多智能体强化学习实训
Multi-agent Reinforcement Learning

Lecture 2:
Value Estimation

教师：张寅 zhangyin98@zju.edu.cn

助教：邓悦 devindeng@zju.edu.cn

王子瑞 ziseoiwong@zju.edu.cn

李成林 chenglinli@zju.edu.cn

浙江大学计算机学院

课程大纲

- **蒙特卡洛方法**
- 时序差分方法
- 资格迹方法
- 表格型时序差分方法：
  - Q-learning
  - SARSA

回顾MDP建模

- 在马尔科夫奖励过程中加入行为，就得到了马尔科夫决策过程(Markov Decision Process, MDP)，MDP由5个元素组成$M =< S, A, P, R, \gamma >$
  - S:状态集合
  - A:行为集合
  - P:状态转移函数，$p(s'|s, a)$取决于状态和行为
  - R:奖励函数，$r(s, a)$取决于状态和行为，如果只取决于状态则退化到$r(s)$
  - γ:未来奖励折扣因子。

蒙特卡洛方法

- 在现实问题中，通常我们不能假设我们对环境有完全的了解，即无法明确地给出状态转移和奖励函数。因此我们需要一种直接从经验数据中学习价值和策略，无需构建马尔可夫决策过程模型的方法。

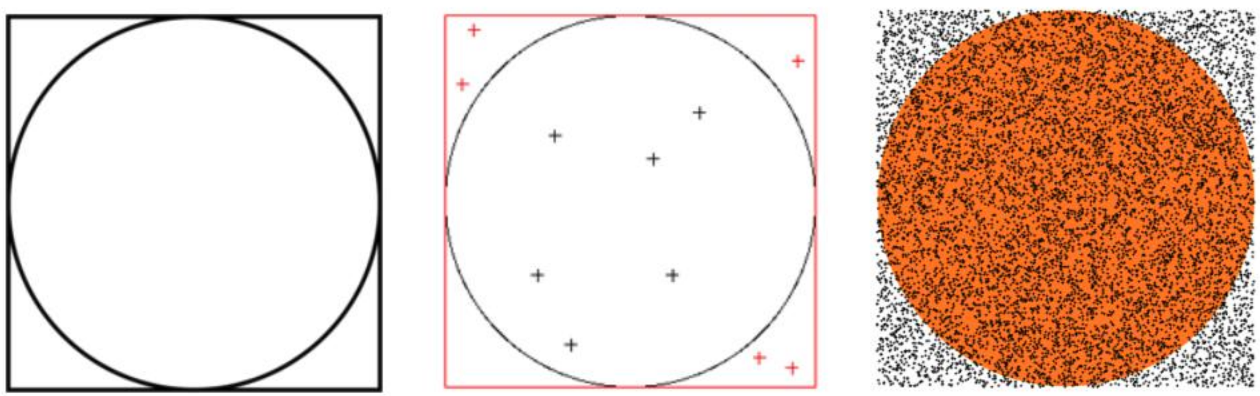
- 在这种情况下，智能体只能和环境进行交互，通过采样到的数据来学习，这类学习方法统称为无模型的强化学习（model-free reinforcement learning）。

蒙特卡洛方法

- 蒙特卡洛方法是一种基于随机采样和统计的方法，得名于摩纳哥的蒙特卡洛赌场，因为这种方法使用了大量的随机模拟。蒙特卡洛方法在强化学习中的基本思想是通过多次采样来估计状态或动作的值函数，随后利用值函数进行策略改进。

蒙特卡洛方法

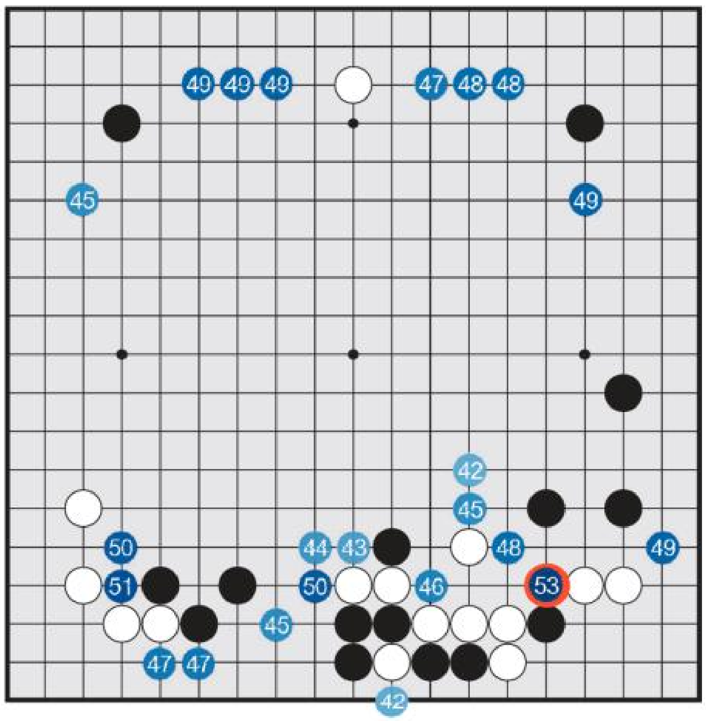
- 蒙特卡洛方法（Monte-Carlo methods，简称 MC）是一类广泛的计算算法。蒙特卡洛方法依赖重复随机抽样来获得数值结果。
- 例如，计算圆的面积

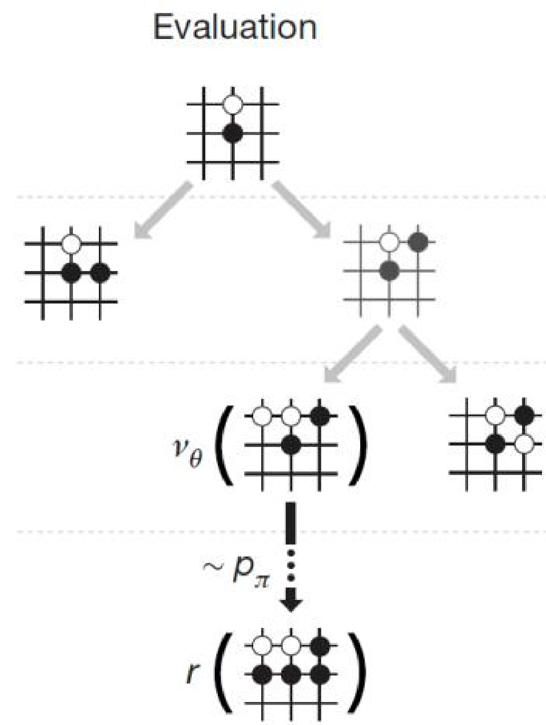


$$\text{圆的面积} \approx \text{正方形面积} \times \frac{\text{圆内点的数量}}{\text{正方形内点的数量}}$$

蒙特卡洛方法

- 例如，AlphaGO中AI对围棋落子胜率的估计





$$当前棋局的胜率 \approx \frac{对弈胜利局数}{对弈总局数}$$（从当前棋局出发）

蒙特卡洛方法

- 目标：从策略π采样的历史经验中估计V^{π}
- 回顾设定：累计奖励（return）是总折扣奖励

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

- 回顾设定：值函数（value function）是期望累计奖励

$$V^{\pi}(s) = \mathbb{E}\left[\mathrm{R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \ldots} \middle| \mathrm{s_0 = s, \pi}\right]$$

$$= \mathbb{E}[\mathrm{G_t | s_0 = s, \pi}] \approx \frac{1}{\mathrm{N}} \sum_{i=1}^{N} \mathrm{G_t^i}$$

蒙特卡洛方法

- 目标：从策略π采样的历史经验中估计V^{π}
- 回顾设定：累计奖励（return）是总折扣奖励

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

- 回顾设定：值函数（value function）是期望累计奖励

$$V^{\pi}(s) = \mathbb{E}[\mathrm{R}(\mathrm{s_0}) + \gamma \mathrm{R}(\mathrm{s_1}) + \gamma^2 \mathrm{R}(\mathrm{s_2}) + \ldots | \mathrm{s_0} = \mathrm{s}, \pi]$$

$$= \mathbb{E}[\mathrm{G_t} | \mathrm{s_0} = \mathrm{s}, \pi] \approx \frac{1}{\mathrm{N}} \sum_{i=1}^{N} \mathrm{G_t^i}$$

蒙特卡洛方法使用策略π从状态s采样N个样本，并使用经验均值累计奖励近似期望累计奖励。

蒙特卡洛方法

- 具体实现：使用策略π采样时间步数量为T的多个回合

$$s_0^i \xrightarrow[a_0^i]{R_1^i} s_1^i \xrightarrow[a_1^i]{R_2^i} \rightarrow \ldots \rightarrow s_T^i \sim \pi$$

- 对于每一个回合中的时间步t中的状态s
  - 更新访问次数 $N(s) \leftarrow N(s) + 1$
  - 更新 return 的总和 $S(s) \leftarrow S(s) + G_t$
  - 估计 return 的均值 $V(s) = S(s)/N(s)$
  - 由大数定律，当 $N(s) \rightarrow \infty$ 有 $V(s) \rightarrow V^{\pi}(s)$

蒙特卡洛方法

- 将公式整理为增量更新的形式，可得

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

$$V(s_t) \leftarrow V(s_t) + (\mathrm{G_t} - V(s_t))/N(s_t)$$

- 增量式公式

$$\begin{aligned} Q_k &= \frac{1}{k}\sum_{i=1}^{k} r_i \\ &= \frac{1}{k}\left(r_k + \sum_{i=1}^{k-1} r_i\right) \\ &= \frac{1}{k}(r_k + (k-1)Q_{k-1}) \\ &= \frac{1}{k}(r_k + kQ_{k-1} - Q_{k-1}) \\ &= Q_{k-1} + \frac{1}{k}[r_k - Q_{k-1}] \end{aligned}$$

蒙特卡洛方法

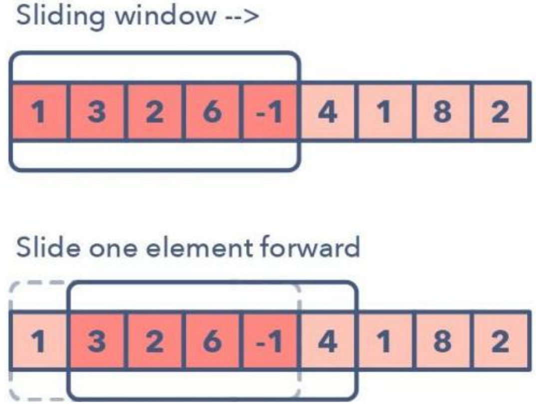
- 将公式整理为增量更新的形式，可得

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

$$V(s_t) \leftarrow V(s_t) + (\mathrm{G_t} - V(s_t))/N(s_t)$$

- 对于非平稳的环境（即环境的动态会随时间发生变化），蒙特卡洛方法可以跟踪一个滑动窗口内的平均值

$$V(s_t) \leftarrow V(s_t) + \alpha(\mathrm{G_t} - V(s_t))$$



蒙特卡洛方法总结

- 直接从经验回合进行学习，不需要模拟/搜索
- 模型无关（model-free），无需环境信息
- 核心思想简单直白：value = mean return
- 使用完整回合进行更新：只能应用于有限长度的马尔可夫决策过程，即所有的回合都应有终止状态。

蒙特卡洛方法

- 示例：二十一点（Blackjack）
  - 状态空间
    - 当前点数之和（12 ~ 21）
    - Dealer展示手牌（Ace ~ 10）
    - 玩家是否有Ace（Yes or No）

蒙特卡洛方法

- 示例：二十一点（Blackjack）
  - 动作空间：
    - “停止”：停止接收卡（回合终止）
    - “继续”：再拿一张卡（无放回）

蒙特卡洛方法

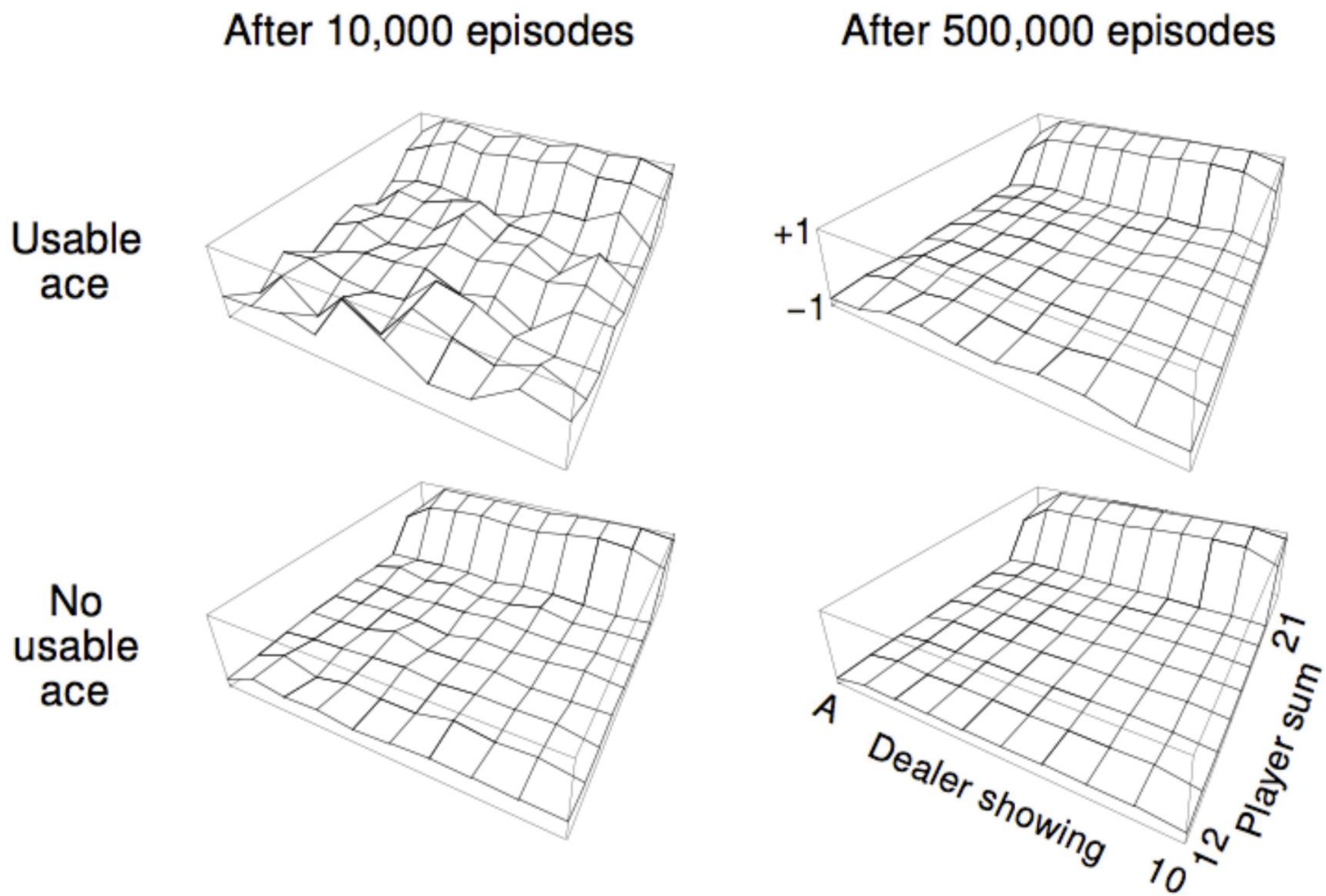
- 示例：二十一点（Blackjack）
  - 奖励设定（“停止”动作）：
    - +1：卡牌点数总和>Dealer
    - 0 ：卡牌点数总和=Dealer
    - -1 ：卡牌点数总和<Dealer

蒙特卡洛方法

- 示例：二十一点（Blackjack）
  - 奖励设定（“继续”动作）：
    - -1:卡牌点数总和>21（游戏终止）
    - 0:其他情况
  - 如果卡牌点数<12自动执行“继续”动作

蒙特卡洛方法

- 给定策略：若手牌点数>=20，执行“停止”动作，否则执行“继续”动作



课程大纲

- 蒙特卡洛方法
- **时序差分方法**
- 资格迹方法
- 表格型时序差分方法：
  - Q-learning
  - SARSA

回顾价值函数V

- 马尔科夫奖励过程中，一个状态的**期望**回报被成为该状态的价值，所有的状态的价值的集合为价值函数，即价值函数$V(s)$以状态作为输入，以其期望回报为输出。
- 该价值函数可以写成为：$V(s) = \mathbb{E}[G_t|S_t = s]$
- 将该函数展开：

$$V(s) = \mathbb{E}[G_t|S_t = s]$$

$$= \mathbb{E}[R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \ldots|S_t = s]$$

$$= \mathbb{E}[R_t + \gamma(R_{t+1} + \gamma R_{t+2} + \ \ldots)|S_t = s]$$

$$= \mathbb{E}[R_t + \gamma V(S_{t+1})|S_t = s]$$

$$= r(s) + \gamma \sum_{s' \in S} p(s'|s)V(s')$$

时序差分方法

$$G_t = R_{t+1} + \gamma R_{t+2} + \dots + \gamma^{T-1} R_T = R_{t+1} + \gamma V(s_{t+1})$$

$$V(s_t) \leftarrow V(s_t) + \alpha\big(\underset{\substack{\uparrow \\ \text{观测值}}}{R_{t+1}} + \gamma V(\underset{\substack{\uparrow \\ \text{对未来的猜测}}}{s_{t+1}}) - V(s_t)\big)$$

- 时序差分方法（ Temporal Difference methods，简称 TD ）能够直接使用经验回合学习，同样也是模型无关的；

时序差分方法

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

$$V(s_t) \leftarrow V(s_t) + \alpha(\underset{\substack{\uparrow \\ \text{观测值}}}{R_{t+1}} + \gamma V(\underset{\substack{\uparrow \\ \text{对未来的猜测}}}{s_{t+1}}) - V(s_t))$$

- 时序差分方法（ Temporal Difference methods，简称 TD ）能够直接使用经验回合学习，同样也是模型无关的；
- 与蒙特卡洛方法不同，时序差分方法结合了自举（bootstrapping），能从不完整的回合中学习；

时序差分方法

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

$$V(s_t) \leftarrow V(s_t) + \alpha\big(\underset{\substack{\uparrow \\ \text{观测值}}}{R_{t+1}} + \gamma V(\underset{\substack{\uparrow \\ \text{对未来的猜测}}}{s_{t+1}}) - V(s_t)\big)$$

- 时序差分方法（ Temporal Difference methods，简称 TD ）能够直接使用经验回合学习，同样也是模型无关的;
- 与蒙特卡洛方法不同，时序差分方法结合了自举（bootstrapping），能从不完整的回合中学习；
- 时序差分通过更新当前预测值，使之接近估计 return，而非真实 return。

时序差分方法 vs. 蒙特卡洛方法

- 目标：从策略π采样的历史经验中估计V^{π}
- 蒙特卡洛方法更新值函数$V(s_t)$ 使其接近G_t ：

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

- 时序差分方法更新值函数$V(s_t)$ 使其接近$R_{t+1} + \gamma V(s_{t+1})$：

$$V(s_t) \leftarrow V(s_t) + \alpha(R_{t+1} + \gamma V(s_{t+1}) - V(s_t))$$

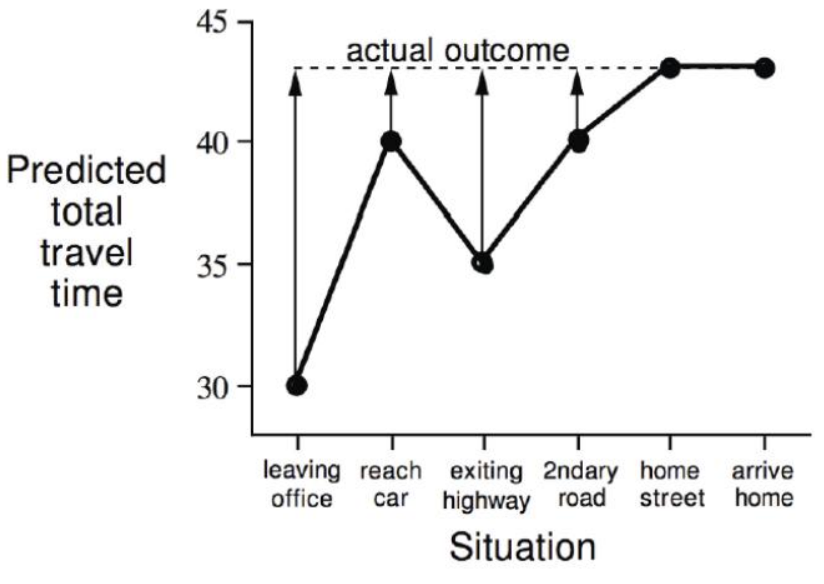
- $R_{t+1} + \gamma V(s_{t+1})$ 为时序差分目标；
- $\delta_t = R_{t+1} + \gamma V(s_{t+1}) - V(s_t)$ 为时序差分误差；

时序差分方法 vs. 蒙特卡洛方法

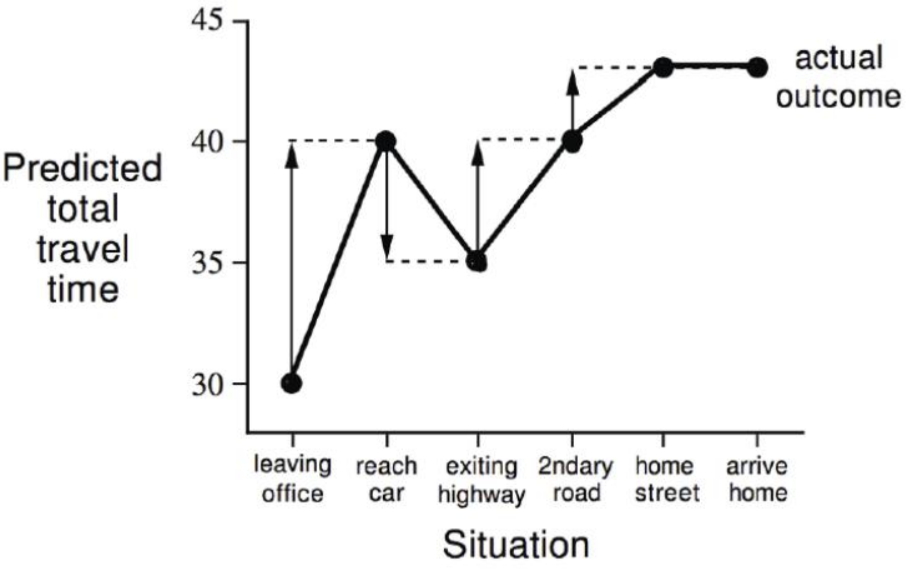
状态	经过的时间 (分钟)	预计所剩时间	预计总时间
离开公司	0	30	30
开始驾车， 下雨	5	35	40
离开高速公路	20	15	35
卡车后跟车	30	10	40
到达家所在街道	40	3	43
直奔家门	43	0	43

时序差分方法 vs. 蒙特卡洛方法

蒙特卡洛方法



时序差分方法



时序差分方法 vs. 蒙特卡洛方法

- 时序差分方法能够在每一步之后进行在线学习，蒙特卡洛方法必须等待回合终止，直到累计奖励已知；

时序差分方法 vs. 蒙特卡洛方法

- 时序差分方法能够在每一步之后进行在线学习，蒙特卡洛方法必须等待回合终止，直到累计奖励已知；
- 时序差分方法能够从不完整的序列中学习，蒙特卡洛方法只能从完整序列中学习；

时序差分方法 vs. 蒙特卡洛方法

- 时序差分方法能够在每一步之后进行在线学习，蒙特卡洛方法必须等待回合终止，直到累计奖励已知；
- 时序差分方法能够从不完整的序列中学习，蒙特卡洛方法只能从完整序列中学习；
- 时序差分方法能够应用于无限长度的马尔可夫决策过程，蒙特卡洛方法只适用于有限长度；

时序差分方法 vs. 蒙特卡洛方法

- 累计奖励$G_t = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{T-1} R_T$是$V^{\pi}$的无偏估计
- 时序差分的真实目标$\underbrace{R_{t+1} + \gamma V^{\pi}(s_{t+1})}$也是$V^{\pi}$的无偏估计

  ↑
  当前估计
- 时序差分目标$R_{t+1} + \gamma V(s_{t+1})$是 V^{π}的有偏估计

时序差分方法 vs. 蒙特卡洛方法

- 累计奖励$G_t = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{T-1} R_T$是$V^{\pi}$的无偏估计
- 时序差分的真实目标$R_{t+1} + \gamma V^{\pi}(s_{t+1})$也是$V^{\pi}$的无偏估计

  当前估计
- 时序差分目标$R_{t+1} + \gamma V(s_{t+1})$是 V^{π}的有偏估计

- 时序差分目标具有更低的方差，这是因为
  - 累计奖励取决于多步随机动作、状态转移和奖励
  - 时序差分取决于单步随机动作、状态转移和奖励

时序差分方法 vs. 蒙特卡洛方法

时序差分方法

$$V(s_t) \leftarrow V(s_t) + \alpha(R_{t+1} + \gamma V(s_{t+1}) - V(s_t))$$

- 低方差，有偏差
  - 更高效
  - 最终收敛到V^{π}
  - 对初始值更敏感

蒙特卡洛方法

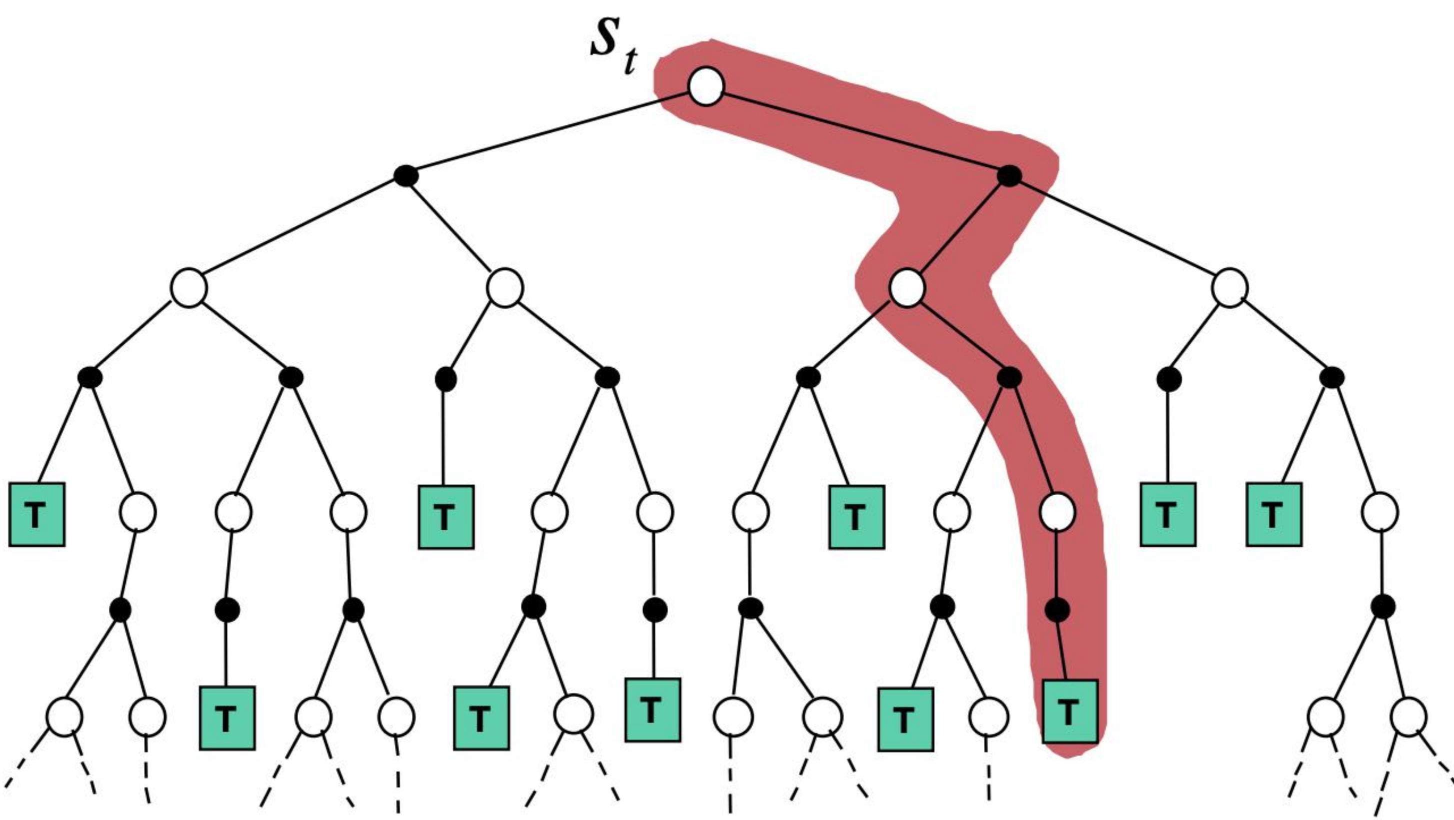
$$V(s_t) \leftarrow V(s_t) + \alpha(G_t - V(s_t))$$

- 高方差，无偏差
  - 良好收敛性
  - 对初始值不敏感
  - 易于理解和使用

时序差分方法 vs. 蒙特卡洛方法

- 蒙特卡洛方法

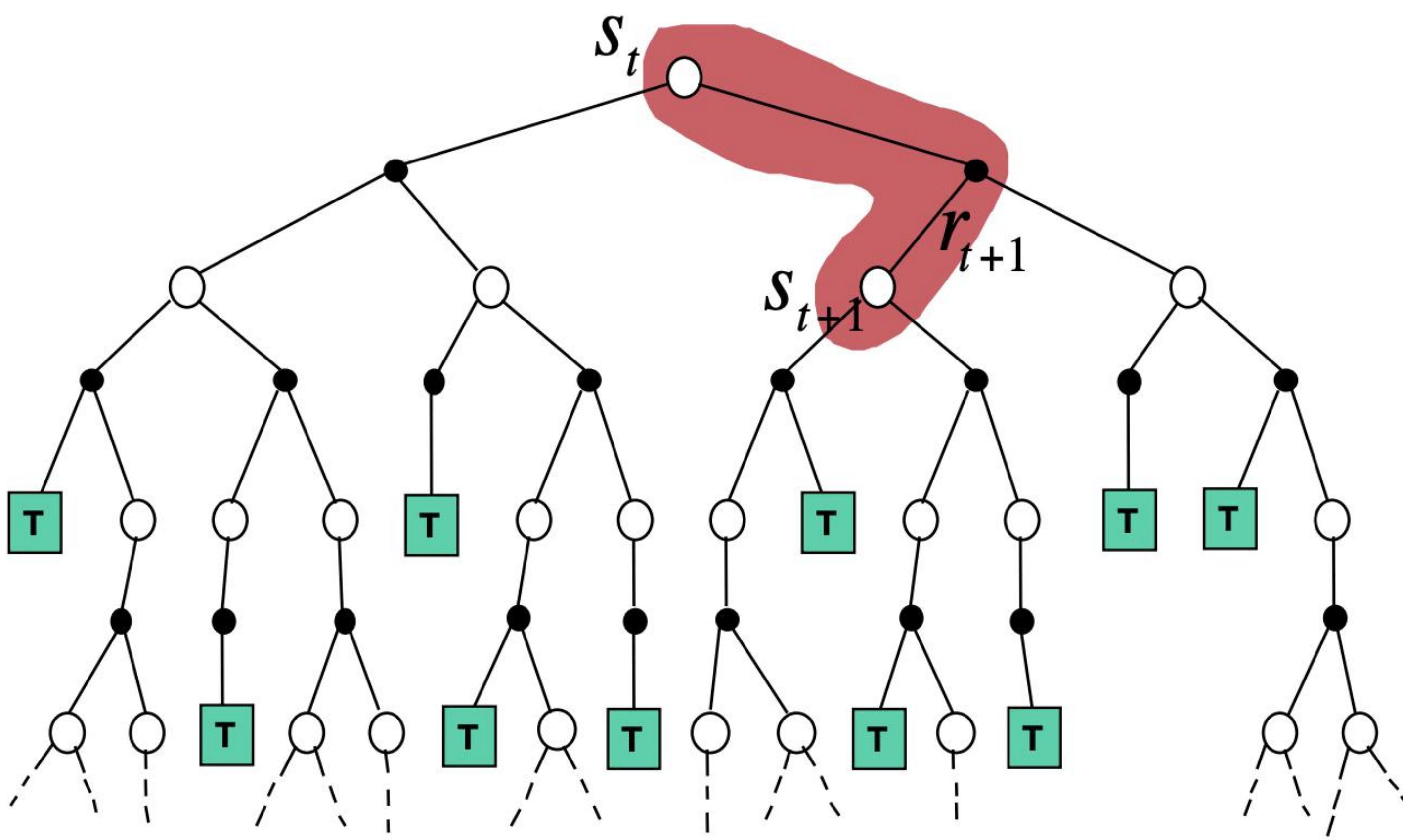
$$V(S_t) \leftarrow V(S_t) + \alpha \left(G_t - V(S_t)\right)$$



时序差分方法 vs. 蒙特卡洛方法

- 时序差分方法

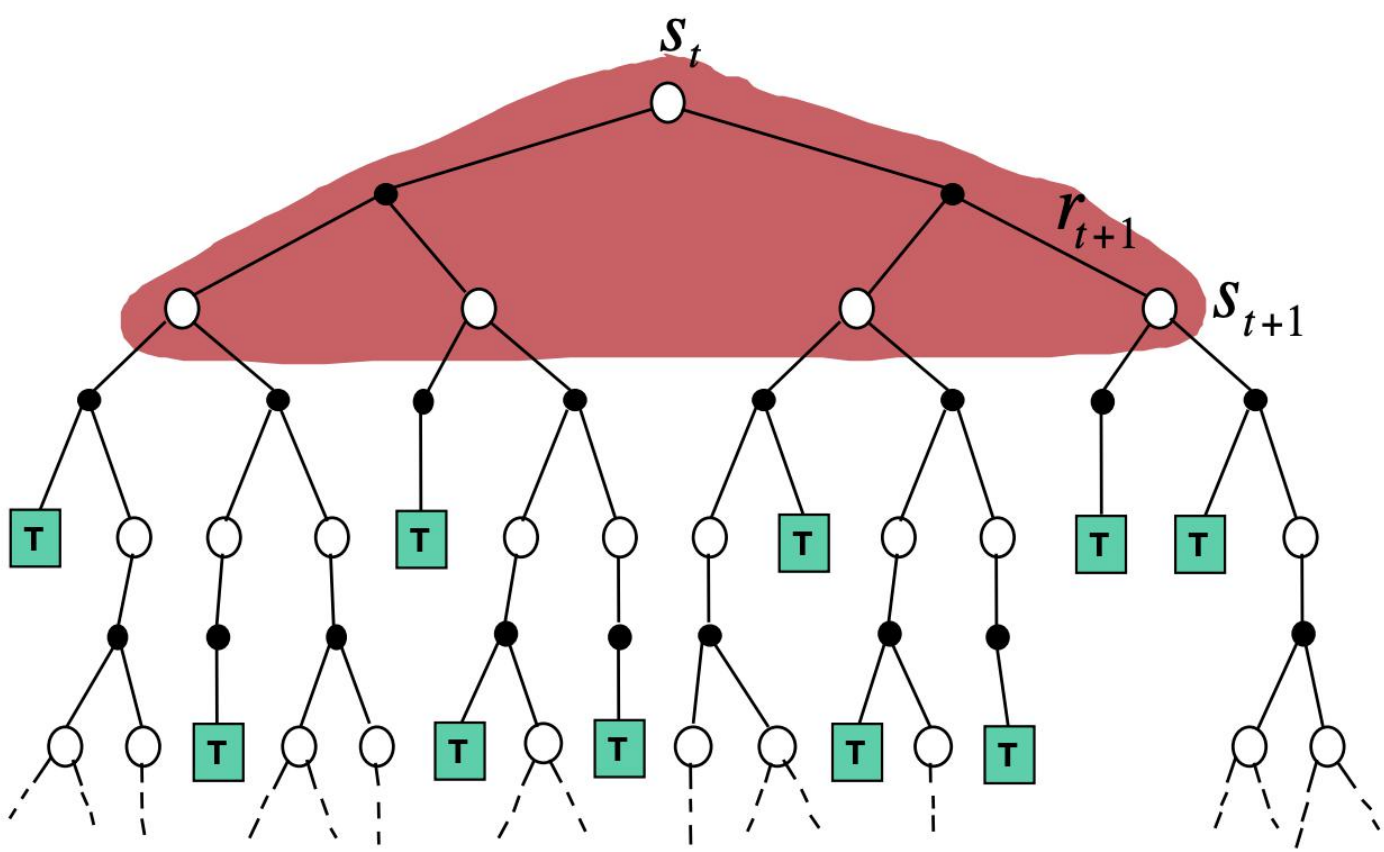
$$V(S_t) \leftarrow V(S_t) + \alpha\left(R_{t+1} + \gamma V(S_{t+1}) - V(S_t)\right)$$



时序差分方法 vs. 蒙特卡洛方法

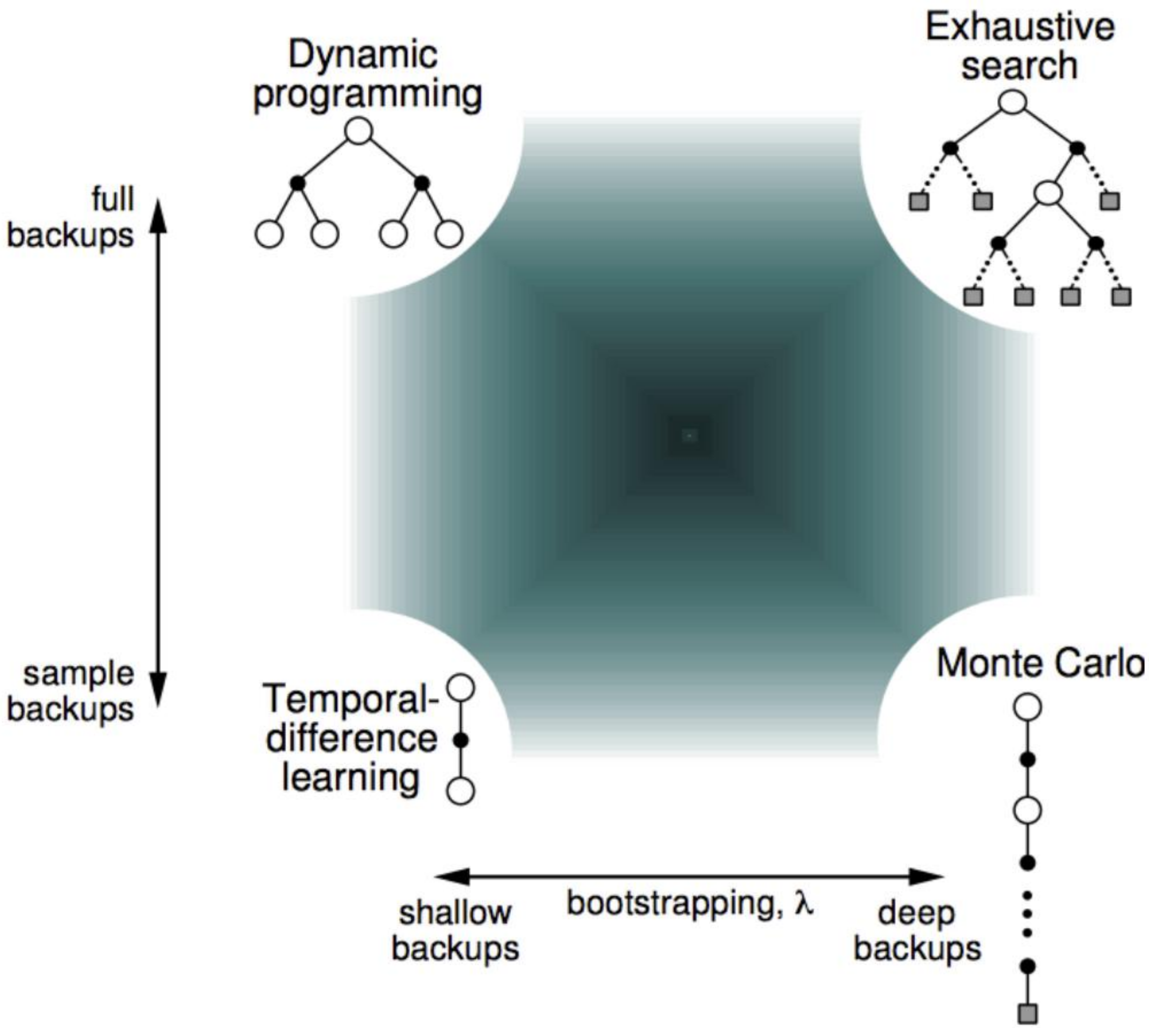
- 动态规划

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



时序差分方法 vs. 蒙特卡洛方法

- 统一视角



课程大纲

- 蒙特卡洛方法
- 时序差分方法
- 资格迹方法
- 表格型时序差分方法：
  - Q-learning
  - SARSA

资格迹方法

- 蒙特卡洛方法（高方差，无偏差）

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

$$G_t = R_{t+1} + \gamma R_{t_2} + \gamma^2 R_{t+3} + \dots + \gamma^{T-t-1} R_T$$

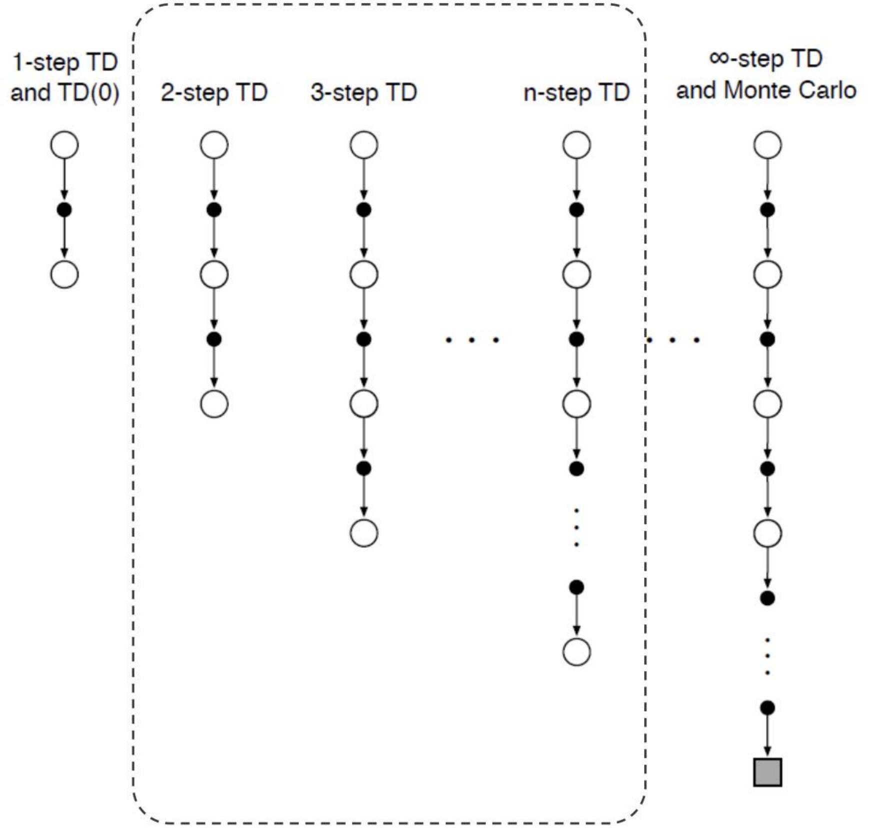
- 时序差分方法（低方差，有偏差）

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

有没有方法介于两者之间？

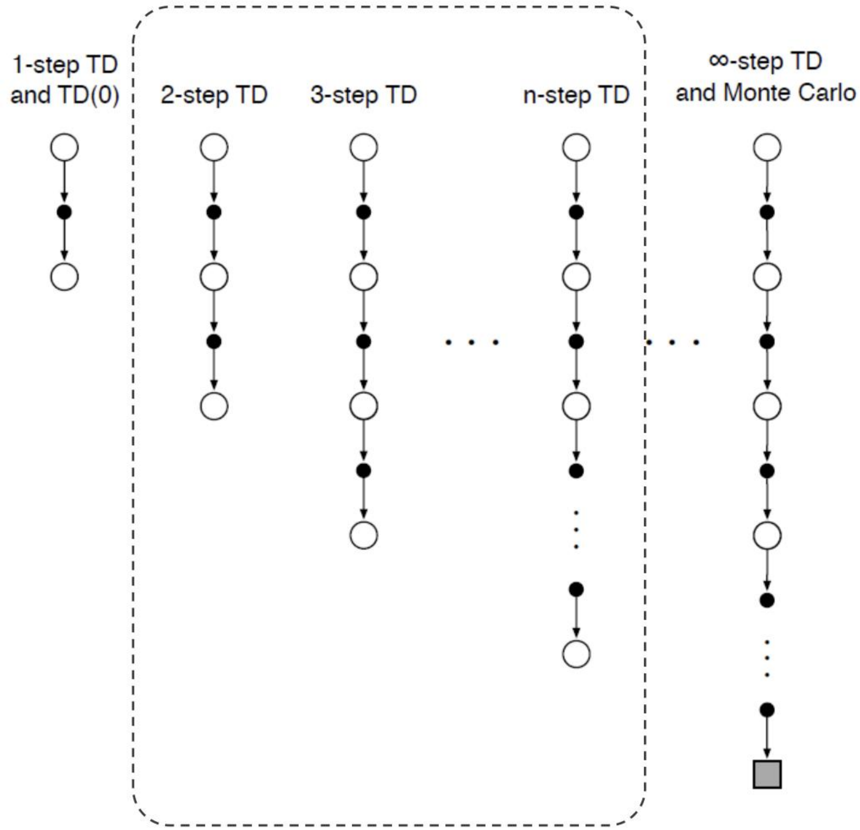
资格迹方法

- 多步时序差分方法



资格迹方法

- 多步时序差分方法



- 无限步时序差分方法等效于蒙特卡洛方法

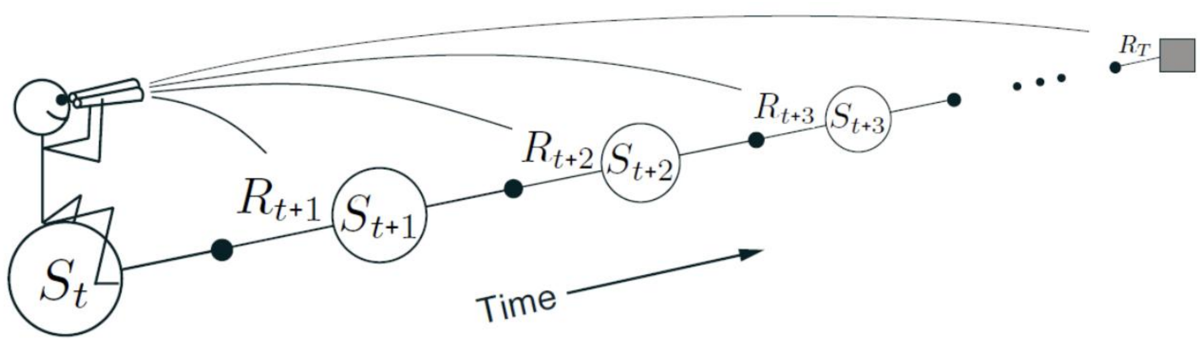
资格迹方法

- 为了实现方差与偏差的平衡，一种可行的方案是将蒙特卡洛方法与时序差分方法融合，实现多步时序差分；
- 定义n步累计奖励：

$$\boldsymbol{G_t^n} = R_{t+1} + \gamma R_{t+2} + \dots + \gamma^{n-1} R_{t+n} + \gamma^n V(s_{t+n})$$

- 可得出n步时序差分学习：

$$V(s_t) \leftarrow V(s_t) + \alpha(\boldsymbol{G_t^n} - V(s_t))$$



资格迹方法

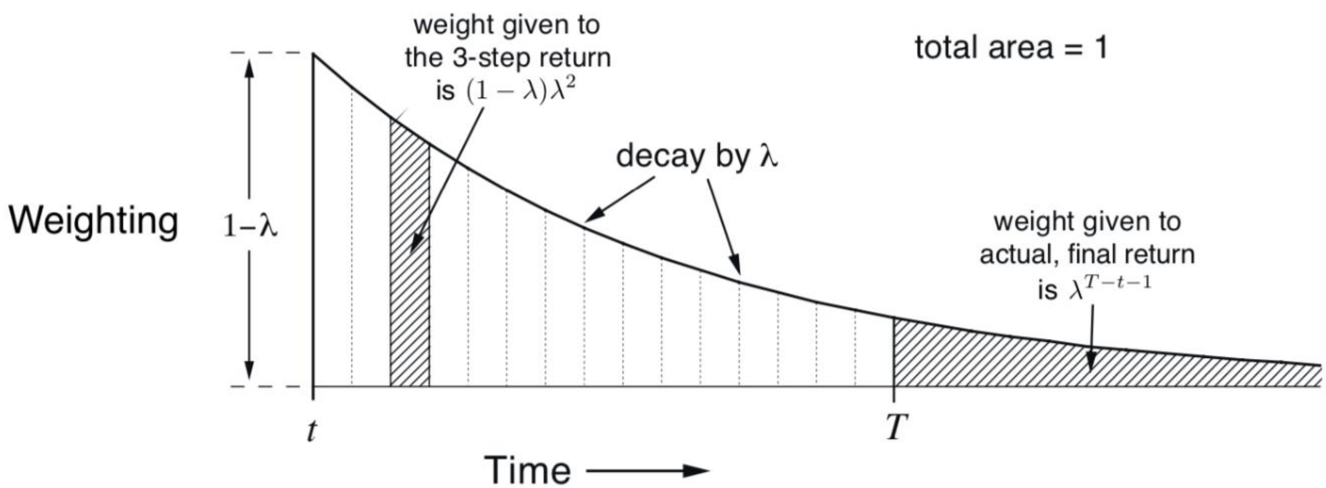
- 资格迹方法（Eligibility Traces methods）统一了时序差分和蒙特卡罗方法；
- 资格迹方法通常使用超参数$\lambda \in [0, 1]$控制值估计蒙特卡罗还是时序差分，通常来说，当$\lambda = 1$时资格迹方法等价于蒙特卡罗方法，当$\lambda = 0$等价于时序差分方法；
- 介于两者之间的方法通常比任何一种极端方法都要好。

资格迹方法

- TD-λ是一种比较常见的资格迹方法，不同时序差分估计值的权重随着其时间步的增加而衰减：

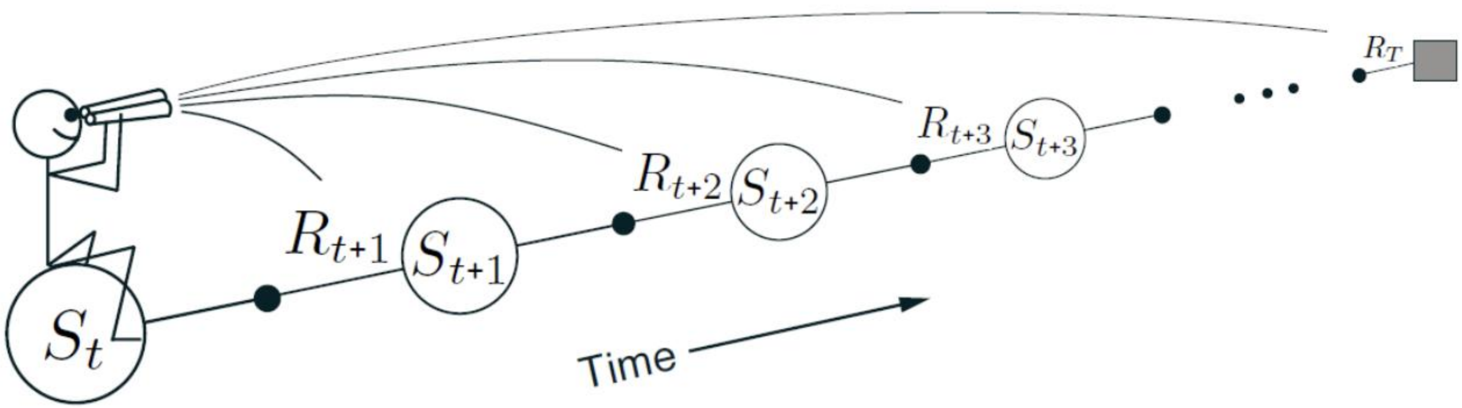
$$G_t^n = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{n-1} R_{t+n} + \gamma^n V(s_{t+n})$$

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



资格迹方法

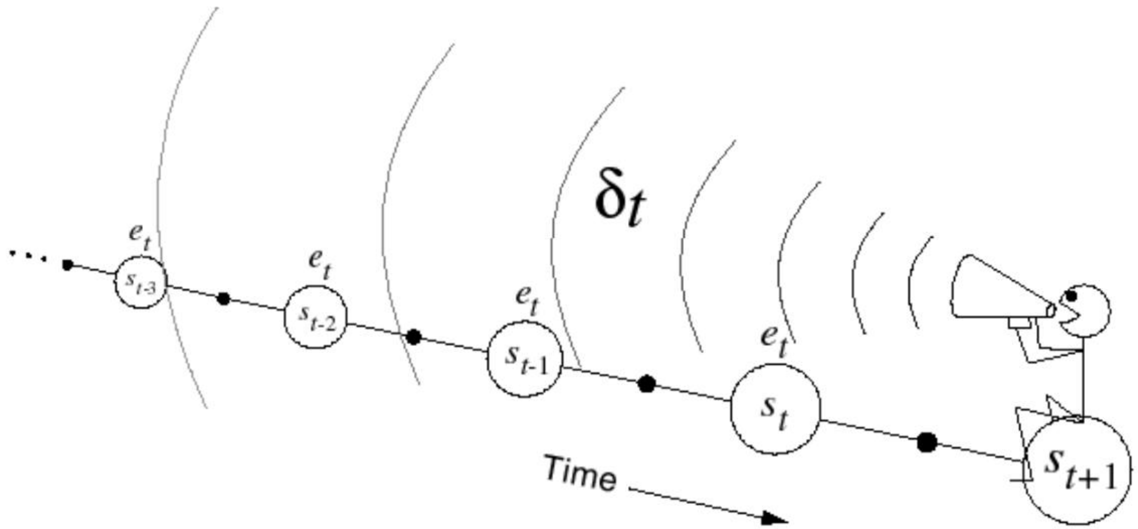
- TD-λ的前向视角：



  - 朝着G_t^λ的方向更新值函数；
  - 通过观测未来的数据计算G_t^λ；
  - 与蒙特卡洛方法类似，只能计算完整的回合。

资格迹方法

- TD-λ的后向视角：



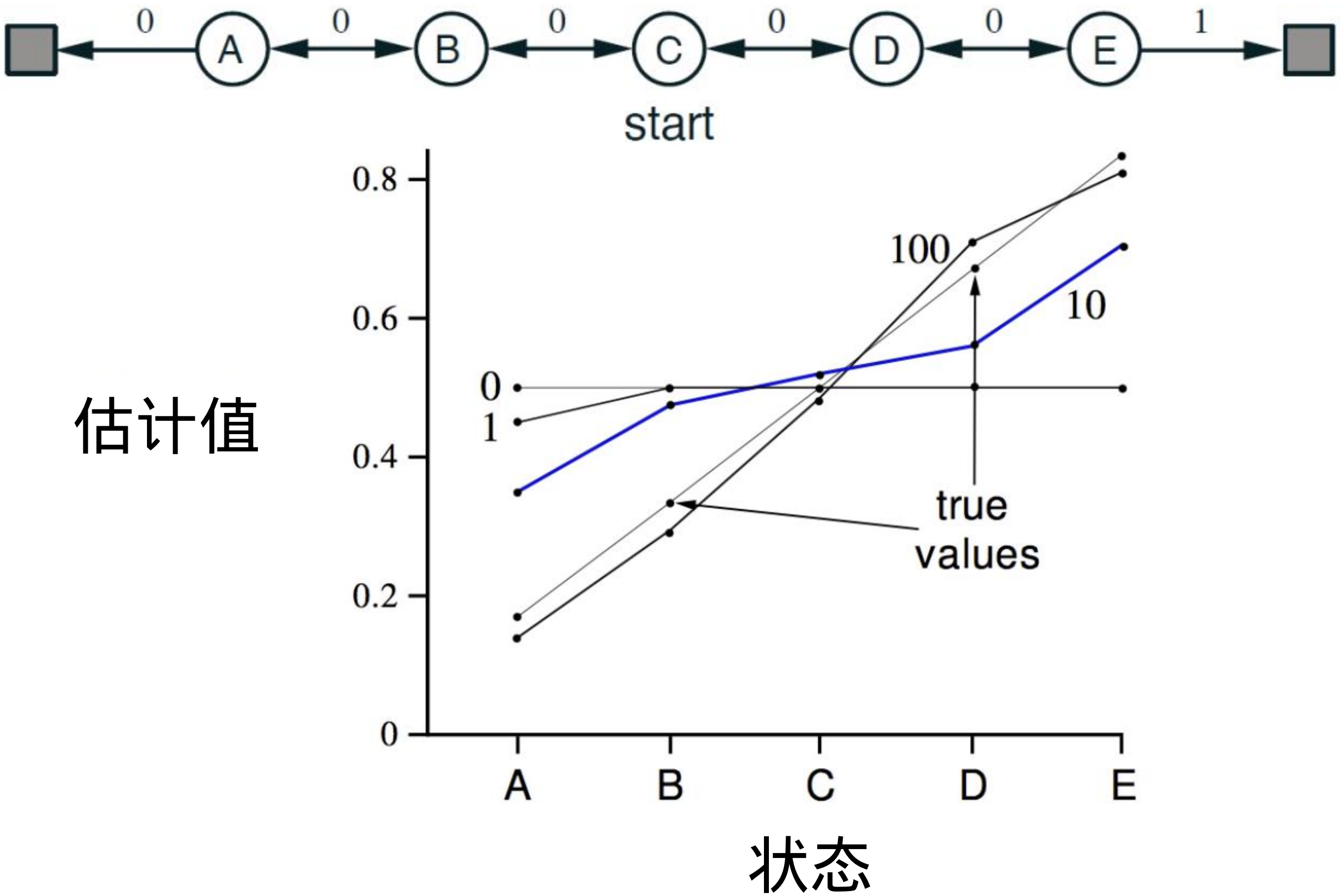
  - 对每个状态s保持资格迹；
  - 更新每个状态s的价值函数；
  - 与TD-error和资格迹$E_t(s)$呈比例关系。

资格迹方法

- 示例1：随机游走
  - 每个回合都从中间状态C开始
  - 每一时刻都有均等的概率向左走或者向右走
  - 到最左侧或者最右侧时，回合结束
  - 如果走到最右侧，得到的奖励为1
  - 如果走到最左侧，得到的奖励为0

资格迹方法

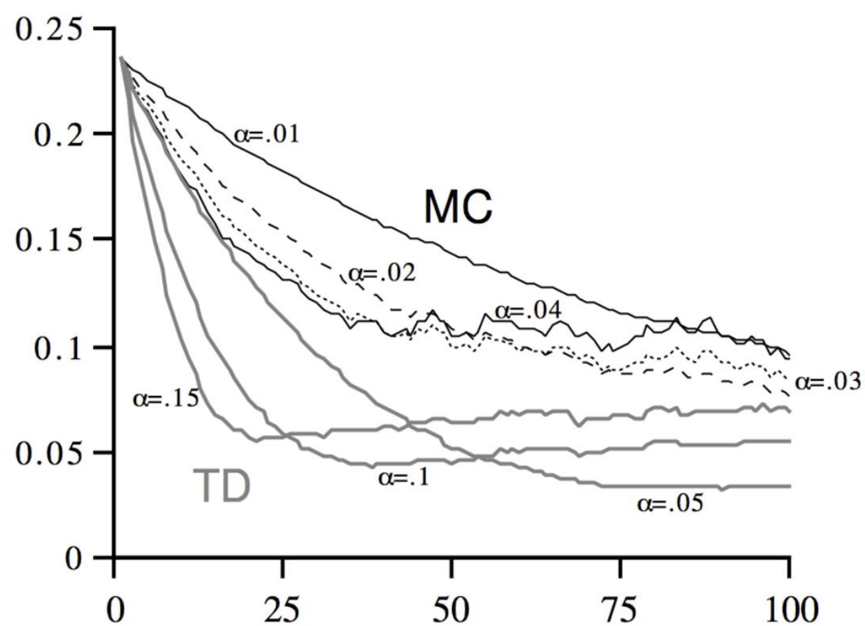
- 示例1：随机游走



资格迹方法

- 示例1：随机游走

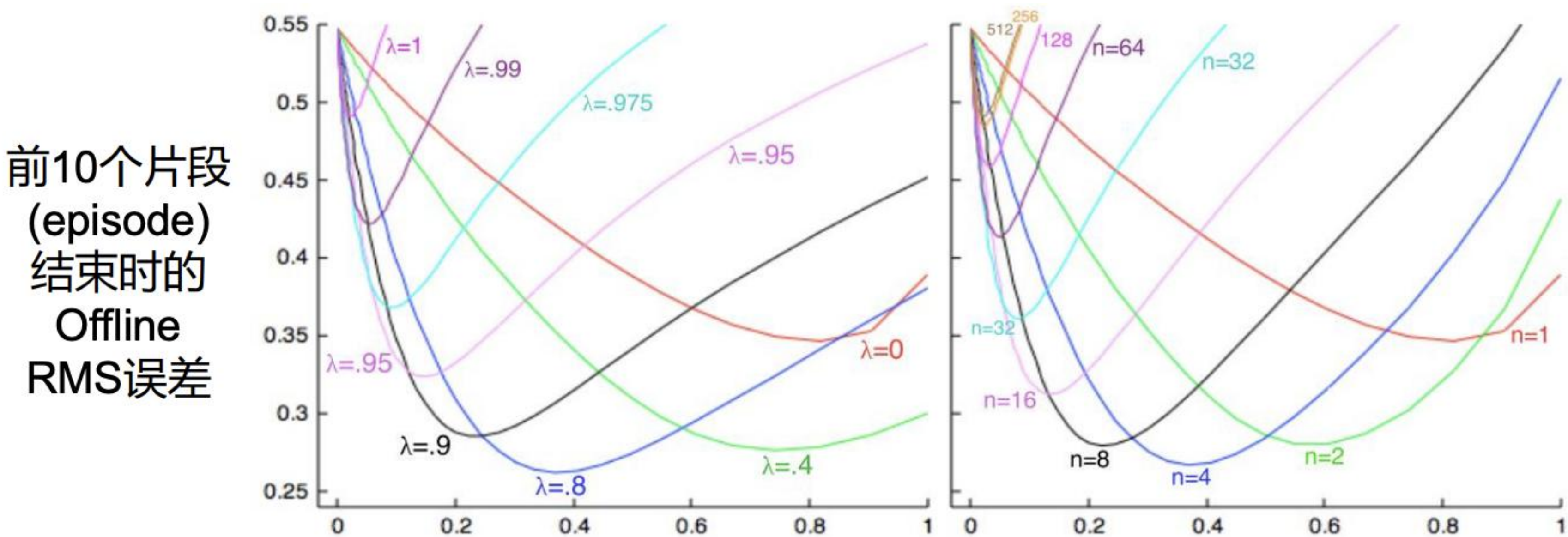
状态对应的
均方误差



回合数

资格迹方法

- 示例1：随机游走

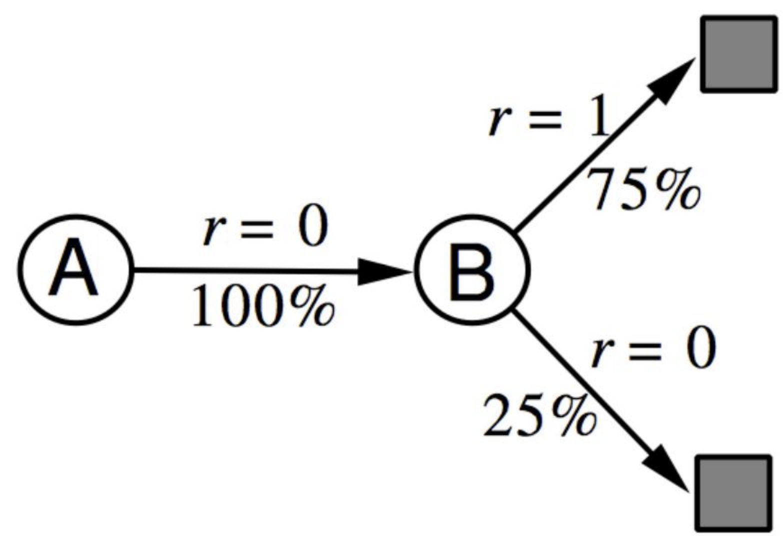


19个状态的随机游走实验结果

- TD-λ与n步时序差分有类似的效果

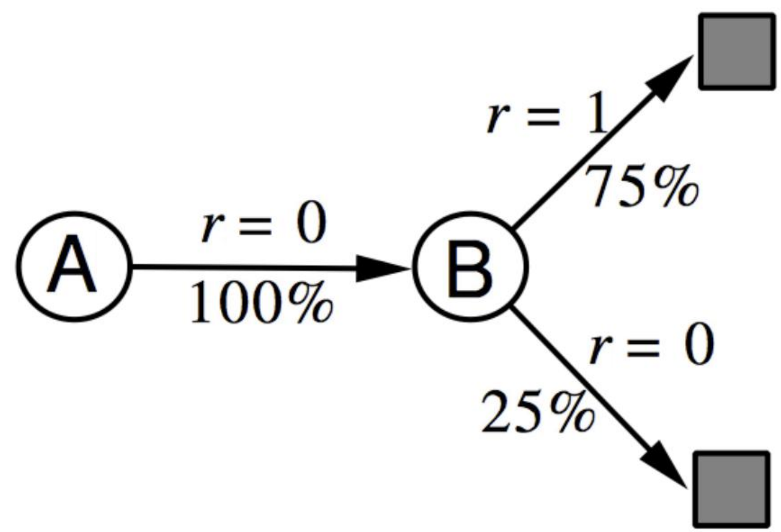
资格迹方法

- 示例2：AB环境
  - 无折扣因子
  - 一共8个回合的样本
    - A -> r = 0 -> B -> r = 0
    - B -> r = 1（6次重复）
    - B -> r = 0
  - 求状态A的价值V_A？



资格迹方法

- 示例2：AB环境
  - 根据蒙特卡洛方法，必须使用完整的回合进行计算，因此状态A的价值V_A=0
  - 单步的时序差分方法融合了状态B的价值V_B，因此状态A的价值V_A=0.75



课程大纲

- 蒙特卡洛方法
- 时序差分方法
- 资格迹方法
- 表格型时序差分方法：
  - Q-learning
  - SARSA

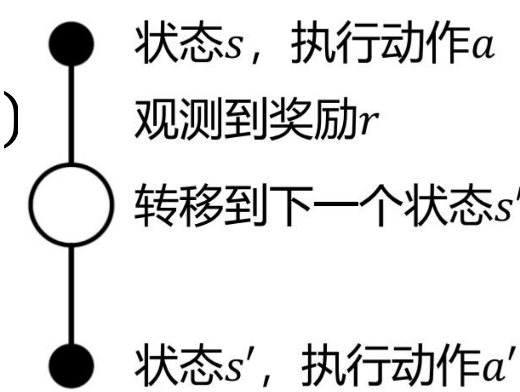
表格型时序差分方法

- 强化学习由策略评估和策略改进组成
- 策略评估的目标是知道什么决策是好的，即估计$V^{\pi}(s_t)$
- 策略提升的目标是根据值函数选择好的行动
  - 基于V函数，$\pi(s)=\arg\max_{a\in A}\sum_{s'\in S}P(s'|s,a)V(s')$

    需要知道环境模型

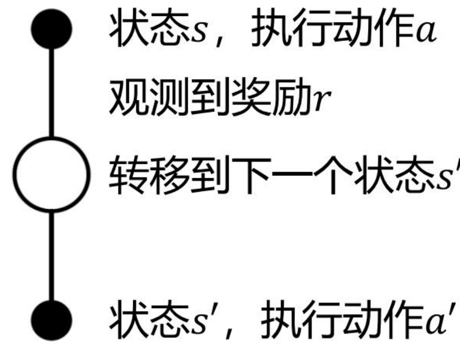
  - 基于Q函数，$\pi(s)=\arg\max_{a\in A}Q(s,a)$

    直接作用于决策



表格型时序差分方法：SARSA

- SARSA是一种针对表格环境中的时序差分方法
- 对策略执行的每个（状态-动作-奖励-状态-动作）元组

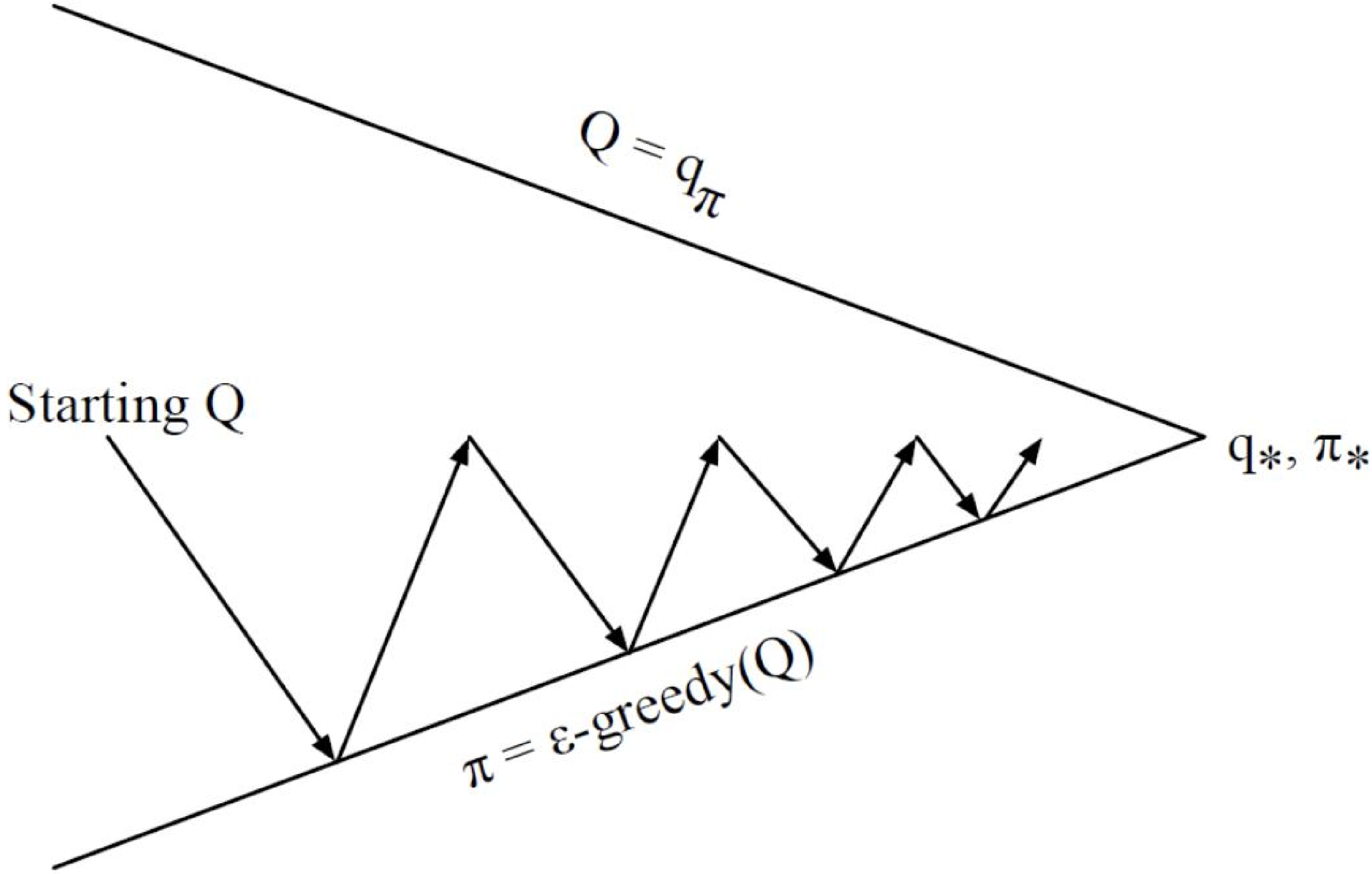


- SARSA的策略评估为更新状态-动作值函数

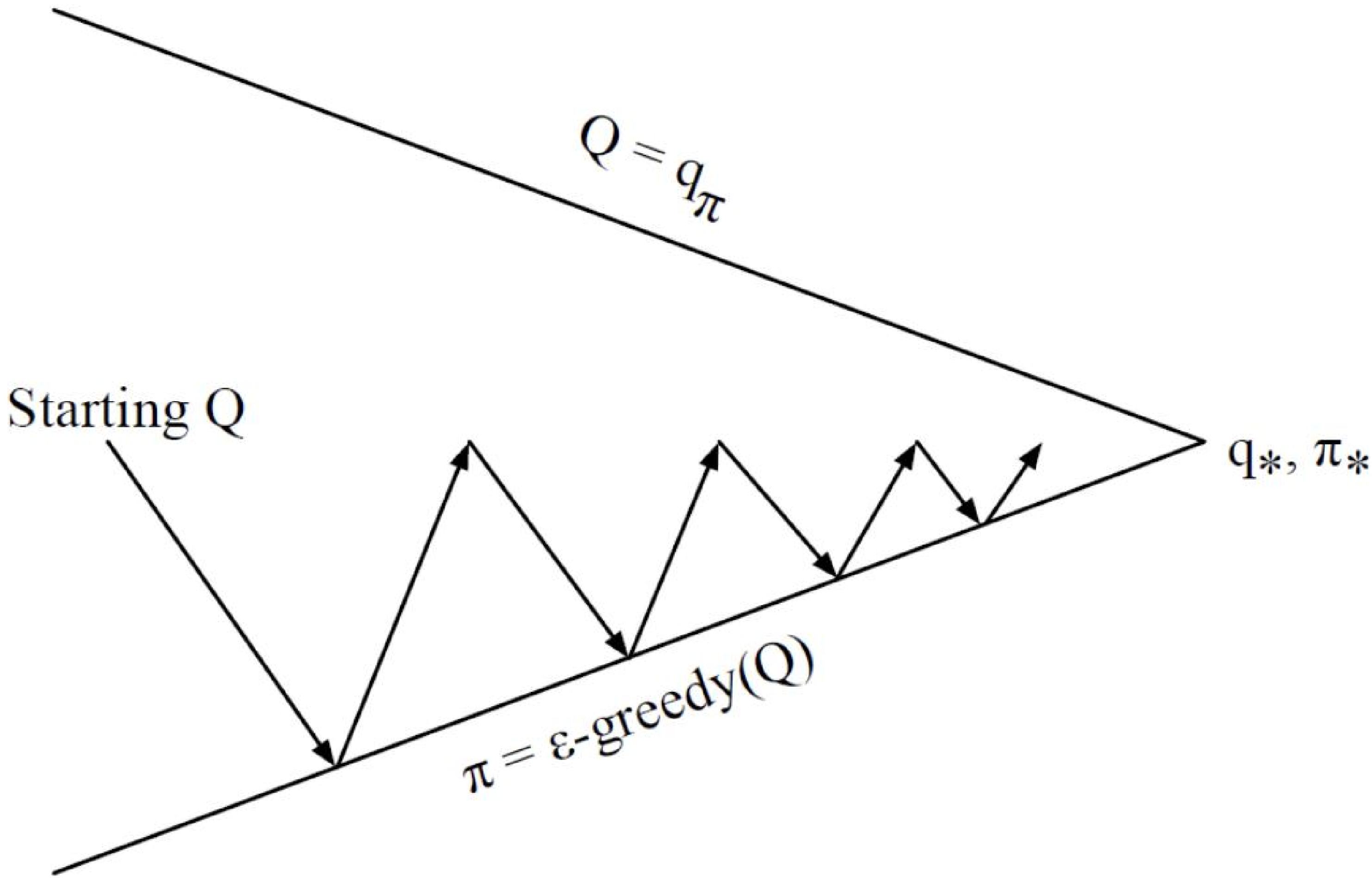
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\left(R_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right)$$

- SARSA的策略改进为$\epsilon - \text{greedy}$

表格型时序差分方法：SARSA



表格型时序差分方法：SARSA



- 迭代更新Q函数的估计值和策略

表格型时序差分方法：SARSA

Sarsa: An on-policy TD control algorithm

Initialize $Q(s,a), \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\textit{terminal-state}, \cdot) = 0$
Repeat (for each episode):
    Initialize S
    Choose A from S using policy derived from Q (e.g., ϵ-greedy)
    Repeat (for each step of episode):
        Take action A, observe R, S'
        Choose A' from S' using policy derived from Q (e.g., ϵ-greedy)
        $Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma Q(S',A') - Q(S,A)\big]$
        $S \leftarrow S'$; $A \leftarrow A'$;
    until S is terminal

表格型时序差分方法：SARSA

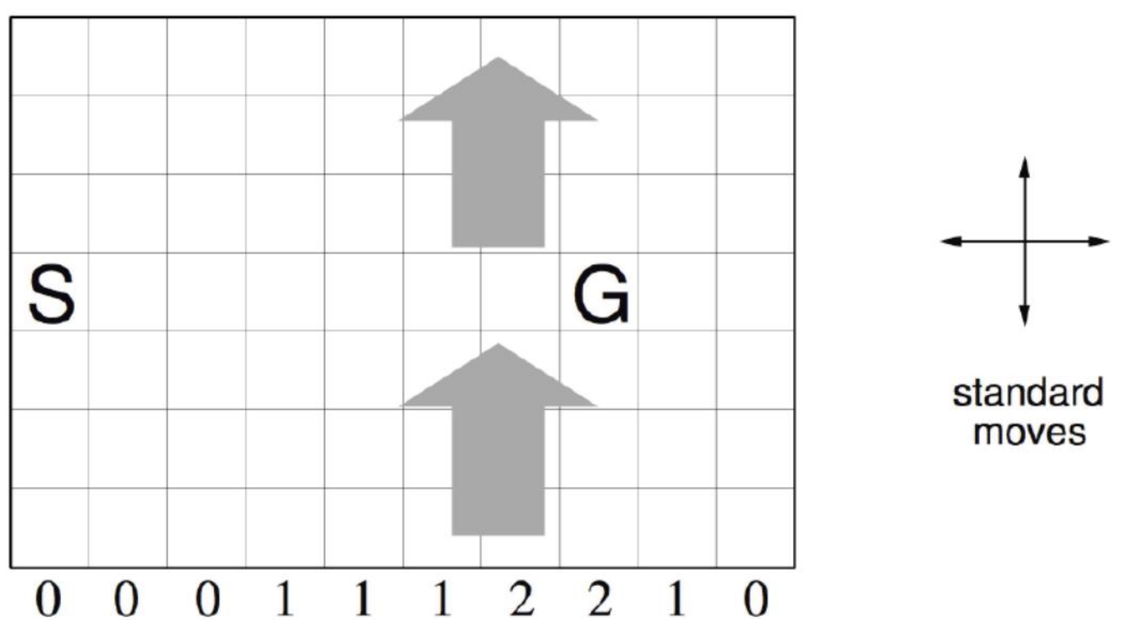
Sarsa: An on-policy TD control algorithm

Initialize $Q(s,a), \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\textit{terminal-state}, \cdot) = 0$
Repeat (for each episode):
  Initialize S
  Choose A from S using policy derived from Q (e.g., ϵ-greedy)
  Repeat (for each step of episode):
    Take action A, observe R, S'
    Choose A' from S' using policy derived from Q (e.g., ϵ-greedy)
    $Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma Q(S',A') - Q(S,A)\big]$
    $S \leftarrow S'$; $A \leftarrow A'$;
  until S is terminal

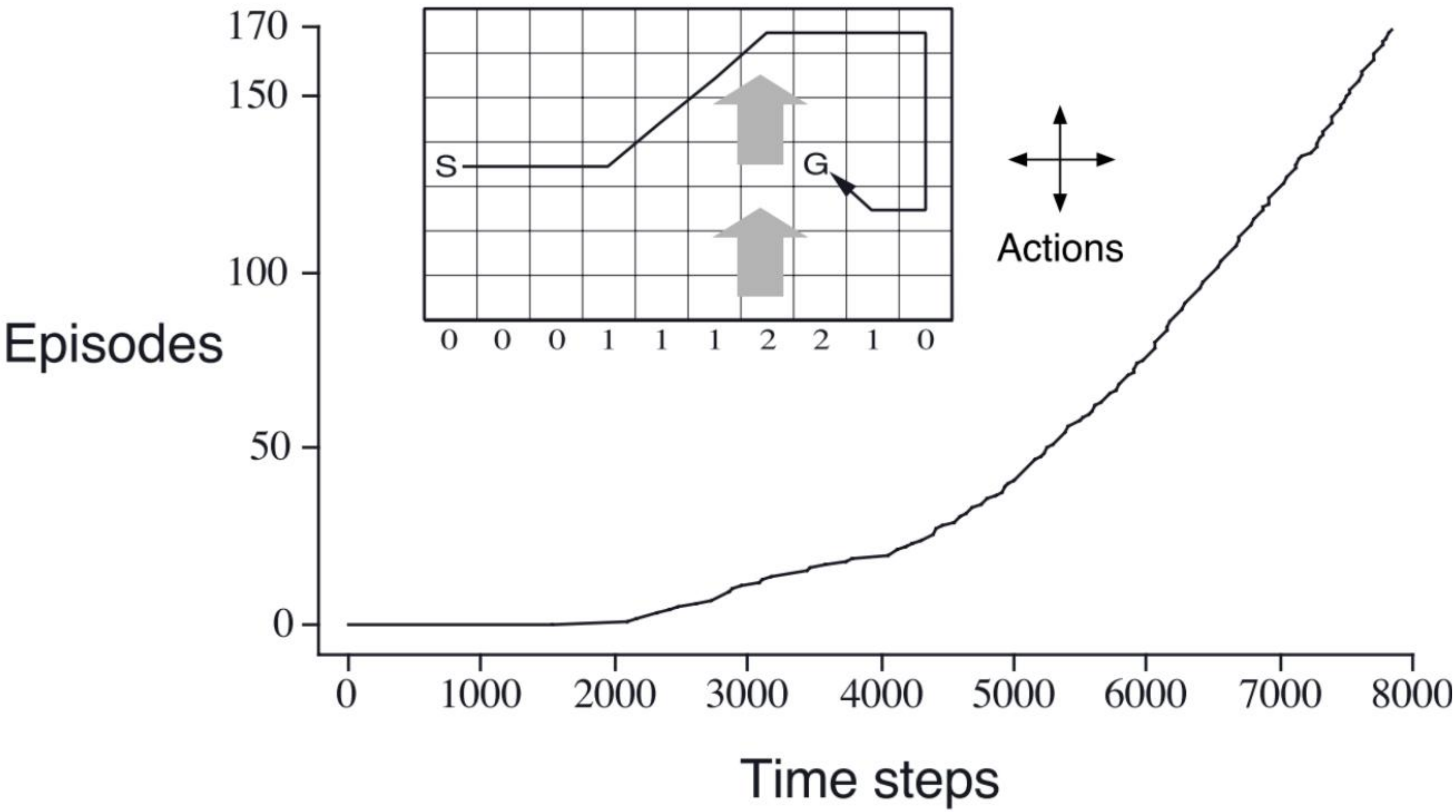
- 在线策略时序差分控制（on-policy TD control）使用当前策略进行动作采样，即SARSA算法中的两个动作“A”都是由当前策略选择的

表格型时序差分方法：SARSA

- SARSA示例：Windy Grid World
  - 每步的奖励为-1，直到智能体抵达目标网格
  - 折扣因子$\gamma = 1$



表格型时序差分方法：SARSA



- 随着训练的进行，SARSA策略越来越快速地抵达目标

表格型时序差分方法：Q-learning

- Q-learning学习状态动作值函数$Q(s, a) \in \mathbb{R}$，是一种离线策略（off-policy）方法

给定策略（可以不同于当前策略）

- $Q(s_t, a_t) = \sum_{t=0}^{T} \gamma^t R(s_t, a_t), a_t \sim \mu(s_t)$

奖励函数

- 迭代式：$Q(s_t, a_t) = R(s_t, a_t) + \gamma Q(s_{t+1}, a_{t+1})$

表格型时序差分方法：Q-learning

- 离线策略学习
  - 目标策略$\pi(a_t|s_t)$进行值函数评估（$V^{\pi}(s_t)$）或$Q^{\pi}(s_t,a_t)$）
  - 行为策略$\mu(a_t|s_t)$收集数据
- 为什么使用离线策略学习
  - 平衡探索（exploration）和利用（exploitation）
  - 通过观察人类或其他智能体学习策略
  - 重用旧策略所产生的经验
  - 遵循一个策略时学习多个策略

表格型时序差分方法：Q-learning

- 具体实现
- 使用行为策略$\mu(\bullet|s_t)$选择动作a_t
- 使用当前策略$\pi(\bullet|s_{t+1})$选择后续动作a'_{t+1}，计算目标

$$Q'(s_t, a_t) = R_t + \gamma Q(s_{t+1}, a'_{t+1})$$

- 使用时序差分更新$Q(s_t, a_t)$的值以拟合目标状态-动作值

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(R_{t+1} + \gamma Q'(s_{t+1}, a'_{t+1}) - Q(s_t, a_t))$$

动作来自当前策略π

表格型时序差分方法：Q-learning

- 目标策略π是关于$Q(s,a)$的贪心策略

$$\pi(s_{t+1}) = \arg\max_{a'} Q(s_{t+1}, a'_{t+1})$$

- 行为策略是关于的$Q(s,a)$的$\epsilon - greedy$策略
- Q-学习目标函数可以简化为

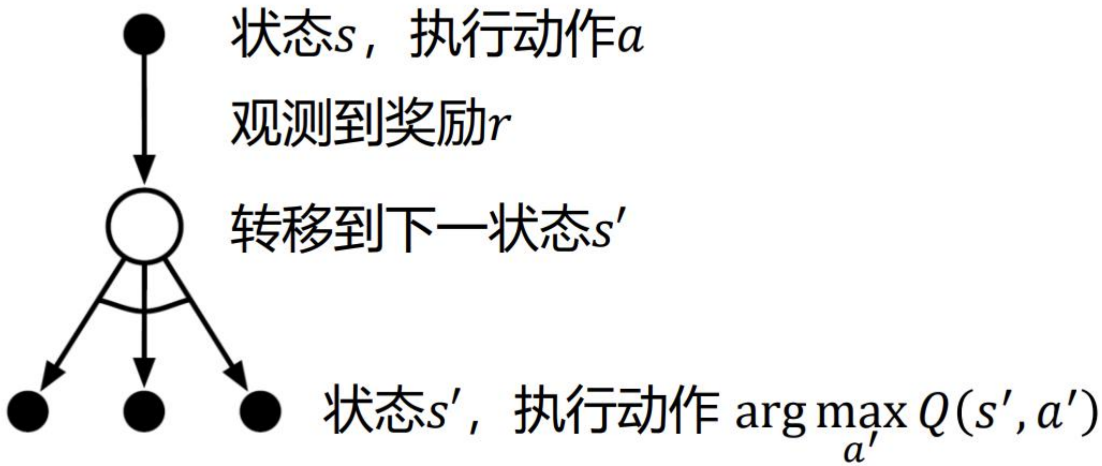
$$Q'(s_t, a_t) = R_t + \gamma \max_{a'} Q(s_{t+1}, a'_{t+1})$$

- Q-学习更新方式

$$Q(s_t, a_t)$$

$$\leftarrow Q(s_t, a_t) + \alpha(R_t + \gamma \max_{a'} Q(s_{t+1}, a'_{t+1}) - Q(s_t, a_t))$$

表格型时序差分方法：Q-learning



$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(R_t + \gamma \max_{a'} Q(s_{t+1}, a'_{t+1}) - Q(s_t, a_t))$$

- 定理：Q-learning能够收敛到最优状态-动作值函数

$$Q(s_t, a_t) \rightarrow Q^*(s_t, a_t)$$

表格型时序差分方法：Q-learning

- 回顾：收缩算子（contraction operator）
- 定义H算子

$$HQ(s_t, a_t) = R_t + \gamma \mathbb{E}_{s_{t+1} \sim p(\cdot|s,\, a)}[\max_{a'} Q(s_{t+1}, a_{t+1})]$$

- 最优值函数Q*是H的不动点，意味着：$Q^* = HQ^*$

表格型时序差分方法：Q-learning

- **直接从Q函数证明**

$$\|\mathbf{H}q_1 - \mathbf{H}q_2\|_\infty$$

$$(\mathbf{H}q)(x,a) = \sum_{y \in \mathcal{X}} \mathsf{P}_a(x,y)\left[r(x,a,y) + \gamma \max_{b \in \mathcal{A}} q(y,b)\right]$$

表格型时序差分方法：Q-learning

- **直接从Q函数证明**

$$\|\mathbf{H}q_1 - \mathbf{H}q_2\|_\infty \qquad (\mathbf{H}q)(x,a) = \sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left[r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q(y,b)\right]$$

$$= \max_{x,a}\left|\sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left[r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q_1(y,b) - r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q_2(y,b)\right]\right|$$

表格型时序差分方法：Q-learning

- **直接从Q函数证明**

$$(\mathbf{H}q)(x,a) = \sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left[r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q(y,b)\right]$$

$$\|\mathbf{H}q_1 - \mathbf{H}q_2\|_\infty$$

$$= \max_{x,a}\left|\sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left[r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q_1(y,b) - r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q_2(y,b)\right]\right|$$

$$= \max_{x,a}\gamma\left|\sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left[\max_{b\in\mathcal{A}} q_1(y,b) - \max_{b\in\mathcal{A}} q_2(y,b)\right]\right|$$

表格型时序差分方法：Q-learning

- **直接从Q函数证明**

$$(\mathbf{H}q)(x,a) = \sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left[r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q(y,b)\right]$$

$$\begin{aligned}
&\|\mathbf{H}q_1 - \mathbf{H}q_2\|_\infty \\
&= \max_{x,a}\left|\sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left[r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q_1(y,b) - r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q_2(y,b)\right]\right| \\
&= \max_{x,a}\gamma\left|\sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left[\max_{b\in\mathcal{A}} q_1(y,b) - \max_{b\in\mathcal{A}} q_2(y,b)\right]\right| \\
&\le \max_{x,a}\gamma\sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left|\max_{b\in\mathcal{A}} q_1(y,b) - \max_{b\in\mathcal{A}} q_2(y,b)\right|
\end{aligned}$$

表格型时序差分方法：Q-learning

- **直接从Q函数证明**

$$(\mathbf{H}q)(x,a) = \sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left[r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q(y,b)\right]$$

$$\begin{aligned}
&\|\mathbf{H}q_1 - \mathbf{H}q_2\|_\infty \\
&= \max_{x,a}\left|\sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left[r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q_1(y,b) - r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q_2(y,b)\right]\right| \\
&= \max_{x,a}\gamma\left|\sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left[\max_{b\in\mathcal{A}} q_1(y,b) - \max_{b\in\mathcal{A}} q_2(y,b)\right]\right| \\
&\le \max_{x,a}\gamma \sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left|\max_{b\in\mathcal{A}} q_1(y,b) - \max_{b\in\mathcal{A}} q_2(y,b)\right| \\
&\le \max_{x,a}\gamma \sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\max_{z,b}\left|q_1(z,b) - q_2(z,b)\right|
\end{aligned}$$

表格型时序差分方法：Q-learning

- **直接从Q函数证明**

$$(\mathbf{H}q)(x,a) = \sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left[r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q(y,b)\right]$$

$$\begin{aligned}
&\|\mathbf{H}q_1 - \mathbf{H}q_2\|_\infty \\
&= \max_{x,a}\left|\sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left[r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q_1(y,b) - r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q_2(y,b)\right]\right| \\
&= \max_{x,a}\gamma\left|\sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left[\max_{b\in\mathcal{A}} q_1(y,b) - \max_{b\in\mathcal{A}} q_2(y,b)\right]\right| \\
&\leq \max_{x,a}\gamma \sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left|\max_{b\in\mathcal{A}} q_1(y,b) - \max_{b\in\mathcal{A}} q_2(y,b)\right| \\
&\leq \max_{x,a}\gamma \sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\max_{z,b}\left|q_1(z,b) - q_2(z,b)\right| \\
&= \max_{x,a}\gamma \sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left\|q_1 - q_2\right\|_\infty \\
&= \gamma\left\|q_1 - q_2\right\|_\infty .
\end{aligned}$$

表格型时序差分方法：Q-learning

- **直接从Q函数证明**

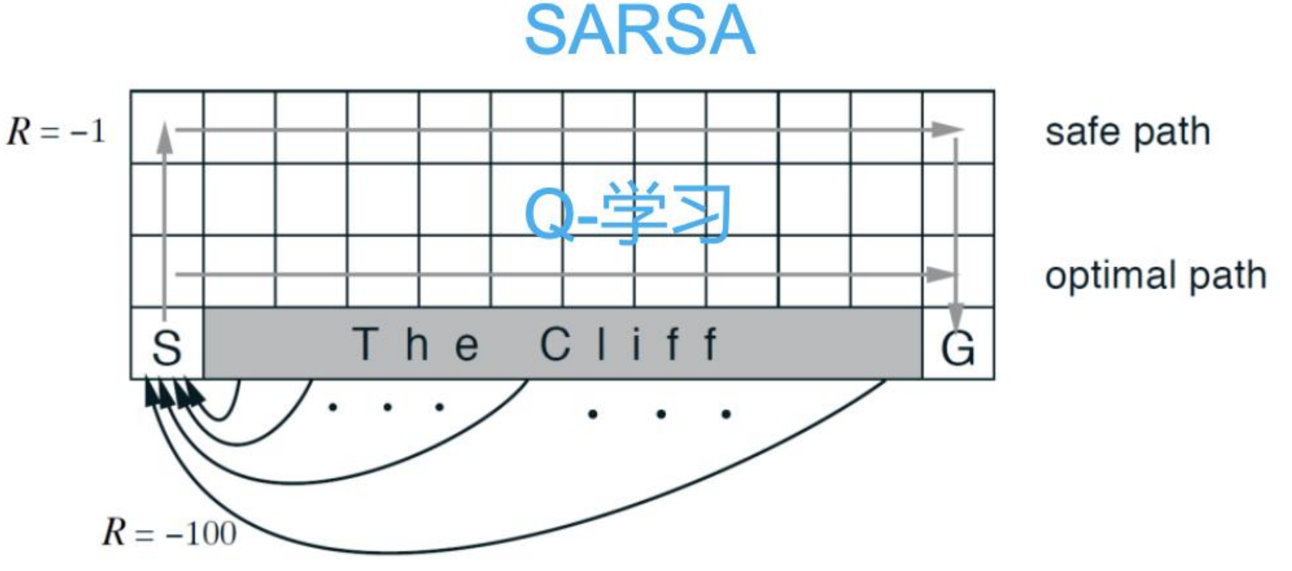
$$(\mathbf{H}q)(x,a) = \sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left[r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q(y,b)\right]$$

$$\begin{aligned}
&\|\mathbf{H}q_1 - \mathbf{H}q_2\|_\infty \\
&= \max_{x,a} \left| \sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left[r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q_1(y,b) - r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q_2(y,b)\right] \right| \\
&= \max_{x,a} \gamma \left| \sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y)\left[\max_{b\in\mathcal{A}} q_1(y,b) - \max_{b\in\mathcal{A}} q_2(y,b)\right] \right| \\
&\leq \max_{x,a} \gamma \sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y) \left|\max_{b\in\mathcal{A}} q_1(y,b) - \max_{b\in\mathcal{A}} q_2(y,b)\right| \\
&\leq \max_{x,a} \gamma \sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y) \max_{z,b} |q_1(z,b) - q_2(z,b)| \\
&= \max_{x,a} \gamma \sum_{y\in\mathcal{X}} \mathsf{P}_a(x,y) \|q_1 - q_2\|_\infty \\
&= \gamma \|q_1 - q_2\|_\infty .
\end{aligned}$$

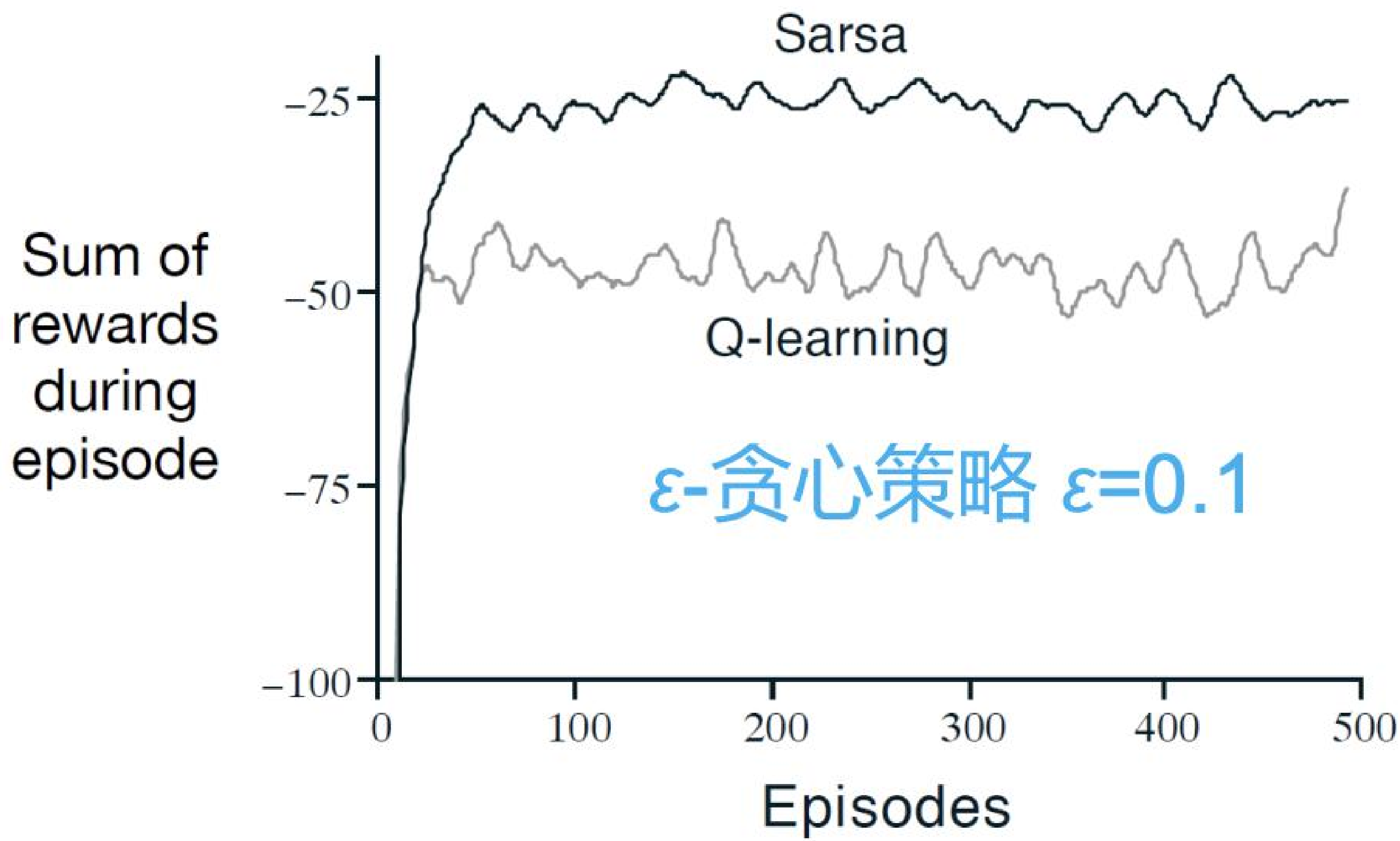
因此 $\|\mathbf{H}q_1 - \mathbf{H}q_2\|_\infty \leq \gamma \|q_1 - q_2\|_\infty$

表格型时序差分方法：Q-learning

- Q-learning示例：悬崖漫步（Cliff-walking）
  - 折扣因子$\gamma = 1$
  - 每一步移动奖励为-1
  - 踏入悬崖会产生-100奖励，并开启下一轮回合



表格型时序差分方法：Q-learning



提问环节

下午实验课

需要完成：

- 运行 gymnasium 中的 Cliff Walking 环境的
- 基于 Cliff Walking 实现表格型 Q-learning 和 SARSA 算法
- 测试 Q-learning 和 SARSA 算法的表现与区别