 $p(s',r|S_t,a)$ 计算得到的。由模型公式和动作集，可以计算状态 s 所有的后继状态 s' 。当没有模型时，后继状态无法全部得到，只能通过试验和采样的方法每次试验得到一个后继状态 s' 。

$$V(S_t) \leftarrow E_\pi[R_{t+1}+\gamma V(S_{t+1})] = \sum_a \pi(a|S_t) \sum_{s',r} p(s.,r|S_t,a)[r+\gamma V(s')] \tag{7}$$

无模型时，我们可以采用蒙特卡洛的方法利用经验平均来估计当前状态的值函数，用它计算值函数的过程如下图所示。



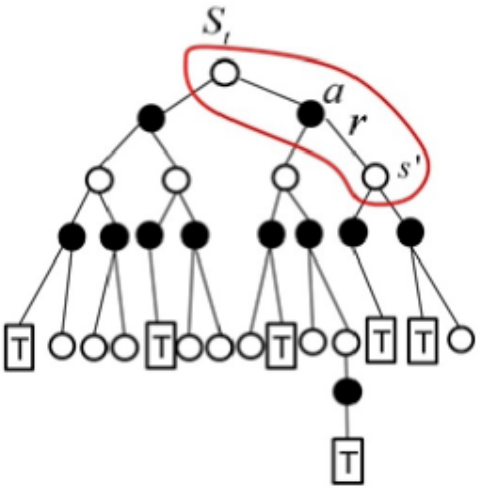
蒙特卡洛方法利用经验平均估计状态的值函数。此处的经验是指一次试验，而一次试验要等到终止状态出现才结束。

$$V(S_t) \leftarrow V(S_t) + \alpha(G_t - V(S_t)) \tag{8}$$

其中， G_t 是状态 S_t 处的折扣累积回报值。

相比于动态规划的方法，蒙特卡洛的方法需要等到每次试验结束，所以学习速度慢，学习效率不高。

时间差分方法结合了蒙特卡洛的采样方法和动态规划方法的 bootstrapping ，它的计算过程如下图所示。

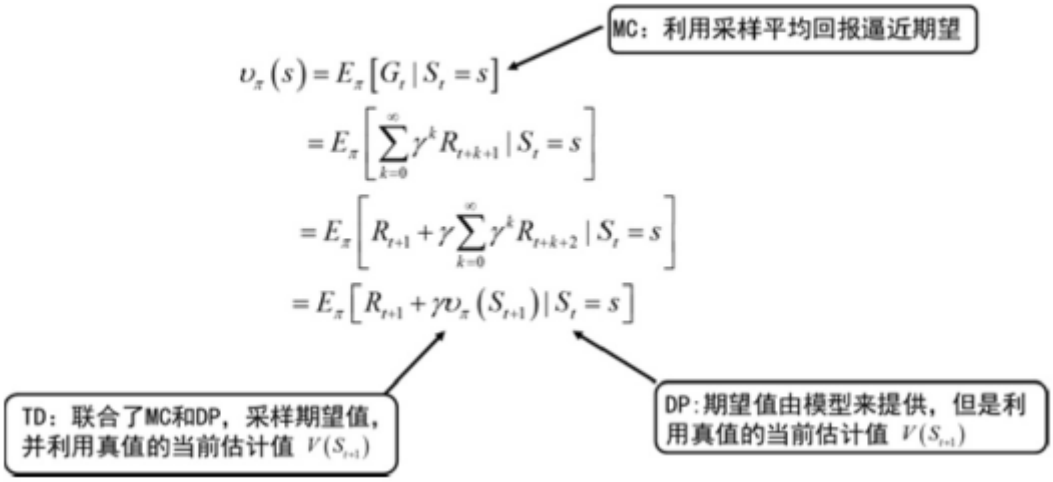


时间差分方法的值函数公式为

$$V(S_t) \leftarrow V(S_t) + \alpha(R_{t+1} + \gamma V(S_{t+1}) - V(S_t)) \tag{9}$$

其中 $R_{t+1} + \gamma V(S_{t+1})$ 称为 TD 目标，与（2）式的 G_t 相对应，两者不同之处是 TD 目标利用了 bootstrapping 方法估计当前值函数。 $\delta_t = R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$ 称为 TD 偏差。

下图为三种方法估计值函数的异同点。蒙特卡洛方法使用的是值函数最原始的定义，该方法利用所有回报的累积和估计值函数；动态规划方法和时间差分方法则利用一步预测方法计算当前状态值函数，它们的共同点是利用了 bootstrapping 方法，不同的是，动态规划方法利用模型计算后继状态，时间差分方法利用试验得到后继状态。



从统计学角度来看，蒙特卡洛方法和时间差分方法都是利用样本估计值函数的方法。我们可以从期望和方差两个指标对比两种方法。

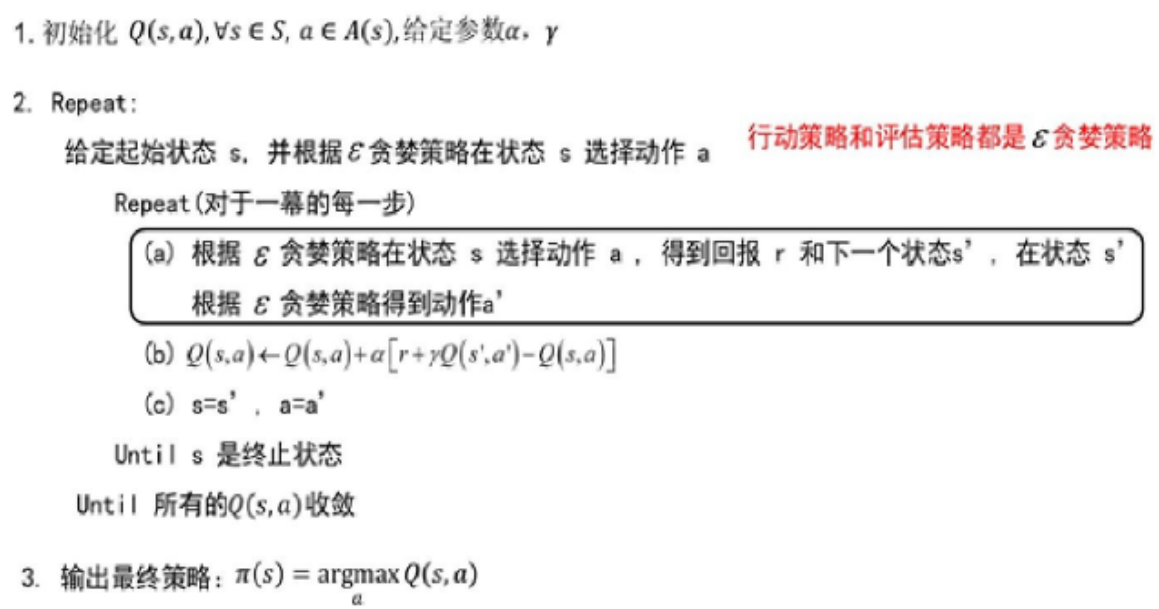
蒙特卡洛方法

蒙特卡洛方法中的返回值 $G_t = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-1} R_T$ ，其期望便是值函数的定义，因此蒙特卡洛是无偏估计。但是，蒙特卡洛方法每次得到的 G_t 值要等到最终状态出现，在这个过程中会经历很多随机的状态和动作，每次得到的 G_t 随机性很大，因此尽管期望等于真值，但方差无穷大。

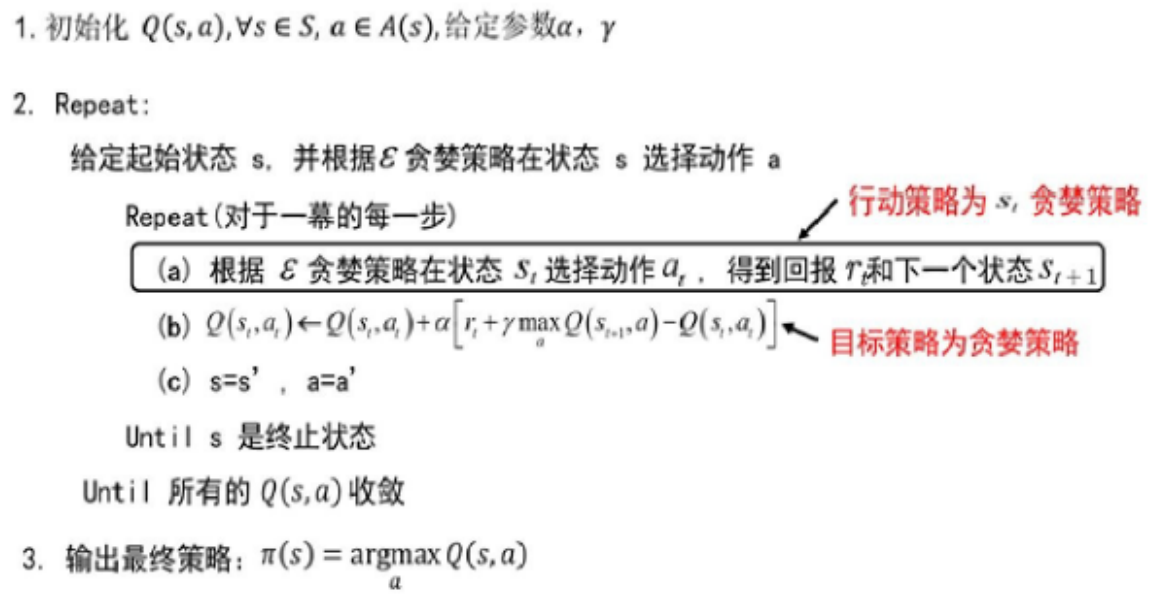
时间差分方法

时间差分的 TD 目标为 $R_{t+1}+\gamma V(S_{t+1})$，若 $V(S_{t+1})$ 采用真实值，则 TD 估计也是无偏估计，然而在试验中 $V(S_{t+1})$ 用的也是估计值，因此时间差分估计方法属于有偏估计。与蒙特卡洛方法相比，时间差分方法只用到了一步随机状态和动作，因此 TD 目标的随机性比蒙特卡洛方法中的 G_t 要小，相应的方差也比蒙特卡洛方法中的方差小。

时间差分方法包括同策略的 Sarsa 方法和异策略的 Q-learning 方法。下图为同策略 Sarsa 强化学习算法，需要注意的是方框中代码表示同策略中的行动策略和评估的策略都是 $\epsilon-greedy$ 策略。与蒙特卡洛方法不同的是，它的值函数更新不同。



下图为异策略的 Q-learning 方法。与 Sarsa方法的不同之处在于，Q-learning方法是异策略的方法。即行动策略采用 $\epsilon-greedy$ 策略，目标策略为贪婪策略。



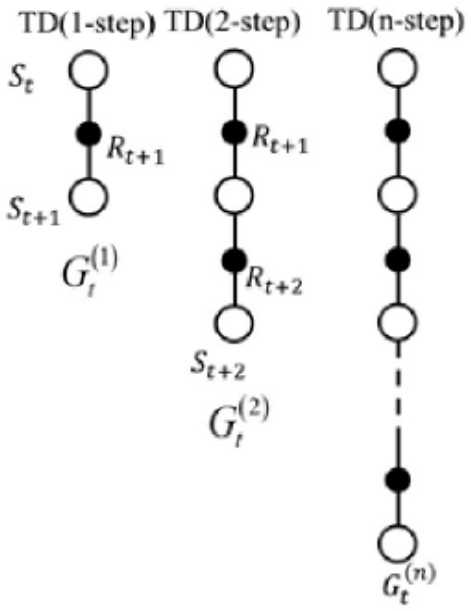
4.2 $TD(\lambda)$ 方法

TD 方法除了常用的 Sarsa 方法和 Q-learning 方法，还包括 $TD(\lambda)$ 方法。下面详细阐述 $TD(\lambda)$ 方法的来龙去脉。

在TD方法中，更新当前值函数时，用到了下一个状态的值函数，也可以用后继第二个状态的值函数来更新当前状态的值函数。我们用 $G_t^{(1)}=R_{t+1}+\gamma V(S_{t+1})$ 表示 TD目标，利用第二步值函数来估计当前值函数可以表示为 $G_t^{(2)}=R_{t+1}+\gamma R_{t+2}+\gamma^2 V(S_{t+2})$。以此类推，利用第 n 步的值函数更新当前值函数可表示为

$$G_t^{(n)} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n V(S_{t+n}) \tag{10}$$

下图为利用 n 步值函数估计当前值函数的示意图。



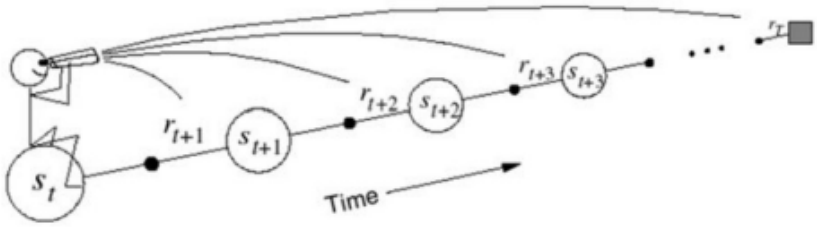
我们可以利用 n 步值函数来估计当前值函数，也就是说当前值函数有 n 中估计方法。因此，我们可以利用加权的方法融合这 n 个估计值，即 $TD(\lambda)$ 方法。

在 $G_t^{(n)}$ 前乘以加权因子 $(1-\lambda)\lambda^{n-1}$ ，之所以要乘以加权，原因在于

$$\begin{aligned} G_t^{\lambda} &= (1-\lambda)G_t^{(1)} + (1-\lambda)\lambda G_t^{(2)} + \cdots + (1-\lambda)\lambda^{n-1} G_t^{(n)} \\ &\approx [(1-\lambda) + (1-\lambda)\lambda + \cdots + (1-\lambda)\lambda^{n-1}] V(S_t) \\ &= V(S_t) \end{aligned} \tag{11}$$

利用 G_t^{λ} 更新当前状态的值函数的方法称为 $TD(\lambda)$ 的方法。一般可以从两个视角理解 $TD(\lambda)$ 。

第一视角是前向视角，该视角也是 G_t^{λ} 的定义。

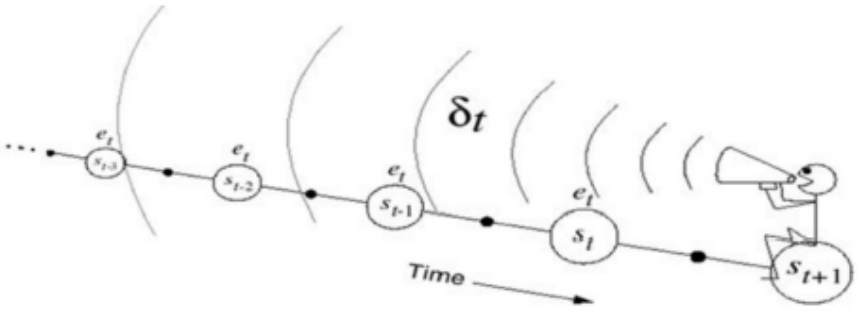


$$V(S_t) \leftarrow V(S_t) + \alpha(G_t^{\lambda} - V(S_t)) \tag{12}$$

其中， $G_t^{\lambda} = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}$ ，而
$G_t^{(n)} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n V(S_{t+n})$ 。

利用 $TD(\lambda)$ 的前向观点估计值函数时， G_t^{λ} 的计算用到了将来时刻的值函数，因此需要整个实验结束后才能计算，这和蒙特卡洛方法相似。

第二视角是后向视角，这种增量式的更新方法不需要等到试验结束就可以更新当前状态的值函数。下图为 $TD(\lambda)$ 后向观点示意图。



假设当前状态为 s_t ， TD 偏差为 δ_t ，那么 s_{t-1} 处的值函数更新应该乘以一个衰减因子 $\gamma\lambda$ ，状态 s_{t-2} 处的值函数更新应该乘以 $(\gamma\lambda)^2$ ，以此类推。

$TD(\lambda)$ 更新过程如下。

首先，计算当前状态的 TD 偏差： $\delta_t = R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$ 。

其次，更新适合度轨迹： $E_t(s) = \begin{cases} \gamma\lambda E_{t-1} & if\ s \neq s_t \\ \gamma\lambda E_{t-1} + 1 & if\ s = s_t \end{cases}$ 。

最后，对于状态空间中的每个状态 s ，更新值函数： $V(s) \leftarrow V(s) + \alpha\delta_t E_t(s)$ 。

其中， $E_t(s)$ 称为适合度轨迹。

前向观点和后向观点的异同：

（1）前向观点需要等到一次试验结束之后再更新当前状态的值函数；后向观点不需要等到值函数结束后再更新值函数，而是每一步都在更新值函数，是增量式方法。

（2）前向观点在一次试验结束后更新值函数时，更新完当前状态的值函数后，此状态的值函数就不再改变。后向观点在每一步计算完当前的 TD 误差后，其他状态的值函数需要利用当前状态的 TD 误差更新。

（3）在一次试验结束后，前向观点和后向观点每个状态的值函数的更新总量是相等的，都是 G_t^λ 。

$Sarsa(\lambda)$ 算法的伪代码如下图所示。

1. 初始化 $Q(s,a), \forall s \in S, a \in A(s)$, 给定参数 α γ
2. Repeat:　　行动策略和评估策略都是 ε 贪婪策略

   给定起始状态 s，并根据 ε 贪婪策略在状态 s 选择动作 a，对所有的 $s \in S, a \in A(s), E(s,a) = 0$

   Repeat（对于一幕的每一步）

   (a) 根据 ε 贪婪策略在状态 s 选择动作 a，得到回报 r 和下一个状态s'，在状态 s' 根据 ε 贪婪策略得到动作a'

   (b) $\delta \leftarrow r + \gamma Q(s',a') - Q(s,a)$, $E(s,a) \leftarrow E(s,a) + 1$

   (c) 对所有的 $s \in S, a \in A(s)$: $Q(s,a) \leftarrow Q(s,a) + \alpha\delta E(s,a)$, $E(s,a) \leftarrow \gamma\lambda E(s,a)$

   (d) s=s'，a=a'

   Until s 是终止状态

   Until 所有的 $Q(s,a)$ 收敛
3. 输出最终策略： $\pi(s) = \arg\max_a Q(s,a)$