强化学习

Reinforcement learning

第4节 无模型（model-free）的预测

张世周

Outlines

- 1.1 简介
- 1.2 蒙特卡洛强化学习
- 1.3 时序差分学习
- 1.4 TD(λ)

简介

1、上节课：

- **如何从理论上解决一个已知的MDP；**
- **Planning by Dynamic Programming**

2、本节课：

- **我们将讨论无模型（model-free）的预测；**
- **从未知的MDP中评估价值函数**

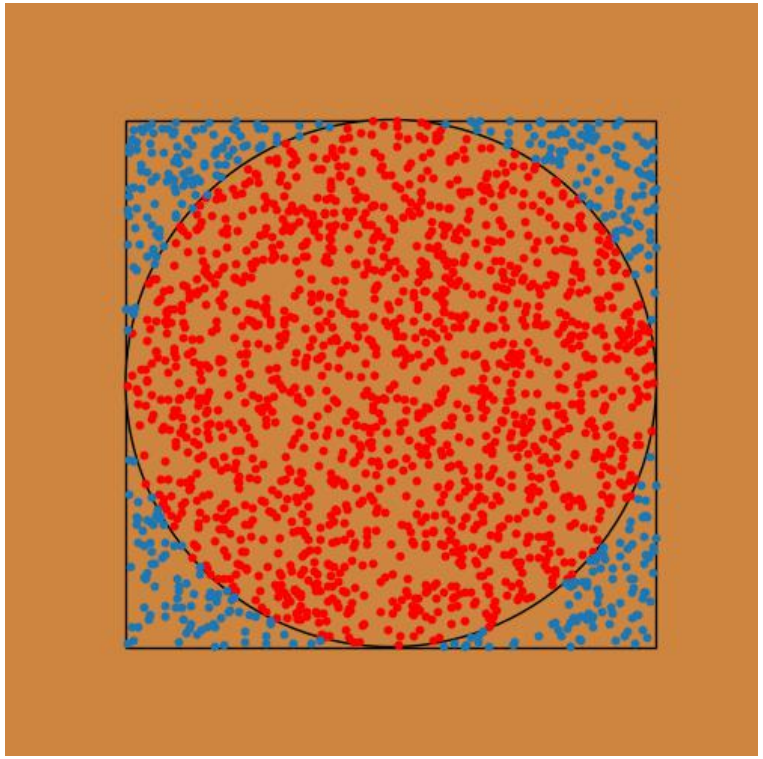
3、下节课：

- **无模型控制，**
- **从未知的MDP中优化价值函数**

蒙特卡洛强化学习

蒙特卡洛思想

赌城蒙特卡洛



蒙特卡洛方法计算圆周率Π

蒙特卡洛强化学习

特点：

- MC直接从经验Episode来学习状态价值
- MC是无模型的方法：即不知道MDP当中的转移/奖励
- MC使用完整的episode（幕）数据：没有“自举”
- MC使用的思想就是用平均收获值代替价值，value= mean return
- 注意：MC仅能应用于分幕式MDP（episodic MDP），即每一幕必然会终结

思想：

分布未知？--->采样！

蒙特卡洛强化学习

蒙特卡洛策略评估

- 目标：给定策略π，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v_π。

$$S_1, A_1, R_2, \ldots, S_k \sim \pi$$

- 回报：总折扣奖励 $G_t = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{T-1} R_T$
- 价值函数：回报的期望 $v_\pi(s) = E_\pi[G_t \mid S_t = s]$
- 蒙特卡罗策略评估：使用经验平均回报代替期望回报

蒙特卡洛强化学习

首次访问（first-visit）蒙特卡洛策略评估

- 在给定一个策略，使用一系列完整Episode评估某一个状态s时，对于每一个Episode，仅当该状态第一次出现时列入计算：
  - 状态出现的次数加1：$N(s) \leftarrow N(s)+1$
  - 总的收获值更新：$S(s) \leftarrow S(s)+G_t$
  - 状态s的价值：$V(s)=S(s)/N(s)$
  - 当 $N(s) \rightarrow \infty$ 时，$V(s) \rightarrow v_\pi(s)$ (大数定律)

蒙特卡洛强化学习

每次访问（every-visit）蒙特卡洛策略评估

在给定一个策略，使用一系列完整Episode评估某一个状态s时，对于每一个Episode，状态s每次出现在状态转移链时，计算的具体公式与上面的一样，但具体意义不一样。

- 状态出现的次数加1：$N(s) \leftarrow N(s)+1$
- 总的收获值更新：$S(s) \leftarrow S(s)+G_t$
- 状态s的价值：$V(s)=S(s)/N(s)$
- 当 $N(s) \rightarrow \infty$ 时，$V(s) \rightarrow v_\pi(s)$ （大数定律）

二十一点游戏（BlackJack）

二十一点是一种流行于赌场的游戏，其目标是使得你的扑克牌点数之和在不超过21的情况下越大越好。所有的人头牌（J、Q、K）的点数为10，A即可当1也可当作11。假设每一个玩家都独立地与庄家进行比赛。游戏开始时，会给各玩家与庄家发两张牌。庄家的牌一张正面朝上一张背面朝上。玩家直接获得21点（一张A与一张10）的情况称之为天和。此时玩家直接获胜，除非庄家也是天和，那就是平局。如果玩家不是天和，那么他可以一张一张地继续要牌，直到他主动停止（停牌）或者牌的点数和超过21点（爆牌）。如果玩家爆牌了就算输掉比赛。如果玩家选择停牌，就轮到庄家行动。庄家根据一个固定的策略进行游戏：他一直要牌，直到点数等于或超过17时停牌。如果庄家爆牌，那么玩家获胜，否则根据谁的点数更靠近21点决定胜负或平局。胜、负、平局分别获得收益+1、-1和0。

蒙特卡洛强化学习

二十一点游戏（BlackJack）

状态空间：（多达200种）

- 当前牌的分数（12 - 21），低于12时，你可以安全的再叫牌，所以没意义。
- 庄家出示的牌（A - 10），庄家显示一张牌面给玩家
- 我有“useable” ace吗？（是或否）A既可以当1点也可以当11点。

行为空间：

- 停止要牌 stick
- 继续要牌 twist

状态转换（Transitions）：牌分小于12时，自动要牌

二十一点游戏

奖励（停止要牌）：

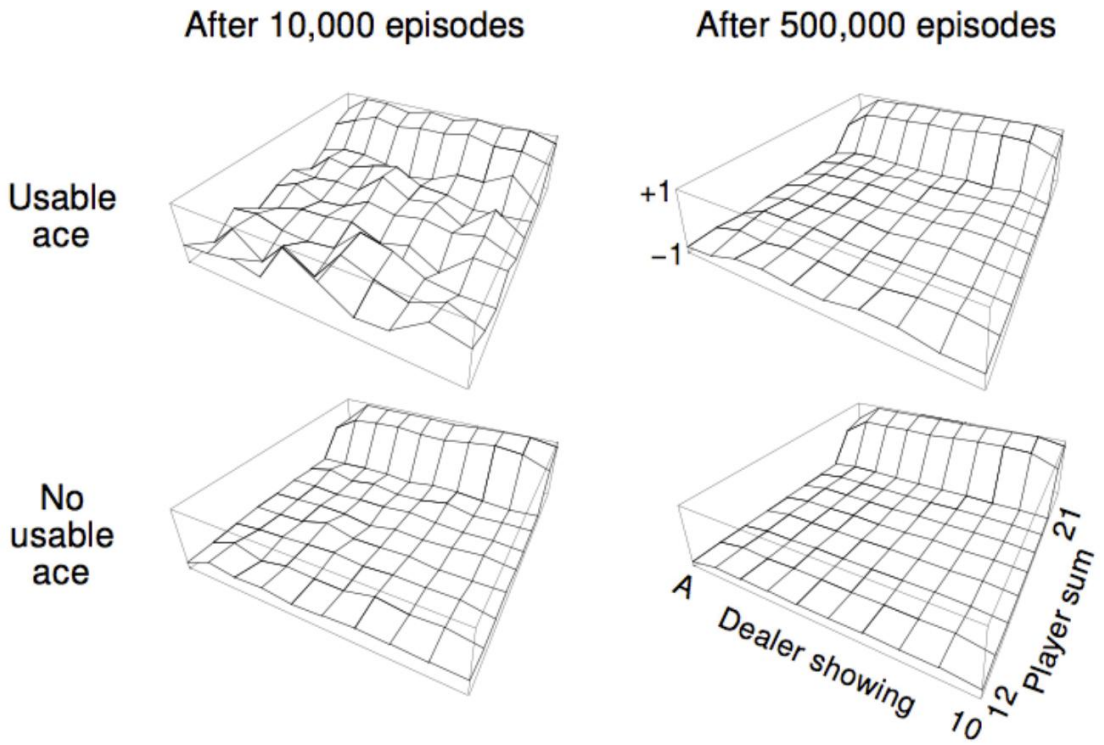
- +1：如果你的牌分数大于庄家分数
- 0： 如果两者分数相同
- -1：如果你的牌分数小于庄家分数

奖励（继续要牌）：

- -1：如果牌的分数>21，并且进入终止状态
- 0：其它情况

蒙特卡洛强化学习

经过蒙特卡洛学习之后的二十一点价值函数



策略：只要总点数不大于20时继续要牌

蒙特卡洛强化学习

增量更新平均值

在使用蒙特卡洛方法求解平均收获时，需要计算平均值。通常计算平均值要预先存储所有的数据，最后使用总和除以此次数。这里介绍了一种增量式方法

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

蒙特卡洛强化学习

蒙特卡洛增量更新---αMC

- 在经历每个完整的episode之后更新 $V(s)$
- 对于每个状态 S_t 和回报 G_t 来说

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

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

- 在处理非平稳问题时，使用这个方法跟踪一个实时更新的平均值是非常有用的，可以扔掉那些已经计算过的Episode信息。

$$V(S_t) \leftarrow V(S_t)+\alpha(G_t - V(S_t))$$

时序差分学习（Temporal Difference Learning）

- 时序差分学习和蒙特卡洛学习一样，直接从经验Episode中学习；
- TD是无模型方法：不需要了解模型本身；
- TD可以从不完整的Episode中学习，通过“自举”（bootstrapping）的思想；
- TD利用猜测的Episode的结果，来更新猜测（同时持续更新这个猜测）。

时序差分学习

MC与TD

- **目标：都是在给定的策略下，从经验中学习价值函数**
- **在MC学习中，使用实际的收获（return）G_t 来更新价值（Value）：**

$$V(S_t) \leftarrow V(S_t) + \alpha(G_t - V(S_t))$$

- **在TD学习中，算法在估计某一个状态的价值时，用的是离开该状态的即刻奖励 R_{t+1} 与下一状态 S_{t+1} 的预估状态价值乘以衰减系数 γ 组成，这符合Bellman方程的描述：**

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

$R_{t+1} + \gamma V(S_{t+1})$ 称为 **TD目标值**

$\delta_t = (R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$ 称为**TD误差**

时序差分学习

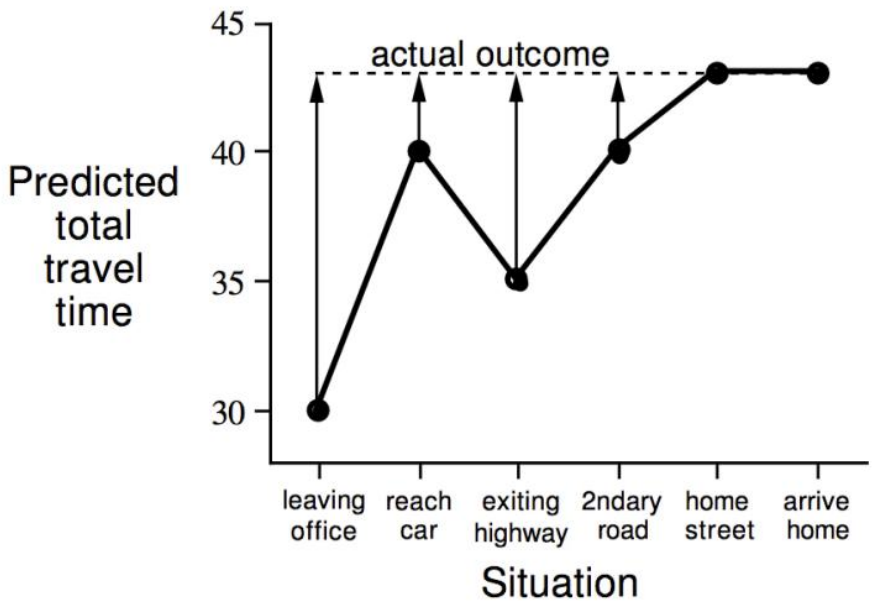
例子：驾车返家

状态	已消耗时间	预计仍需耗时	预计总耗时
离开办公室	0	30	30
取车，发现下雨	5	35	40
离开高速公路	20	15	35
被迫跟在卡车后	30	10	40
到达家所在街区	40	3	43
进入家门	43	0	43

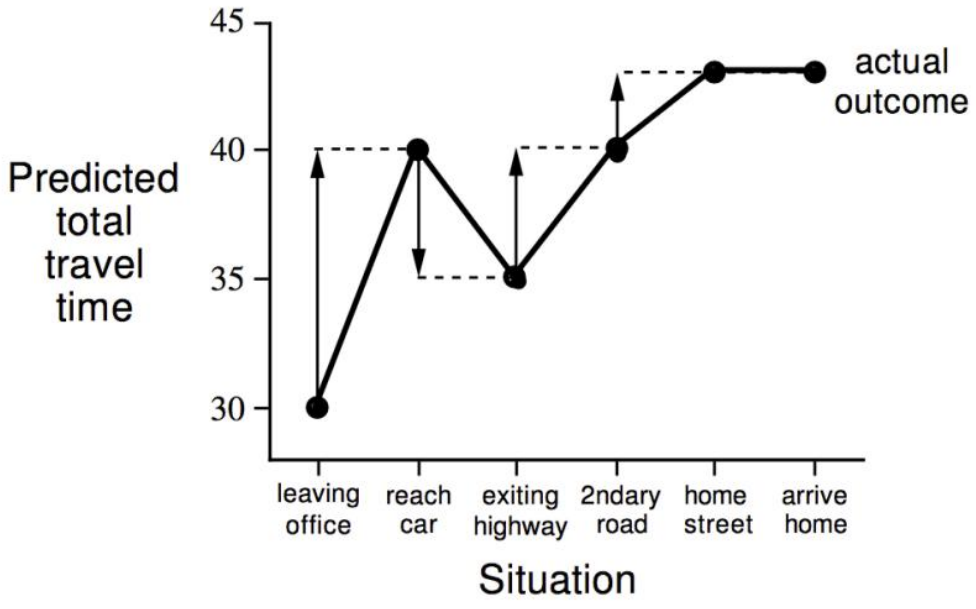
时序差分学习

驾车返家：MC vs TD

Changes recommended by Monte Carlo methods (α=1)



Changes recommended by TD methods (α=1)



时序差分学习

MC和TD的优劣势

TD 在知道结果之前可以学习

- **TD可以在每步之后在线学习**
- **MC必须等到最后结果才能学习**

TD 可以在没有结果时学习

- **TD可以从不完整的序列中学习**
- **MC只能从完整的序列中学习**
- **TD可以在持续进行的环境里学习**
- **MC必须在有结果的序列中学习**

时序差分学习

偏差/方差权衡 https://www.zhihu.com/question/22983179

- G_t：回报，是基于某一策略状态价值的无偏估计

$$G_t = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{T-1} R_T$$

- **TD target**：TD目标值，是基于下一状态预估价值计算的当前预估收获，是当前状态实际价值的有偏估计

$$R_{t+1} + \gamma V(S_{t+1})$$

- TD目标值与回报相比具有更低的方差
  - 回报的值取决于很多随机的动作、转移概率和收益
  - TD目标值只取决于一个随机的动作、转移概率和收益

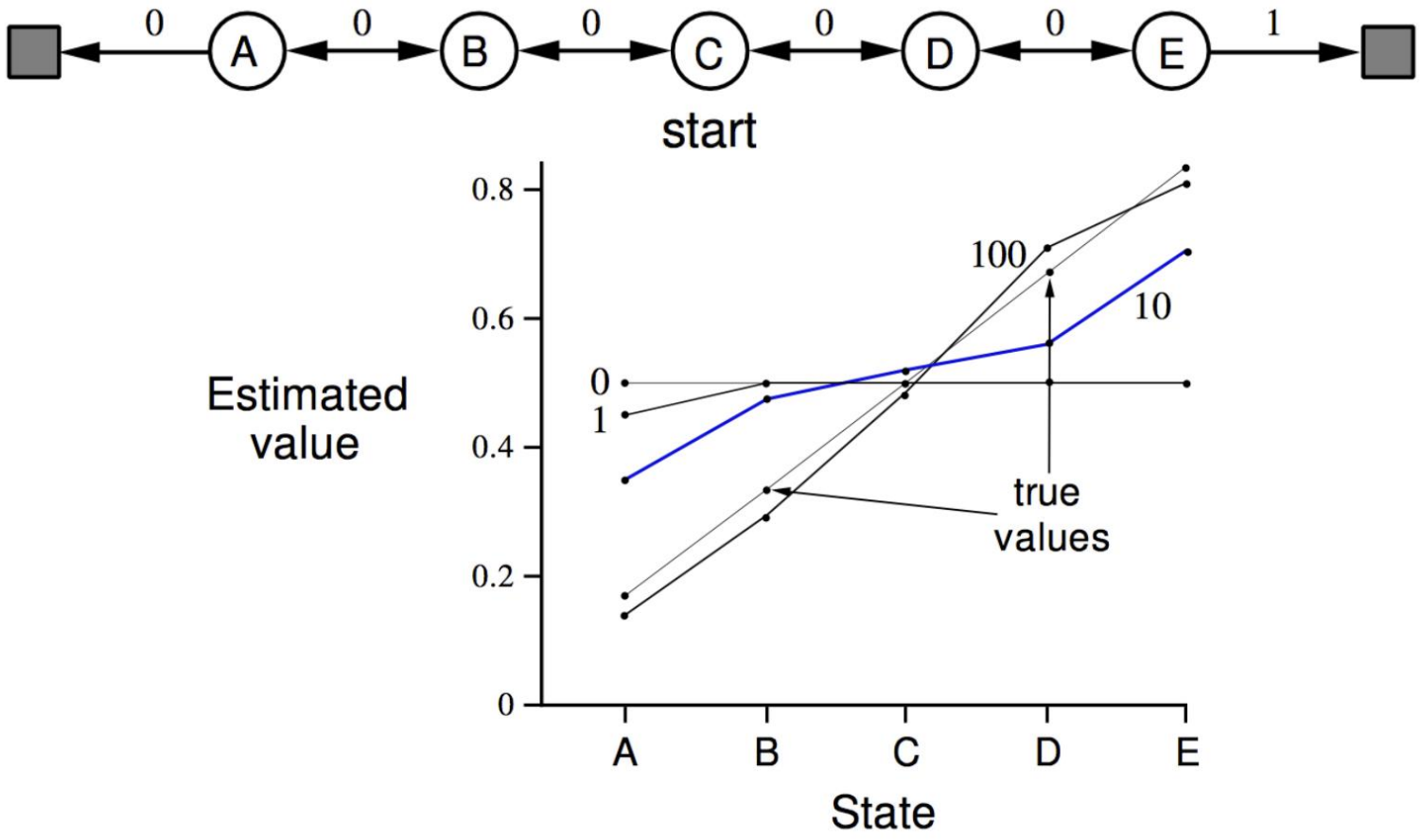
时序差分学习

MC和TD的优劣势（2）

- **MC 没有偏差（bias），但有着较高的方差（Variance），且对初始值不敏感；**
- **TD 低方差（variance）, 但有一定程度的偏差，对初始值较敏感，通常比MC 更高效；**

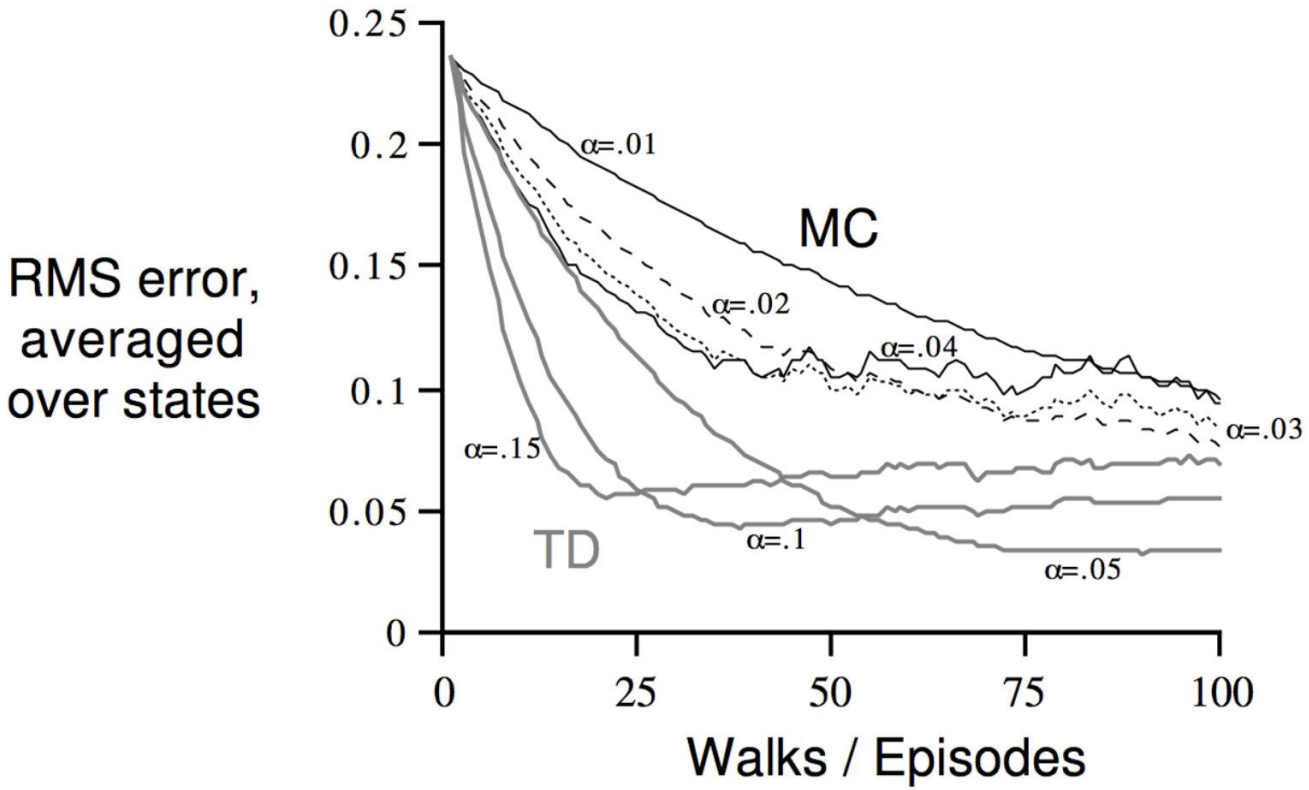
时序差分学习

例子：评估随机游走的价值函数



时序差分学习

随机游走：MC vs TD方法结果对比



时序差分学习

批（Batch） MC 和 TD

- **MC和TD的收敛：当经验无穷多时** $V(s) \to v_\pi(s)$
- **但是对于有限经验的批处理解决方案呢？**

$$s_1^1, a_1^1, r_2^1, \ldots, s_{T_1}^1$$

$$\vdots$$

$$s_1^K, a_1^K, r_2^K, \ldots, s_{T_K}^K$$

- **例如重复采样 k ∈ [1,K]**
- **将 MC 或 TD(0) 应用于第 k 个episode**

时序差分学习

示例：AB

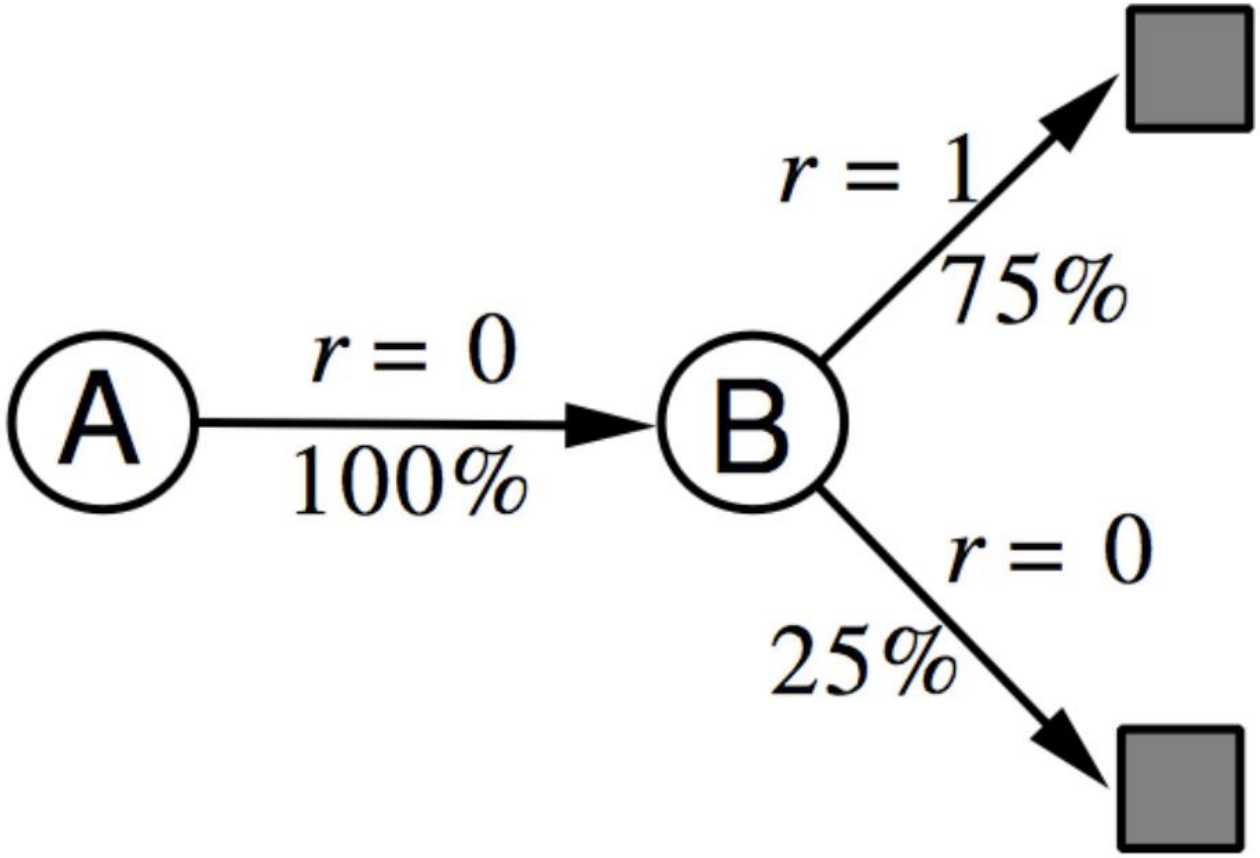
已知：现有两个状态(A和B)，MDP未知，衰减系数为1，有如下表所示8个完整Episode的经验及对应的即时奖励，其中除了第1个Episode有状态转移外，其余7个均只有一个状态。

Episode	状态转移及奖励
1	A:0,B:0
2	B:1
3	B:1
4	B:1
5	B:1
6	B:1
7	B:1
8	B:0

问题：依据仅有的Episode，计算状态A，B的价值分别是多少，即V(A)=？， V(B)=？

时序差分学习

MC:
V(A)=0
V(B)=3/4
TD:
V(A)=V(B)=3/4



时序差分学习---确定性等价评估

- MC算法试图收敛至一个能够最小化状态价值与实际收获的均方差的解决方案，这一均方差用公式表示为：

$$\sum_{k=1}^{K}\sum_{t=1}^{T_k}(G_t^k - V(s_t^k))^2$$

- 在AB例子中使用MC算法可得 $V(A)=0$
- TD算法则收敛至一个根据已有经验构建的最大可能的马尔可夫模型的状态价值，也就是说TD算法将首先根据已有经验估计状态间的转移概率：同时估计某一个状态的即时奖励：

$$\hat{\mathcal{P}}_{s,s'}^{a} = \frac{1}{N(s,a)}\sum_{k=1}^{K}\sum_{t=1}^{T_k}\mathbf{1}(s_t^k, a_t^k, s_{t+1}^k = s, a, s')$$

$$\hat{\mathcal{R}}_{s}^{a} = \frac{1}{N(s,a)}\sum_{k=1}^{K}\sum_{t=1}^{T_k}\mathbf{1}(s_t^k, a_t^k = s, a)r_t^k$$

- 在AB例子中使用TD算法可得 $V(A)=0.75$

时序差分学习

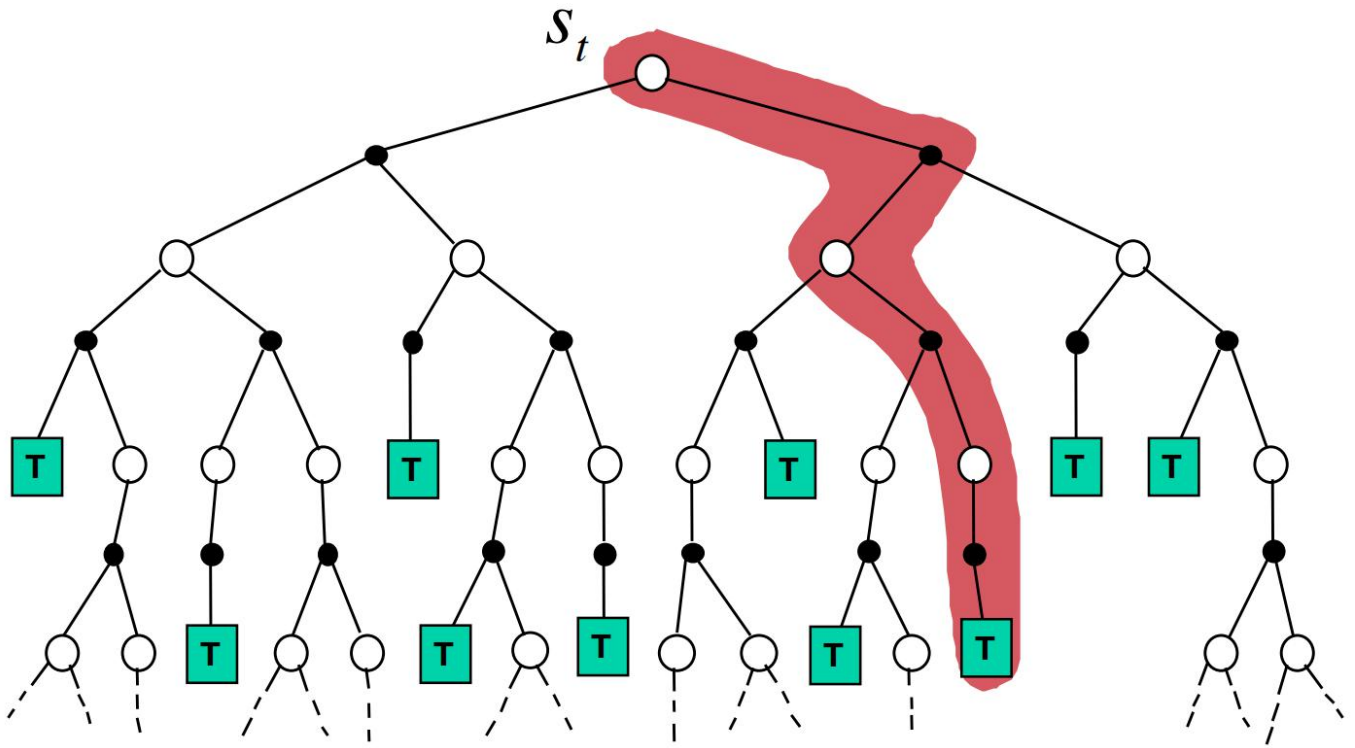
MC和TD的优劣势（3）

- **TD算法使用了MDP问题的马尔科夫属性，在Markov 环境下更有效；**
- **MC算法并不利用马尔可夫属性，通常在非Markov环境下更有效。**

时序差分学习

MC采样

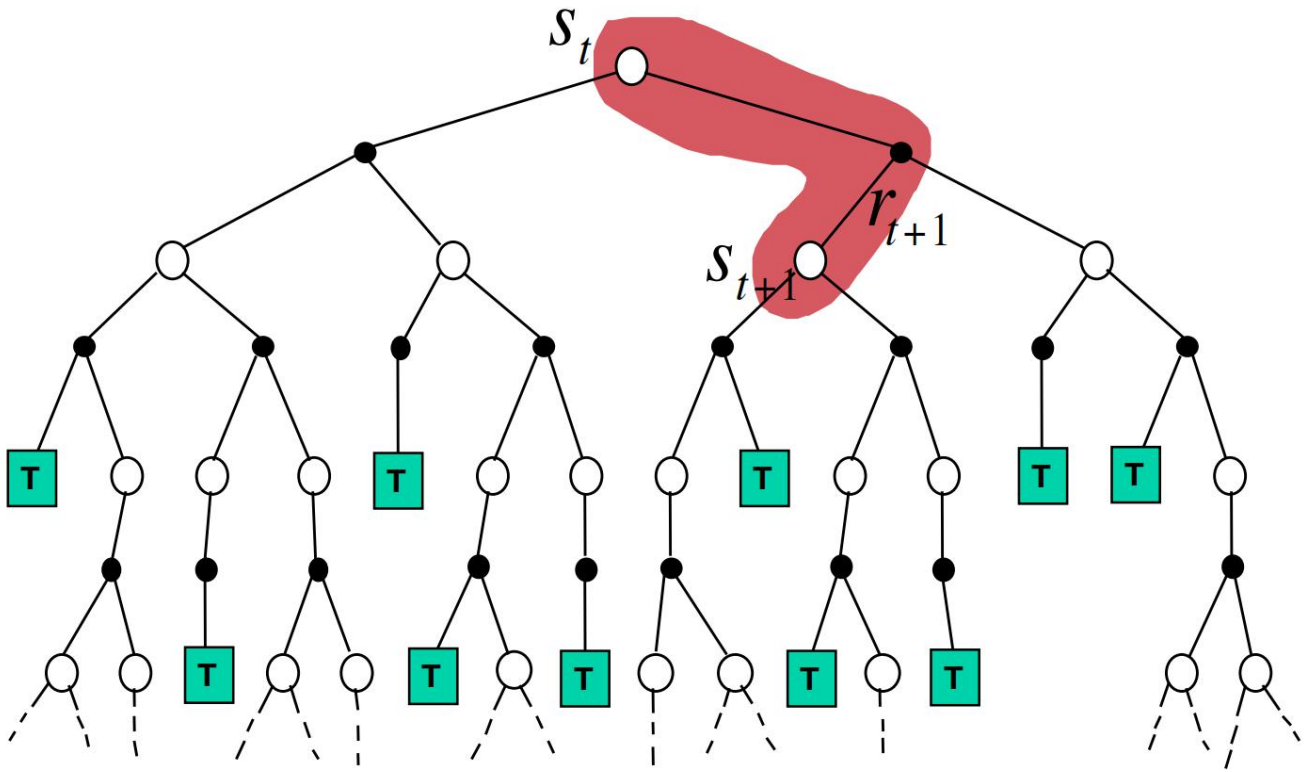
$$V(S_t) \leftarrow V(S_t) + \alpha\left(G_t - V(S_t)\right)$$



时序差分学习

TD采样

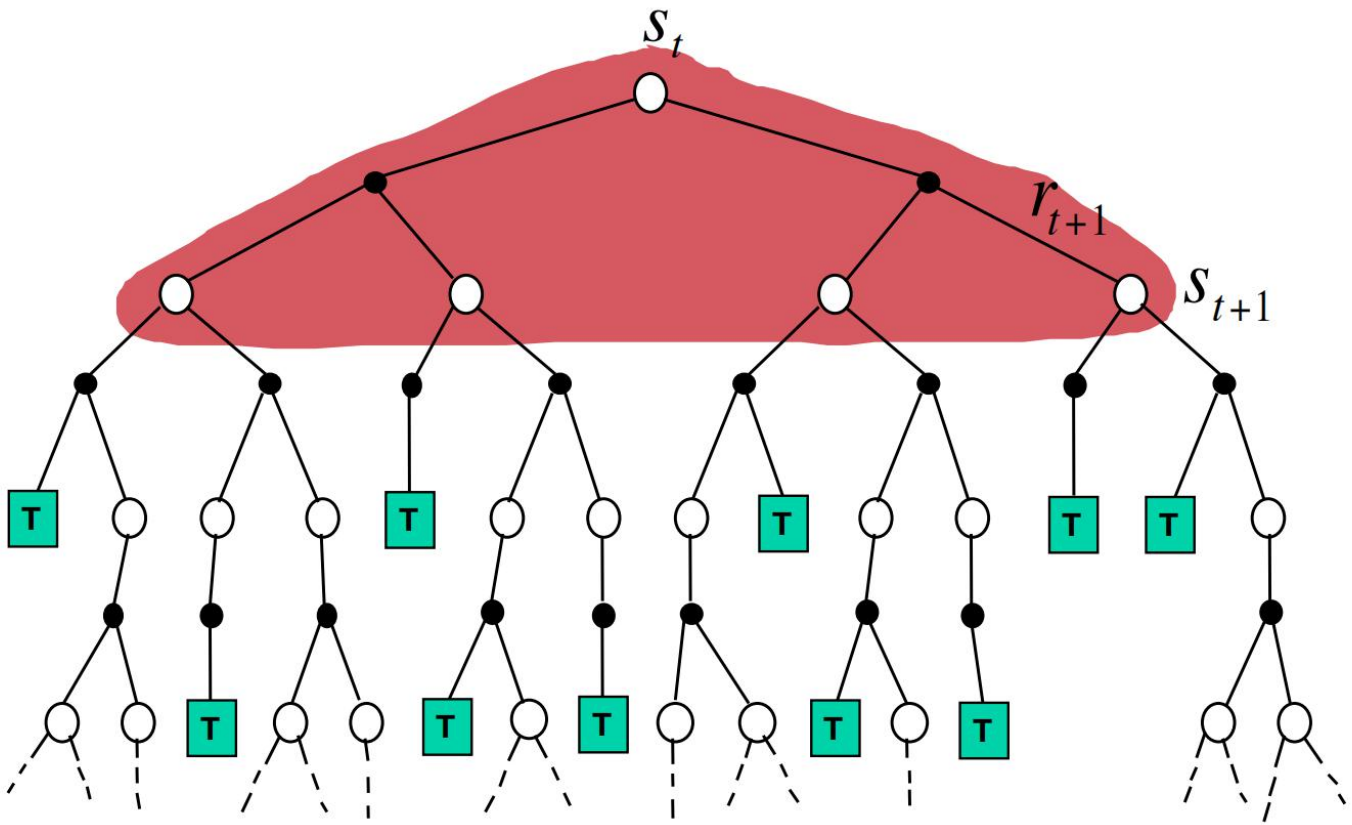
$$V(S_t) \leftarrow V(S_t) + \alpha\left(R_{t+1} + \gamma V(S_{t+1}) - V(S_t)\right)$$



时序差分学习

DP：没有采样，根据完整模型，依靠预估数据更新状态价值

$$V(S_t) \leftarrow \mathbb{E}_\pi \left[R_{t+1} + \gamma V(S_{t+1}) \right]$$



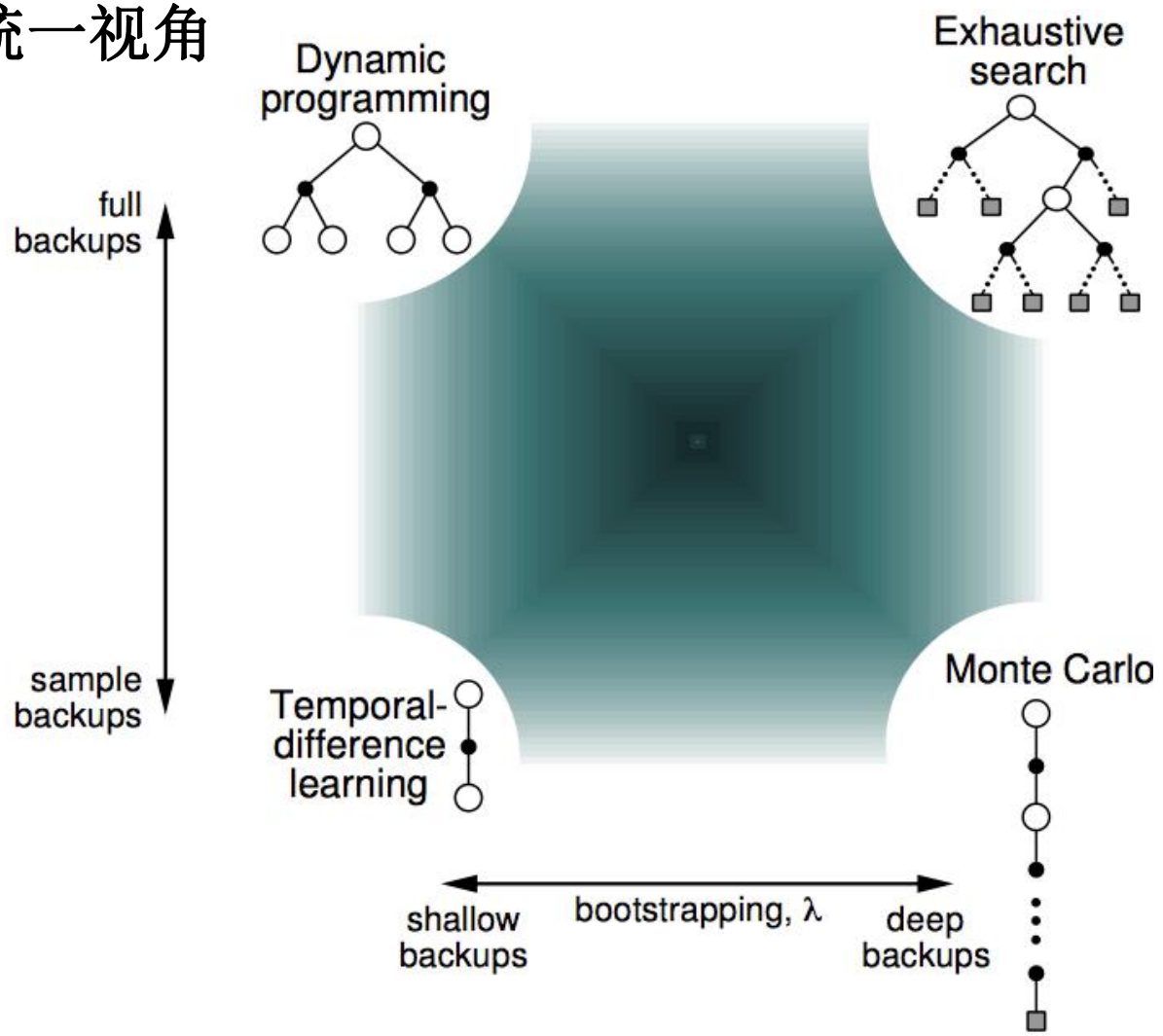
时序差分学习

自举（Bootstrapping）和采样（Sampling）

- 自举（Bootstrapping）：更新涉及之前的估计
  - MC没有自举
  - DP有自举
  - TD有自举
- 采样（Sampling）：更新涉及采样
  - MC有采样
  - DP没有采样
  - TD有采样

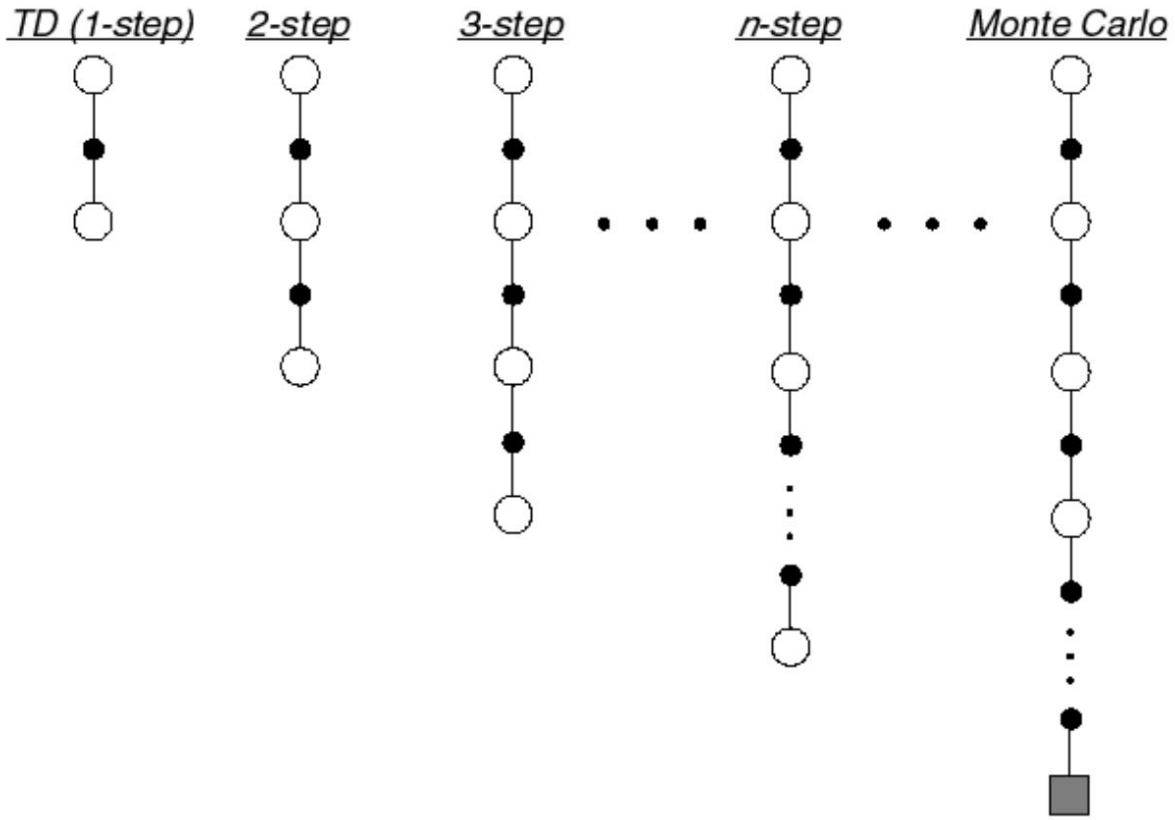
时序差分学习

强化学习各算法的统一视角



TD(λ)

先前所介绍的TD算法实际上都是TD(0)算法，括号内的数字0表示的是在当前状态下往前多看1步，要是往前多看2步更新状态价值会怎样？



TD(λ)

n-步回报

- TD或TD(0)是基于1-步预测的，MC则是基于∞-步预测的：

$$
\begin{aligned}
n=1 \quad (TD) \quad & G_t^{(1)} = R_{t+1} + \gamma V(S_{t+1}) \\
n=2 \quad & G_t^{(2)} = R_{t+1} + \gamma R_{t+2} + \gamma^2 V(S_{t+2}) \\
\vdots \quad & \vdots \\
n=\infty \quad (MC) \quad & G_t^{(\infty)} = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{T-1} R_T
\end{aligned}
$$

- 定义n-步回报

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

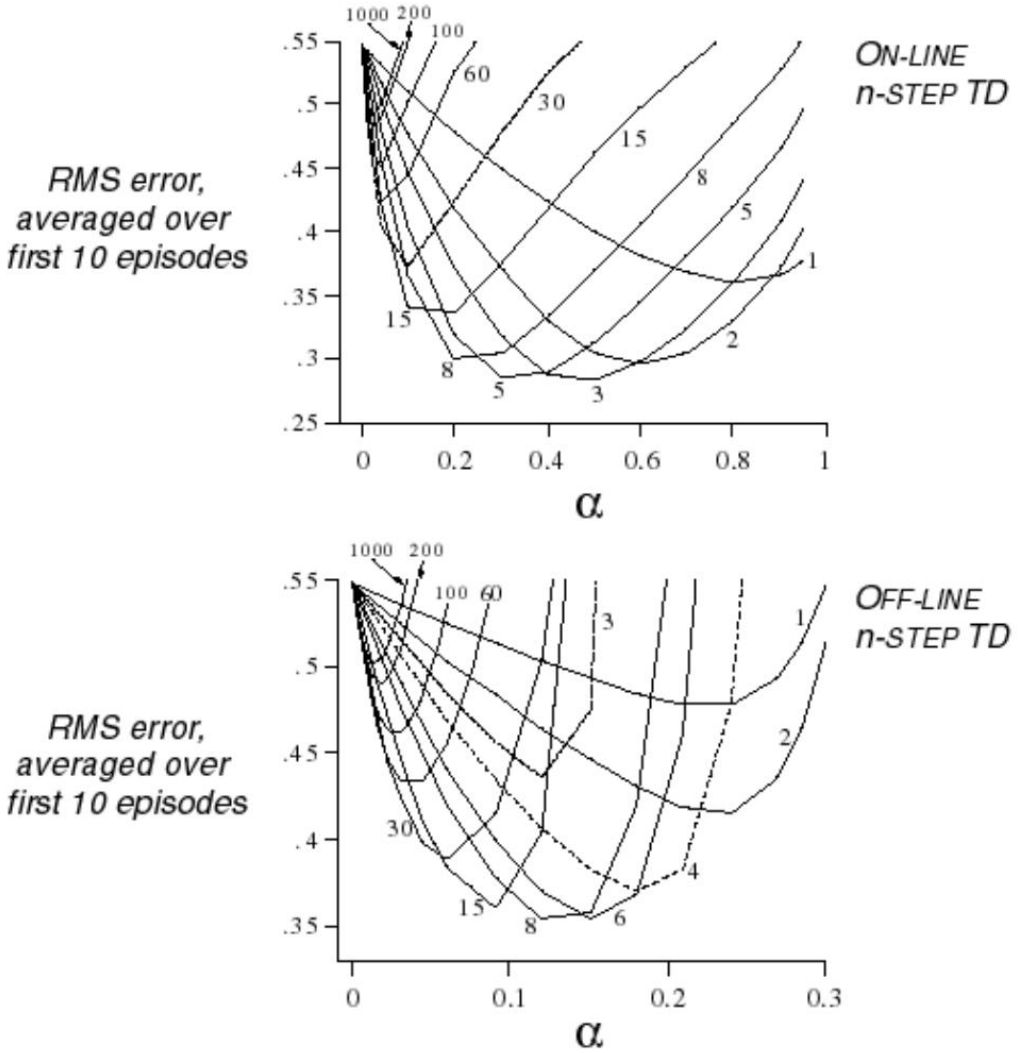
- n-步TD学习

$$V(S_t) \leftarrow V(S_t) + \alpha \left(G_t^{(n)} - V(S_t) \right)$$

TD(λ)

示例——大规模随机游走

离线：经历所有10个episode后进行状态价值更新；
在线：至多经历一个episode就更新依次状态价值。



(1)离线和在线曲线形态差别不明显
(2)从步数来看，步数越长约接近MC，均方差越大。（在线3-5步最佳，离线6-8步最佳）
(3)最优步数对应于问题

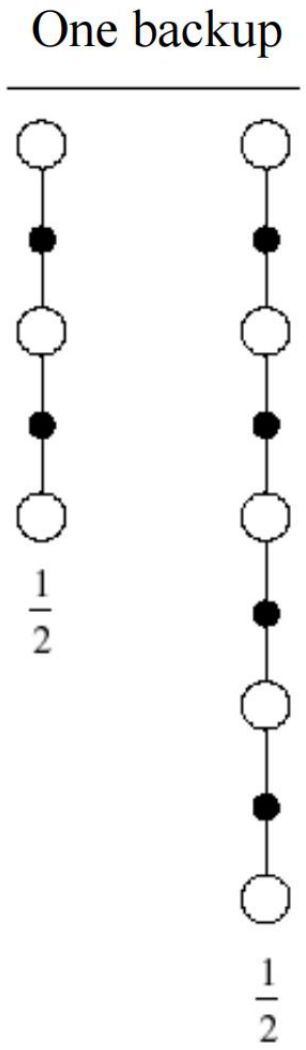
TD(λ)

平均n-步回报

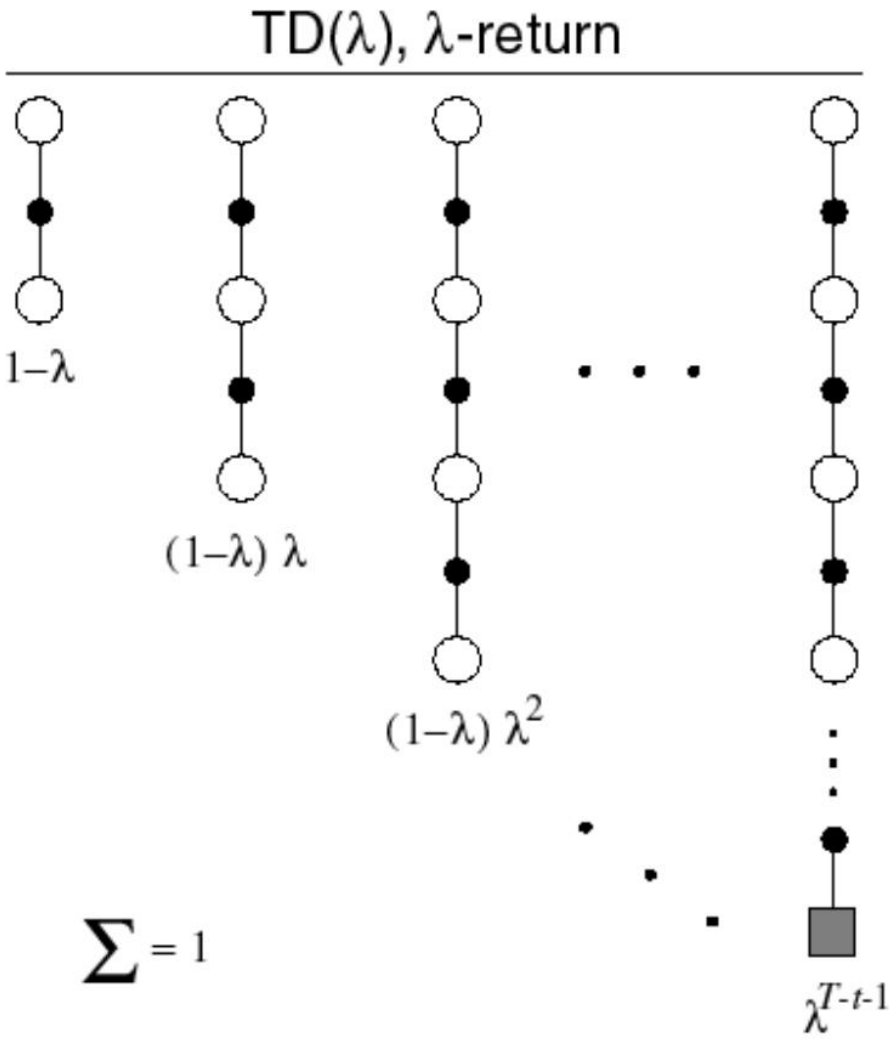
- 我们可以对不同的n步回报取平均，比如对2-步回报和4-步回报取平均：

$$\frac{1}{2}G^{(2)} + \frac{1}{2}G^{(4)}$$

- 结合两个不同步数的回报信息，是不是我们可以更有效率地从所有的步数中收集回报信息？



TD(λ)



- λ-回报 G_t^λ 结合了所有的n-步回报 $G_t^{(n)}$
- 使用权重 $(1-\lambda)\lambda^{n-1}$

$$G_t^\lambda = (1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}G_t^{(n)}$$

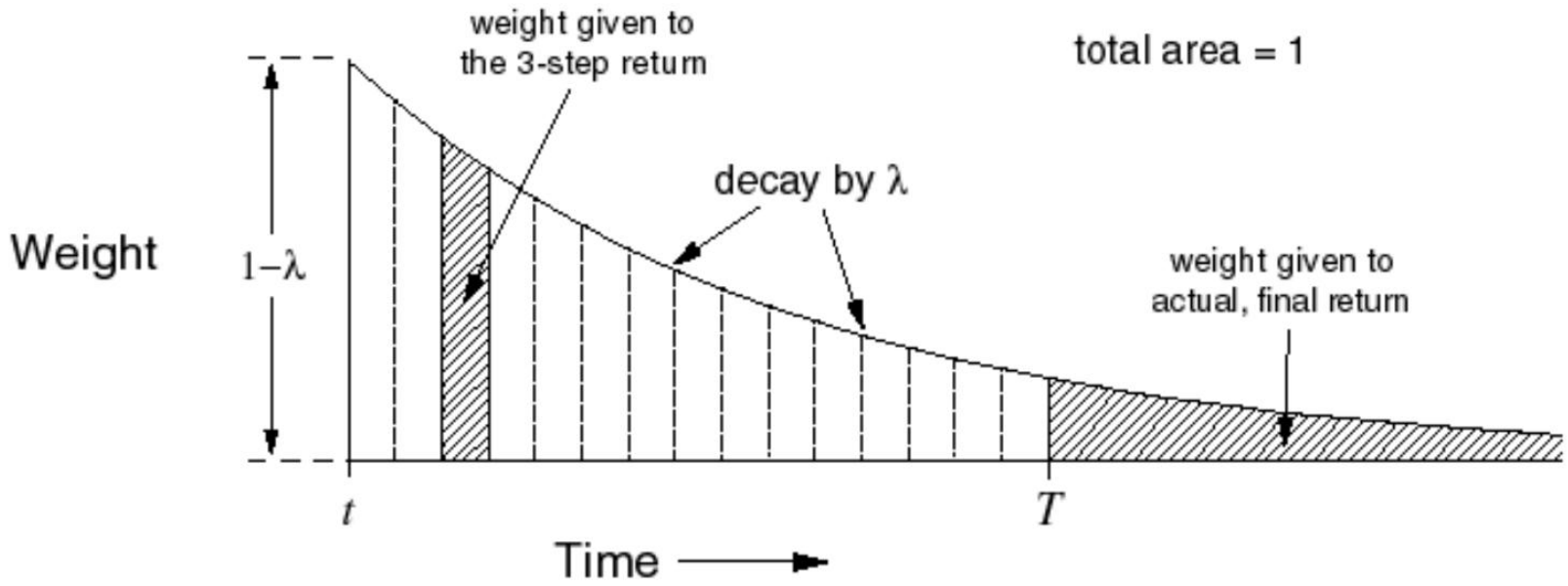
- 前向认识TD(λ)

$$V(S_t) \leftarrow V(S_t) + \alpha(G_t^\lambda - V(S_t))$$

λ=0时，退化成TD(0)
λ=1时，退化成MC

TD(λ)

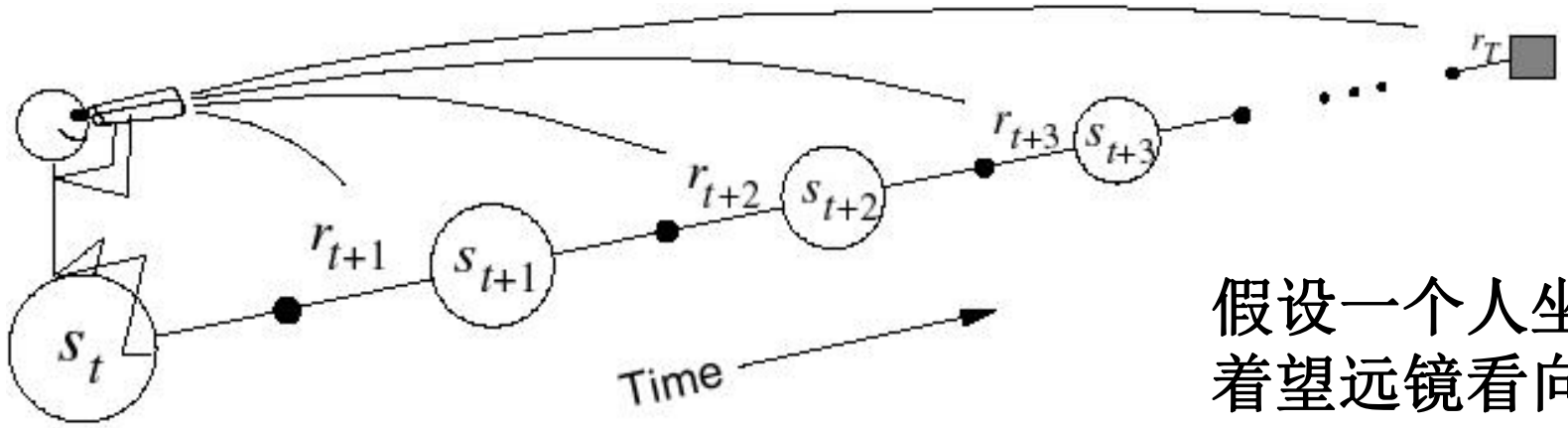
TD(λ)加权函数



$$G_t^{\lambda} = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}$$

TD(λ)

前向视角认识TD(λ)



假设一个人坐在状态流上拿着望远镜看向前方，前方是那些将来的状态。当估计当前状态的值函数时，从TD(λ)的定义中可以看到，它需要用到将来时刻的值函数。

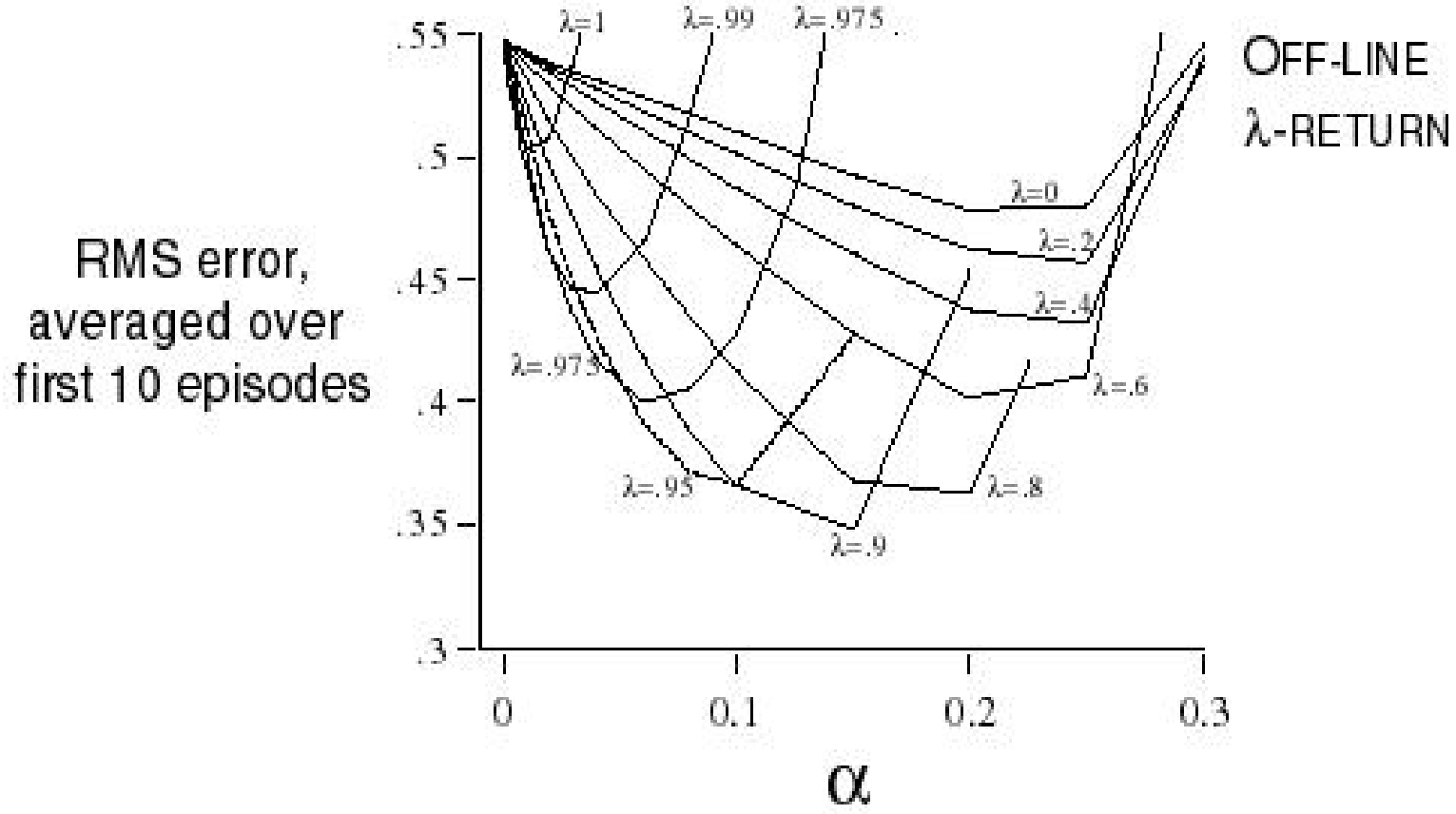
- 面向 λ-回报更新值函数
- 前向视角展望未来以计算 G_t^λ
- 与MC类似，但是MC只能计算完整的episodes

TD(λ)

前向视角认识TD(λ)在大规模随机游走例子的结果

λ=0时，TD(0)

λ=1时，MC



TD(λ)

反向视角认识TD(λ)

- 前向视角提供理论
- 反向视角提供机制
- 在线更新、每一步更新、从不完整序列中更新

TD(λ)

资格迹（Eligibility Traces）

这是之前见过的一个例子，老鼠在连续接受了3次响铃和1次亮灯信号后遭到了电击，那么在分析遭电击的原因时，到底是响铃的因素较重要还是亮灯的因素更重要呢？（信度分配）

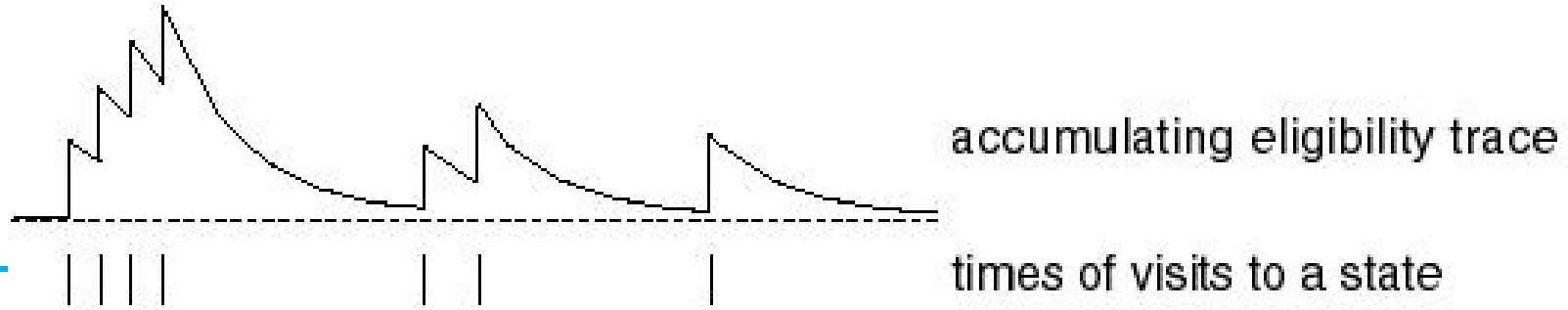
- 频率启发 Frequency heuristic：将原因归因于出现频率最高的状态
- 就近启发 Recency heuristic：将原因归因于较近的几次状态

资格迹结合了这两种启发：

$$E_0(s) = 0$$

每一个状态引入一个资格迹

$$E_t(s) = \gamma\lambda E_{t-1}(s) + 1(S_t = s)$$

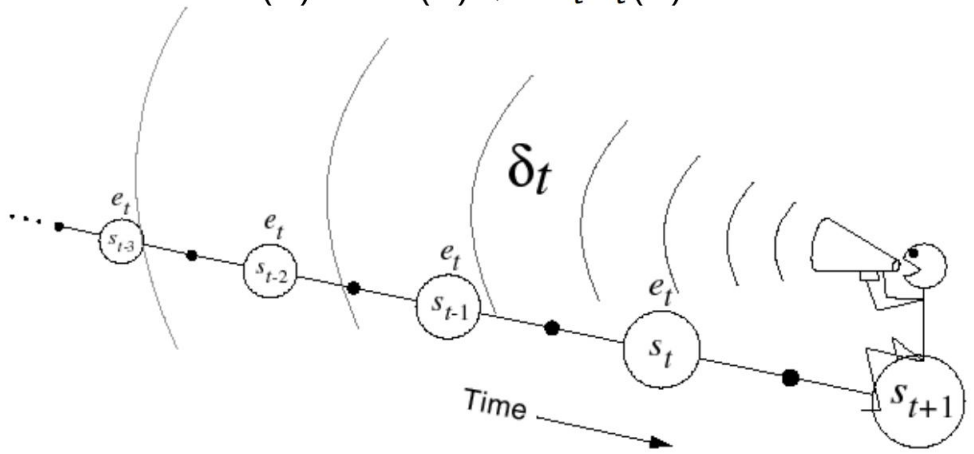


TD(λ)

反向视角TD(λ)

- 对于每一个状态s保持一个资格迹
- 对于每个状态s更新价值函数V(s)
- 与 TD 误差 δ_t 和资格迹 $E_t(s)$ 成比例

$$\delta_t = R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$$
$$V(s) \leftarrow V(s) + \alpha \delta_t E_t(s)$$



TD(λ)

TD(λ) 和 TD(0)

- 当λ=0时，只有当前的状态得到更新

$$E_t(s) = \mathbf{1}(S_t = s)$$
$$V(s) \leftarrow V(s) + \alpha\delta_t E_t(s)$$

- 这完全等同于 **TD(0)** 更新

$$V(S_t) \leftarrow V(S_t) + \alpha\delta_t$$

TD(λ)

TD(λ) and MC

- 当λ=1时，信用被推迟到episode结束
- 考虑具有离线更新的情节环境
- 在一个episode的过程中，TD(1) 的总更新数与 MC 的总更新数相同

定理

前向视角和反向视角 TD(λ) 的离线更新总和相同

$$\sum_{t=1}^{T} \alpha \delta_t E_t(s) = \sum_{t=1}^{T} \alpha (G_t^{\lambda} - V(S_t)) 1(S_t = s)$$

TD(λ)

MC 和 TD(1)

- 考虑在时间步 k 处访问 s 一次的episode，
- 自访问以来TD(1) 资格迹随时间折扣，

$$E_t(s) = \gamma E_{t-1}(s) + \mathbf{1}(S_t = s)$$
$$= \begin{cases} 0 & \text{if } t < k \\ \gamma^{t-k} & \text{if } t \geq k \end{cases}$$

- TD(1)更新在线累积误差

$$\sum_{t=1}^{T-1} \alpha \delta_t E_t(s) = \alpha \sum_{t=k}^{T-1} \gamma^{t-k} \delta_t = \alpha \left(G_k - V(S_k) \right)$$

- 到episode结束时，它累积了总误差

$$\delta_k + \gamma \delta_{k+1} + \gamma^2 \delta_{k+2} + \ldots + \gamma^{T-1-k} \delta_{T-1}$$

TD(λ)

当 λ = 1 时，TD 的误差总和可伸缩为 MC 误差

$$
\begin{aligned}
&\delta_t + \gamma\delta_{t+1} + \gamma^2\delta_{t+2} + \ldots + \gamma^{T-1-t}\delta_{T-1} \\
&= R_{t+1} + \gamma V(S_{t+1}) - V(S_t) \\
&+ \gamma R_{t+2} + \gamma^2 V(S_{t+2}) - \gamma V(S_{t+1}) \\
&+ \gamma^2 R_{t+3} + \gamma^3 V(S_{t+3}) - \gamma^2 V(S_{t+2}) \\
&\quad\vdots \\
&+ \gamma^{T-1-t} R_T + \gamma^{T-t} V(S_T) - \gamma^{T-1-t} V(S_{T-1}) \\
&= R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \ldots + \gamma^{T-1-t} R_T - V(S_t) \\
&= G_t - V(S_t)
\end{aligned}
$$

TD(λ)

TD(λ) 和 TD(1)

- **TD(1) 大致相当于每次访问 Monte-Carlo**
- 误差是在线累积、循序渐进的
- **如果值函数仅在episode结束时离线更新，那么总更新和MC完全一样**

TD(λ)

对于一般 λ，TD 误差也可伸缩到 λ 误差，$G_t^\lambda - V(S_t)$

$$
\begin{aligned}
G_t^\lambda - V(S_t) = -V(S_t) \quad &+ \quad (1-\lambda)\lambda^0 \left(R_{t+1} + \gamma V(S_{t+1})\right) \\
&+ \quad (1-\lambda)\lambda^1 \left(R_{t+1} + \gamma R_{t+2} + \gamma^2 V(S_{t+2})\right) \\
&+ \quad (1-\lambda)\lambda^2 \left(R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 V(S_{t+3})\right) \\
&+ \quad \ldots \\
= -V(S_t) \quad &+ \quad (\gamma\lambda)^0 \left(R_{t+1} + \gamma V(S_{t+1}) - \gamma\lambda V(S_{t+1})\right) \\
&+ \quad (\gamma\lambda)^1 \left(R_{t+2} + \gamma V(S_{t+2}) - \gamma\lambda V(S_{t+2})\right) \\
&+ \quad (\gamma\lambda)^2 \left(R_{t+3} + \gamma V(S_{t+3}) - \gamma\lambda V(S_{t+3})\right) \\
&+ \quad \ldots \\
= \quad &\quad (\gamma\lambda)^0 \left(R_{t+1} + \gamma V(S_{t+1}) - V(S_t)\right) \\
&+ \quad (\gamma\lambda)^1 \left(R_{t+2} + \gamma V(S_{t+2}) - V(S_{t+1})\right) \\
&+ \quad (\gamma\lambda)^2 \left(R_{t+3} + \gamma V(S_{t+3}) - V(S_{t+2})\right) \\
&+ \quad \ldots \\
= \delta_t + \gamma\lambda\delta_{t+1} + (\gamma\lambda)^2\delta_{t+2} + \ldots &
\end{aligned}
$$

TD(λ)

前向和反向 TD(λ)

- 当在时间步 k 处访问 s 的一次episode，
- 自访问以来TD(λ) 资格迹折扣的时间，

$$E_t(s) = \gamma\lambda E_{t-1}(s) + \mathbf{1}(S_t = s)$$
$$= \begin{cases} 0 & \text{if } t < k \\ (\gamma\lambda)^{t-k} & \text{if } t \geq k \end{cases}$$

- 反向 TD(λ) 更新在线累积误差

$$\sum_{t=1}^{T} \alpha\delta_t E_t(s) = \alpha \sum_{t=k}^{T} (\gamma\lambda)^{t-k}\delta_t = \alpha\left(G_k^{\lambda} - V(S_k)\right)$$

- 到episode结束时，它累积了 λ-return 的总误差
- 对于多次访问 s，$E_t(s)$ 会累积很多误差

TD(λ)

前向和反向TD的离线等价

- 离线更新
  - 更新在episode中累积
  - 但在episode结束时批量应用

TD(λ)

前向和反向 TD 在线等价

- 在线更新
  - TD(λ) 更新在episode中的每一步都在线应用
  - 前视和后视 TD(λ) 略有不同
  - 新：精确在线 TD(λ) 实现完美等价
  - 通过使用稍微不同形式的资格迹
  - Sutton 和 von Seijen，ICML 2014

TD(λ)

前向视角和反向视角 TD(λ)总结

Offline updates	$\lambda = 0$	$\lambda \in (0,1)$	$\lambda = 1$
Backward view	TD(0)	TD(λ)	TD(1)
	$=$	$=$	$=$
Forward view	TD(0)	Forward TD(λ)	MC
Online updates	$\lambda = 0$	$\lambda \in (0,1)$	$\lambda = 1$
Backward view	TD(0)	TD(λ)	TD(1)
	$=$	$\neq$	$\neq$
Forward view	TD(0)	Forward TD(λ)	MC
	$=$	$=$	$=$
Exact Online	TD(0)	Exact Online TD(λ)	Exact Online TD(1)

= 这里表示在episode结束时总更新的等效性。

The End