强化学习基本原理与编程实现04：蒙特卡洛与时间差分

郭宪

2019.10.20

人工智能学院
College of Artificial Intelligence

Nankai University

强化学习方法分类

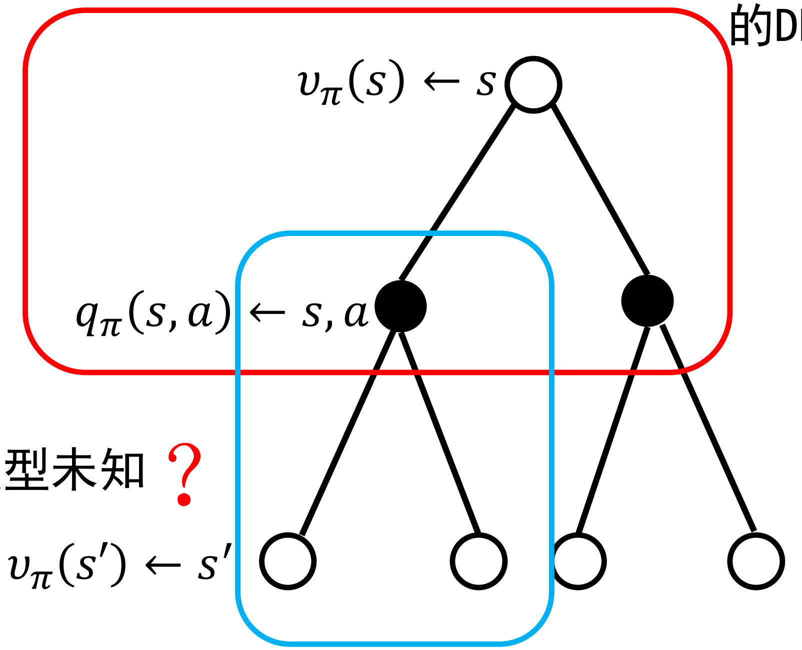
- 序贯决策问题
  - 马尔科夫决策过程 MDP (S,A,P,R,γ)
    - (S,A,P,R,γ) 基于模型的动态规划方法
      - 策略迭代
      - 值迭代
      - 策略搜索
    - $(S,A,P?\ R?\ \gamma?\ )$ 无模型的强化学习方法
      - 蒙特卡洛方法
      - 时间差分方法（TD方法）

本节讲蒙特卡罗方法

模型未知(model-free)

给定策略π 构造值函数：

基于模型已知的DP方法



$$v_\pi(s) = \sum_{a\in A} \pi(a|s)\left(R_s^a + \gamma \sum_{s'\in S} P_{ss'}^a v_\pi(s')\right)$$

当智能体采用策略π时，累积回报服从一个概率分布，累积回报在状态s处的期望值定义为值函数：

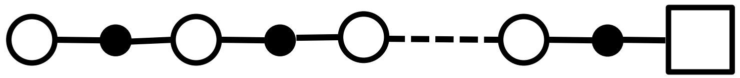
$$v_\pi(s) = E_\pi[G_t|S_t = s] = E_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}|S_t = s\right]$$

状态-行为值函数：

$$q_\pi(s,a) = E_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}|S_t = s, A_t = a\right]$$

蒙特卡洛方法利用经验平均代替随机变量的期望：

一次实验（an episode）： $S_1, A_1, R_2, \cdots, S_k \sim \pi$

终止状态： 

计算状态s后的折扣回报返回值：

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

蒙特卡罗策略评估

动态规划策略评估算法

输入：需要评估的策略 π 状态转移概率 $P_{ss'}^{a}$ 回报函数 R_s^a ，折扣因子 γ

初始化值函数：$V(s)=0$

一次状态扫描

Repeat k=0,1,…

for every s do

$$v_{k+1}(s)=\sum_{a\in A}\pi(a|s)\left(R_s^a+\gamma\sum_{s'\in S}P_{ss'}^{a}v_k(s')\right)$$

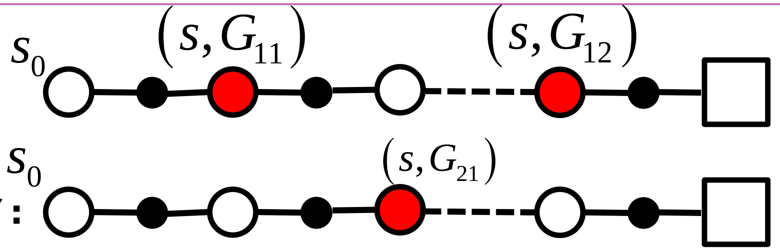
end for

Until $v_{k+1}=v_k$

输出： $v(s)$

一次试验：

第二次试验：



⋮

蒙特卡罗方法利用经验平均代替随机变量的期望。

First visit MC策略评估： $\upsilon(s)=\dfrac{G_{11}(s)+G_{21}(s)+\cdots}{N(s)}$

every visit MC策略评估：

$$\upsilon(s)=\frac{G_{11}(s)+G_{12}(s)+\cdots G_{21}(s)+\cdots}{N(s)}$$

根据大数定律：$\upsilon(s)\to\upsilon_{\pi}(s)\ as\ N(s)\to\infty$

Remark: 当只对一个点的值感兴趣，或者只对特定区域感兴趣时，蒙特卡洛方法效率很高

蒙特卡罗策略评估

动态规划策略评估算法

输入：需要评估的策略 π 状态转移概率 $P_{ss'}^{a}$ 回报函数 R_s^a ，折扣因子 γ

初始化值函数：$V(s) = 0$

一次状态扫描

Repeat k=0,1,…

for every s do

$$v_{k+1}(s) = \sum_{a\in A} \pi(a|s)\left(R_s^a + \gamma \sum_{s'\in S} P_{ss'}^{a} v_k(s')\right)$$

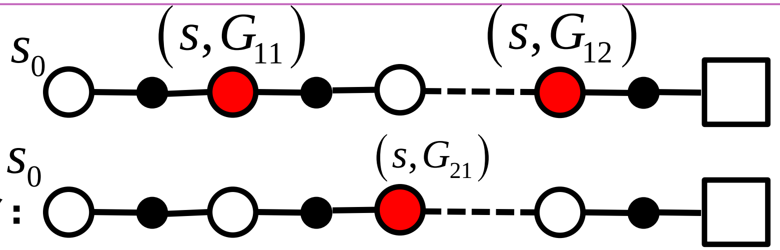
end for

Until $v_{k+1} = v_k$

输出：$v(s)$

一次试验：

第二次试验：



⋮

First visit MC and every visit MC

根据大数定律：$\upsilon(s) \to \upsilon_\pi(s)$ as $N(s) \to \infty$

探索型初始化状态：

每个状态都有一定的几率作为初始状态

	1	2	3
4	5	6	7
8	9	10	11
12	13	14	

蒙特卡洛评估—行为值函数

1. 对于模型未知的情况，最重要的是评估行为值函数

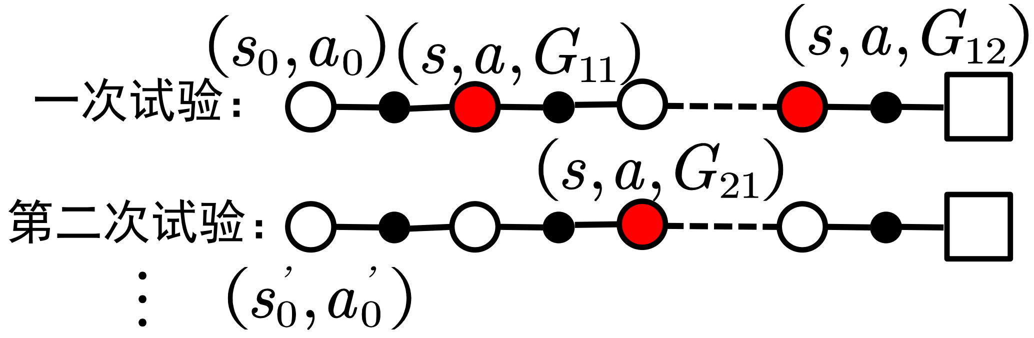
   因为对于模型已知的情况，状态值函数足矣

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_\pi(s')$$

2. 没有模型时，只有状态值就不够了，必须直接显式地评估：$q_\pi(s,a)$

行为值函数的定义：$q_\pi(s,a) = E_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a\right]$

蒙特卡洛策略评估：



$$\hat{q}(s,a) = \frac{G_{11}(s,a) + G_{12}(s,a) + \cdots + G_{21}(s,a) + \cdots}{N(s,a)}$$

根据大数定律：

$$\hat{q}(s,a) \to q_\pi(s,a) \;\; as \;\; N(s,a) \to \infty$$

蒙特卡罗策略改进

蒙特卡洛策略改进：

对于每个状态s，通过最大化动作值函数，来进行策略的改进。

$$\pi(s) = \arg\max_{a} q(s,a)$$

我们需要评估当前状态下的每个动作的值函数

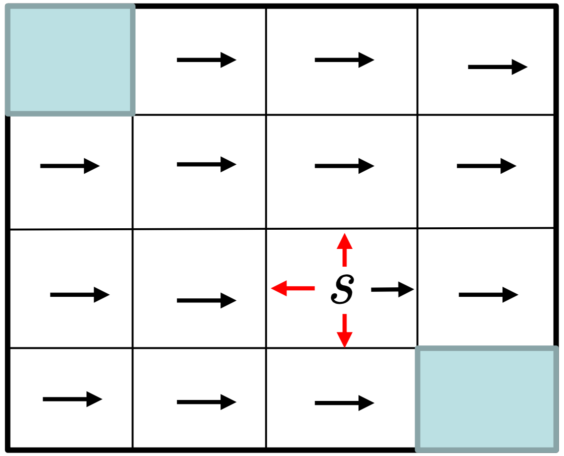
$$q_\pi(s,a) \quad for \ a \in A$$

需要访问所有的状态-行为对 (s,a)

行为值函数的定义：

$$q_\pi(s,a) = E_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,|S_t = s, A_t = a\right]$$

当前的策略： $\pi(\cdot|\cdot) = \text{‘}e\text{’}$



$q(s,a_1)$ $q(s,a_2)$ $q(s,a_3)$ $q(s,a_4)$

蒙特卡罗强化学习：探索初始化

[1] 初始化所有：

$s \in S, a \in A(s)$，$Q(s,a) \leftarrow arbitrary,$
$\pi(s) \leftarrow arbitrary, \mathrm{Re}\,t\,urns(s,a) \leftarrow emptylist$

[2] Repeat:

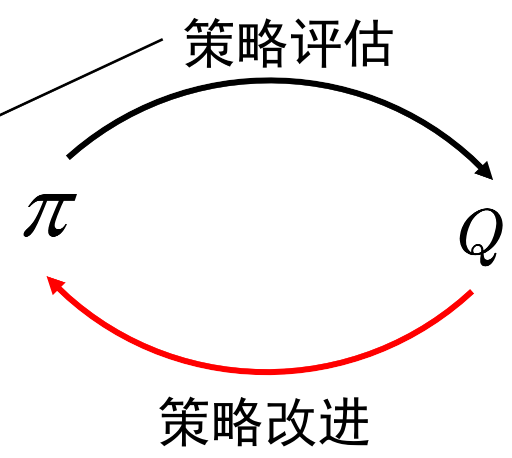
随机选择 $S_0 \in S, A_0 \in A(S_0)$ ，从 S_0, A_0 开始以策略 π

生成一个实验(episode)，对每个在这个实验中出现的状态和动作对(s,a):

策略评估

[3]

$G \leftarrow s, a$第一次出现后的回报
将G附加于回报Returns(s, a)上
$\mathrm{Q}(s,a) \leftarrow average(\mathrm{Re}\,t\,urns(s,a))$对回报取均值



[4] 对该实验中的每一个s:
$\pi(s) \leftarrow \arg\ \max_a Q\,(s,a)$

策略改进

蒙特卡罗强化学习：无探索初始化

[1] 初始化所有：

$$s \in S, a \in A(s), Q(s,a) \leftarrow arbitrary,$$

$$\pi(s) \leftarrow arbitrary, \mathrm{Re}turns(s,a) \leftarrow empty\,list$$

[2] Repeat:

随机选择 $S_0 \in S, A_0 \in A(S_0)$ ，从 S_0, A_0 开始以策略 π

生成一个实验(episode)，对每对在这个实验中出现的状态和动作，s，a：

策略评估

[3] $G \leftarrow s,a$第一次出现后的回报

将G附加于回报Returns(s, a)上

$Q(s,a) \leftarrow average(\mathrm{Re}turns(s,a))$对回报取均值

[4] 对该实验中的每一个s：

$\pi(s) \leftarrow \arg\max_a Q(s,a)$ 策略改进

探索性初始化：迭代中每一幕的初始状态随机分配，以保证迭代过程中每个状态行为对都能被选中。假设所有的动作都被无限频繁选中。

$\Downarrow$ 初始状态不变？

设计探索策略，保证每个状态都被访问到即可

$\Updownarrow$

策略必须是永久温和的，即对所有的s和a满足：$\pi(a|s) > 0$

On-policy

Off-policy

On-policy MC

策略必须是永久温和的，即对所有的s和a满足：

$\pi(a|s)>0$

典型的策略为：ε-soft 策略，即

$$\pi(a|s)\leftarrow\begin{cases}1-\varepsilon+\dfrac{\varepsilon}{|A(s)|} & if\ a=\arg\max_a Q(s,a)\\ \dfrac{\varepsilon}{|A(s)|} & if\ a\neq\arg\max_a Q(s,a)\end{cases}$$

所谓on-policy是指产生数据的策略与所评估和改善的策略是一个策略。

此处产生数据的策略和评估及改善的策略都是 ε-soft 策略

[1] 初始化所有：$s\in S, a\in A(s), Q(s,a)\leftarrow arbitrary$

$Returns(s,a)\leftarrow empty\ list$

$\pi(s)\leftarrow arbitrary\ \varepsilon$-soft策略,

Repeat:

[2] 从 S_0, A_0 开始以策略 π 生成一次实验(episode)，

[3] 对每对在这个实验中出现的状态和动作，s，a：

$G\leftarrow s,a$第一次出现后的回报

将G附加于回报Returns(s, a)上

$Q(s,a)\leftarrow average(Returns(s,a))$对回报取均值

策略评估

[4] 对该实验中的每一个s：

策略改进

$$\pi(a|s)\leftarrow\begin{cases}1-\varepsilon+\dfrac{\varepsilon}{|A(s)|} & if\ a=\arg\max_a Q(s,a)\\ \dfrac{\varepsilon}{|A(s)|} & if\ a\neq\arg\max_a Q(s,a)\end{cases}$$

off-policy MC

学习算法的困境：需要以最优的动作采样，但同时又需要探索所有的动作。

On-policy: 在探索和利用之间进行了妥协

对该实验中的每一个s:

$$\pi(a|s) \leftarrow \begin{cases} 1-\varepsilon+\dfrac{\varepsilon}{|A(s)|} & if\ \ a=\arg\max_a Q(s,a) \\ \dfrac{\varepsilon}{|A(s)|} & if\ \ a\neq \arg\max_a Q(s,a) \end{cases}$$

off-policy: 更直接的方法是利用两个策略，

一个是不断改进的策略（称为目标策略，target policy），

另外一个负责探索（称为动作策略，behavior policy），产生更多的动作

off-policy MC

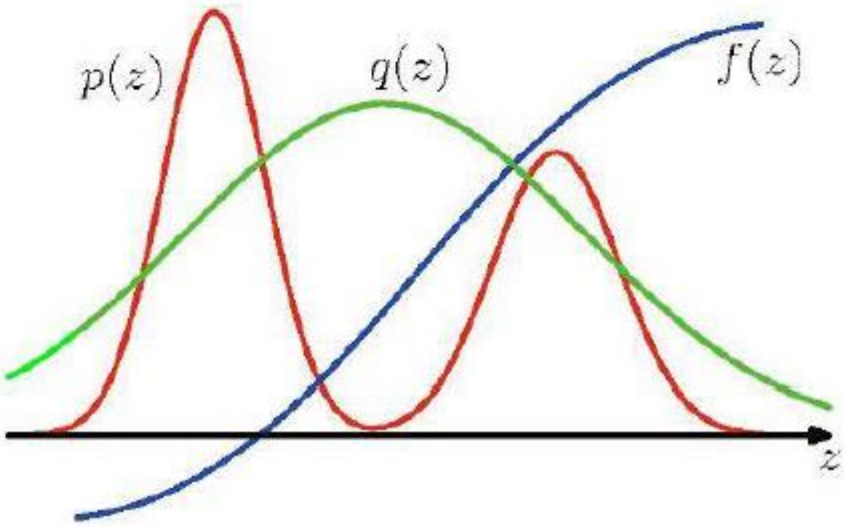
●为了保证探索性，学习和评估的目标策略 π 与用来产生样本的行为策略 μ 不同。根据行为策略 μ 中获得的经验更新目标策略 π 的值。

●例如：π 是贪婪策略（最终的最优策略）， μ 是探索策略（如ε-$soft$ 策略）

●Off-policy 的行为策略和目标策略需要满足覆盖性条件：即 μ 产生的行为覆盖或包含 π 可能产生的行为，用式子表示即为：满足$\pi(a|s)>0$ 的任何（s,a)均满足$\mu\left(a|s\right)>0$

●重要性采样(importance sampling):给每个回报赋以一定的权重，该权重为两个策略下轨迹可能性的比值

重要性采样



重要性采样

重要性采样求积分：

$$E[f]=\int f(z)p(z)dz$$

$$=\int f(z)\frac{p(z)}{q(z)}q(z)dz$$

$$\approx \frac{1}{N}\sum_n \frac{p(z^n)}{q(z^n)}f(z^n),\ z^n \sim q(z)$$

定义重要性权重：$\omega^n = p(z^n)/q(z^n)$

普通的重要性采样求积分：$E[f]=\frac{1}{N}\sum_n \omega^n f(z^n)$

重要性采样积分：无偏估计，但方差无穷大

减小方差的方法：加权重要性采样求积分

$$E[f]\approx\sum_{n=1}^{N}\frac{\omega^n}{\sum_{m=1}^{N}\omega^m}f(z^n)$$

MC 重要性采样

在策略 π 下，t 时刻后轨迹的概率为：

$$\Pr\left(A_t, S_{t+1}, \cdots, S_T\right) = \prod_{k=t}^{T-1} \pi\left(A_k \mid S_k\right) p\left(S_{k+1} \mid S_k, A_k\right)$$

在目标策略和行为策略下，每个回报都使用概率进行加权

$$\rho_t^T = \frac{\prod_{k=t}^{T-1} \pi\left(A_k \mid S_k\right) p\left(S_{k+1} \mid S_k, A_k\right)}{\prod_{k=t}^{T-1} \mu\left(A_k \mid S_k\right) p\left(S_{k+1} \mid S_k, A_k\right)} = \prod_{k=t}^{T-1} \frac{\pi\left(A_k \mid S_k\right)}{\mu\left(A_k \mid S_k\right)}$$

普通重要性采样，值估计：

从t到T(t)的返回值

时间t后的第一次终止时刻

$$V(s) = \frac{\sum_{t \in \mathcal{T}(s)} \rho_t^{T(t)} G_t}{\left|\mathcal{T}(s)\right|}$$

状态s被访问过的所有时刻的集合

s　　　　　s

t = 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

$$\mathcal{T}(s) = \{4,15\} \quad T(4) = 7, T(15) = 19$$

加权重要性采样，值估计：

$$V(s) = \frac{\sum_{t \in \mathcal{T}(s)} \rho_t^{T(t)} G_t}{\sum_{t \in \mathcal{T}(s)} \rho_t^{T(t)}}$$

Off-policy every visit MC

初始化，对于所有的

$s \in S, a \in A(s)$:

$Q(s,a) \leftarrow$ 任意

$C(s,a) \leftarrow 0$

$\pi(s) \leftarrow$ 相对于Q的贪婪策略

Repeat forever:

利用软策略μ产生一次实验：

$S_0, A_0, R_1, \cdots, S_{T-1}, A_{T-1}, R_T, S_T$

$G \leftarrow 0$

$W \leftarrow 1$

$For\ t = T-1, T-2, \cdots down to\ 0:$

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

$C(S_t, A_t) \leftarrow C(S_t, A_t) + W$

策略评估

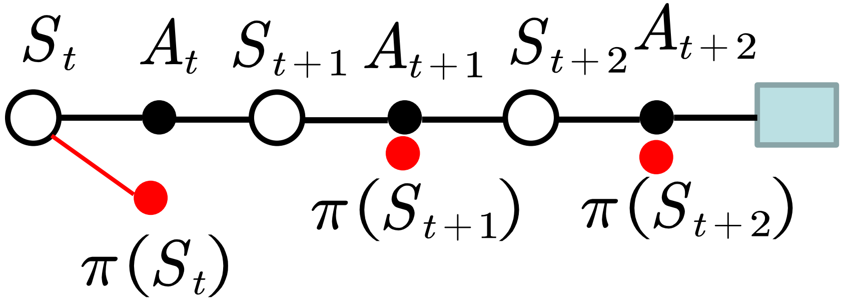
$\boxed{Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)}\left[G - Q(S_t, A_t)\right]}$

$\boxed{\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)}$

策略改善

如果$A_t \neq \pi(S_t)$则退出for循环

$W \leftarrow W \frac{1}{\mu(A_t | S_t)}$



强化学习方法分类

序贯决策问题

⇓

马尔科夫决策过程 MDP (S, A, P, R, γ)

- (S, A, P, R, γ) → 基于模型的动态规划方法
  - 策略迭代
  - 值迭代
  - 策略搜索
- $(S, A, P?\ R?\ \gamma?\ )$ → 无模型的强化学习方法
  - 蒙特卡罗方法
  - 时间差分方法（TD方法）

本节讲时间差分方法

DP, MC and TD

DP方法估计值函数：

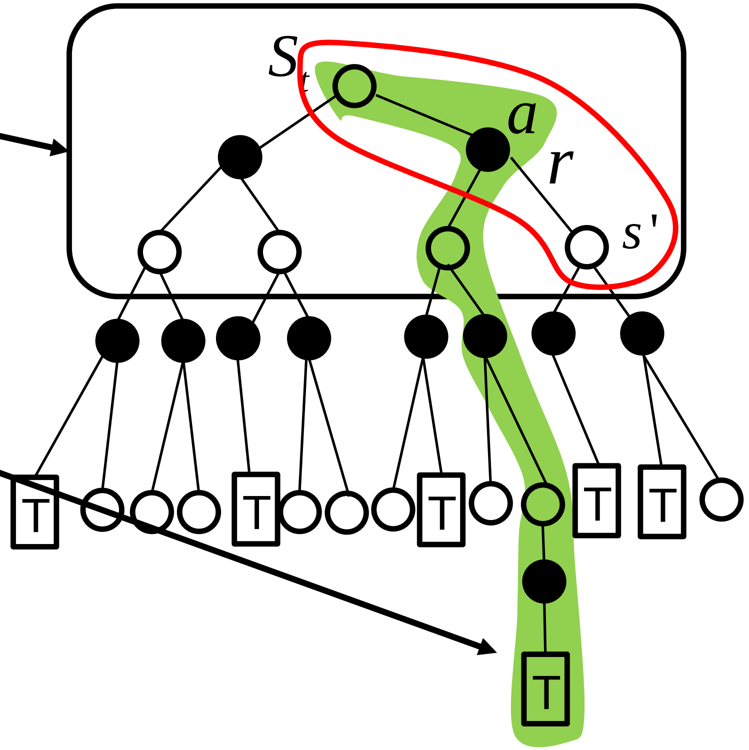
$$v_{l+1}(s)=\max_a R_s^a+\gamma\sum_{s\prime\in S}P_{ss\prime}^a v_l(s')$$

MC方法估计值函数：

$$v(s_t)=\mathbb{E}_\pi\left(r_{t+1}+\gamma r_{t+2}+\gamma^2 r_{t+3}+\cdots\right)$$

时间差分的方法评估值函数：

$$v(s_t)=\mathbb{E}_\pi\left(r_{t+1}+\gamma v\left(s_{t+1}\right)\right)$$



DP, MC and TD

MC：利用采样平均回报逼近期望

$$\begin{aligned}
\upsilon_\pi(s) &= E_\pi\left[G_t \mid S_t = s\right] \\
&= E_\pi\left[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1} \mid S_t = s\right] \\
&= E_\pi\left[R_{t+1} + \gamma\sum_{k=0}^{\infty}\gamma^k R_{t+k+2} \mid S_t = s\right] \\
&= E_\pi\left[R_{t+1} + \gamma\upsilon_\pi\left(S_{t+1}\right) \mid S_t = s\right]
\end{aligned}$$

TD：联合了MC和DP，采样期望值，并利用真值的当前估计值 $V(S_{t+1})$

DP:期望值由模型来提供，但是利用真值的当前估计值 $V(S_{t+1})$

MC and TD 偏差与方差平衡

增量式MC方法估计值函数：

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

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

⇒ G_t 是值函数 $v_\pi(S_t)$ 的无偏估计

最简单的时间差分学习算法：TD(0)

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

⇒ 真实的TD目标 $R_{t+1} + \gamma v_\pi(S_{t+1})$ 是无偏估计，但 $R_{t+1} + \gamma V(S_{t+1})$ 是有偏估计

$R_{t+1} + \gamma V(S_{t+1})$ 称为TD目标

TD目标 $R_{t+1} + \gamma v_\pi(S_{t+1})$ 的方差比MC的返回值 G_t 要小很多。因为MC的返回值依赖于很多随机动作，转移概率和回报。TD目标仅依赖于一个随机动作，转移概率和回报。

时间差分学习

TD学习是采样更新: 一个样本

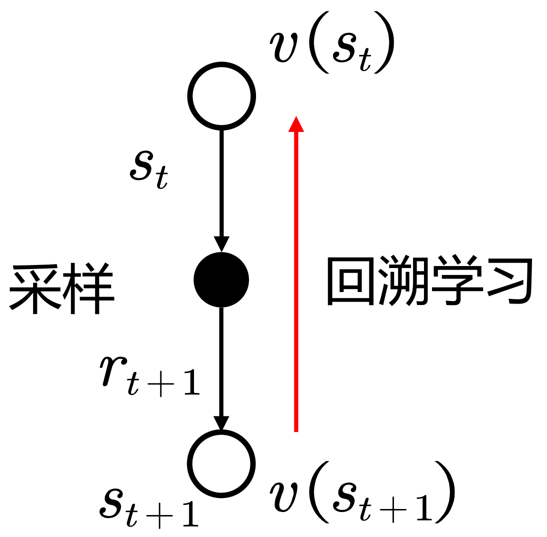
$$V(S_t) \leftarrow V(S_t) + \alpha\left(\boxed{R_{t+1} + \gamma V(S_{t+1})} - V(S_t)\right)$$

DP学习是期望更新: 先计算期望

$$v_\pi(s) = \sum_{a\in A} \pi(a|s)\left(R_s^a + \gamma \sum_{s'\in S} P_{ss'}^a v_\pi(s')\right)$$

TD偏差，新的估计值与旧的估计值的差:

$$\delta_t = R_{t+1} + \gamma v(s_{t+1}) - v(s_t)$$



时间差分学习的优势

TD学习是采样更新：

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

DP学习是期望更新：

$$v_\pi(s) = \sum_{a \in A} \pi(a|s)\left(R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_\pi(s')\right)$$

(1) TD不需要环境模型，回报函数模型，下一个状态的概率分布

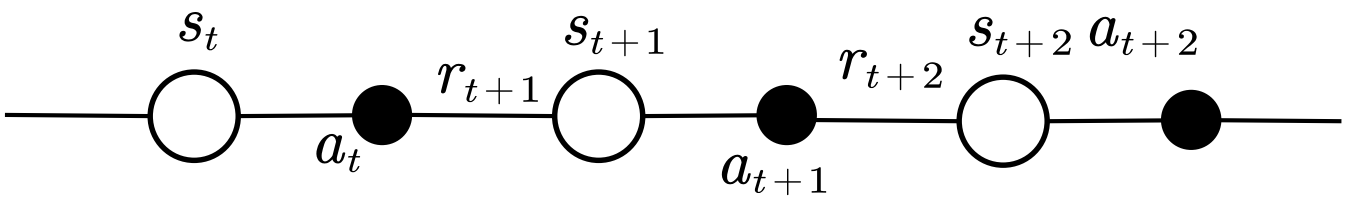
(2) 根MC相比，TD只需要等待一个时间步。

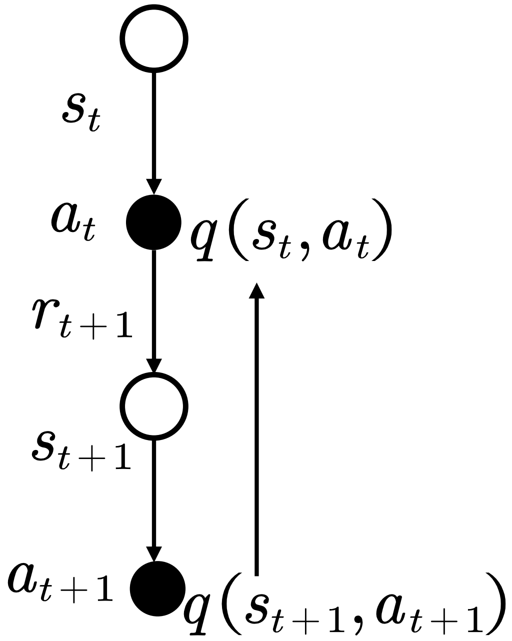
(3)TD只评估当前的动作，与后继动作没关系。

TD方法收敛已经证明

Sarsa：On-Policy TD

学习行为值函数:





$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$$

最基本的数据单元:

$$(s_t, a_t, r_t, s_{t+1}, a_{t+1})$$

这种学习方法称为：Sarsa

Sarsa：On-Policy TD

1. 初始化 $Q(s,a), \forall s \in S,\ a \in A(s)$,给定参数$\alpha$，$\gamma$

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

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) $Q(s,a) \leftarrow Q(s,a) + \alpha\left[r + \gamma Q(s',a') - Q(s,a)\right]$

(c) s=s’，a=a’

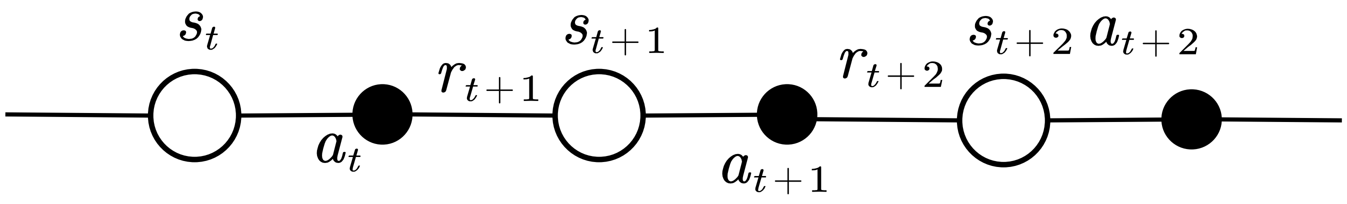
Until s 是终止状态

Until 所有的$Q(s,a)$收敛

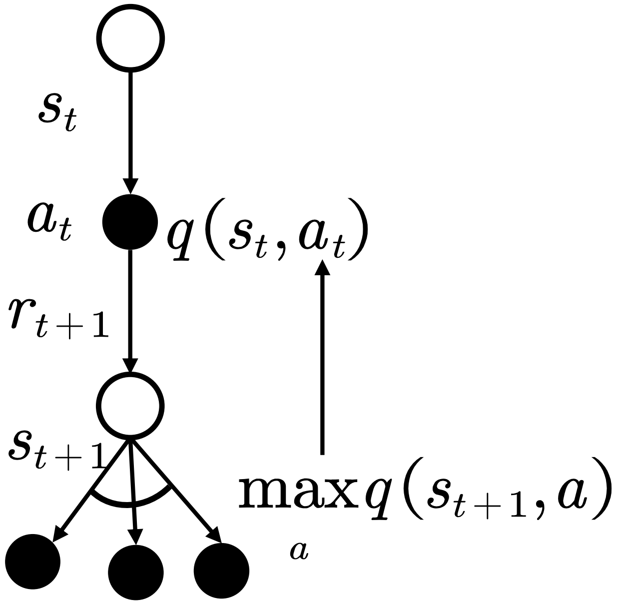
3. 输出最终策略：$\pi(s) = \underset{a}{\operatorname{argmax}}\, Q(s,a)$

Qlearning: Off-policy TD

学习行为值函数：



$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\left[r_{t+1} + \gamma \max_a Q(s_{t+1},a) - Q(s_t,a_t)\right]$$



最基本的数据单元：

(s_t, a_t, r_t, s_{t+1})

学到的行为值函数直接逼近最优行为值函数：$q_*(s,a)$

Qlearning: Off-policy TD

1. 初始化 $Q(s,a), \forall s \in S, a \in A(s)$,给定参数$\alpha$，$\gamma$

2. Repeat:

   给定起始状态 s，并根据 ε贪婪策略在状态 s 选择动作 a

   Repeat(对于一幕的每一步)

   行动策略为 ε 贪婪策略

   (a) 根据 ε 贪婪策略在状态 s_t 选择动作a_t ， 得到回报r_t 和下一个状态 s_{t+1}

   (b) $Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\left[r_t + \gamma \max_a Q(s_{t+1},a) - Q(s_t,a_t)\right]$

   目标策略为贪婪策略

   (c) s=s'， a=a'

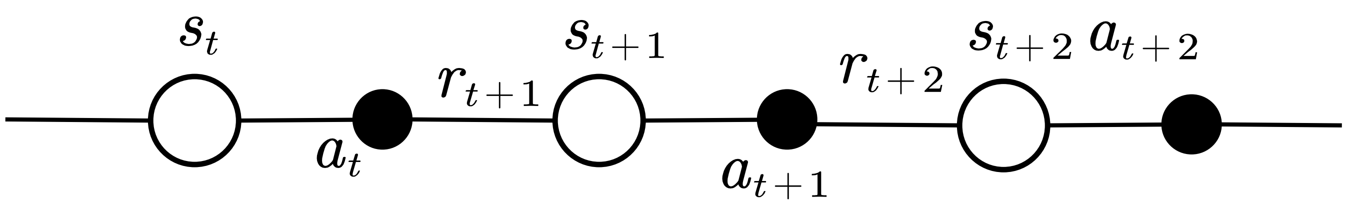
   Until s 是终止状态

   Until 所有的 $Q(s,a)$ 收敛

3. 输出最终策略：$\pi(s) = \underset{a}{\operatorname{argmax}}\, Q(s,a)$

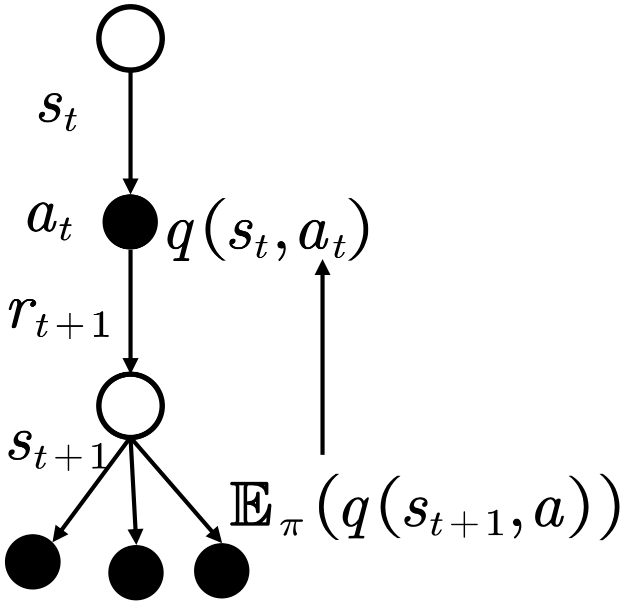
Expected Sarsa

学习行为值函数：



$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[r_{t+1} + \gamma \mathbb{E}_\pi Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)]$$

$$\leftarrow Q(s_t,a_t) + \alpha\left[r_{t+1} + \gamma \sum_a \pi(a|s_{t+1}) Q(s_{t+1},a) - Q(s_t,a_t)\right]$$



学到的行为值函数直接逼近期望行为值函数

Double Qlearning

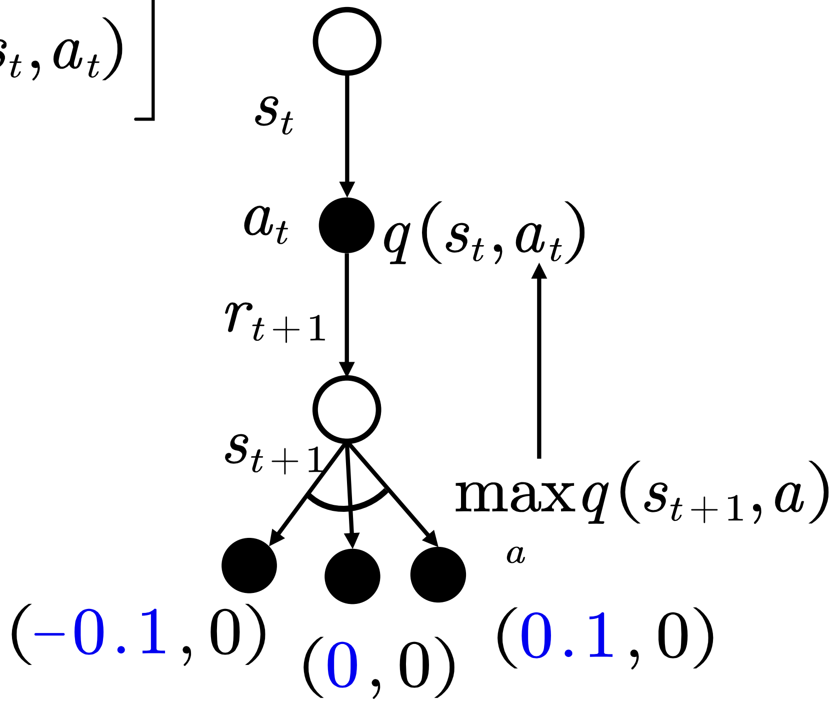
Qlearning更新：

$$Q(s_t,a_t)\leftarrow Q(s_t,a_t)+\alpha\left[r_{t+1}+\gamma\max_a Q(s_{t+1},a)-Q(s_t,a_t)\right]$$

最优化操作，容易导致偏差，称为最大化偏差

Double-Qlearning 更新：

$$Q_1(s_t,a_t)\leftarrow Q_1(s_t,a_t)+\alpha\left[r_{t+1}+\gamma Q_2\left(s_{t+1},\arg\max_a Q_1(s_{t+1},a)\right)-Q_1(s_t,a_t)\right]$$



Double Qlearning

Double Q-learning, for estimating $Q_1 \approx Q_2 \approx q_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q_1(s, a)$ and $Q_2(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, such that $Q(terminal, \cdot) = 0$

Loop for each episode:
    Initialize S
    Loop for each step of episode:
        Choose A from S using the policy ε-greedy in $Q_1 + Q_2$
        Take action A, observe R, S'
        With 0.5 probabilility:

$$Q_1(S, A) \leftarrow Q_1(S, A) + \alpha\Big(R + \gamma Q_2\big(S', \operatorname{arg\,max}_a Q_1(S', a)\big) - Q_1(S, A)\Big)$$

        else:

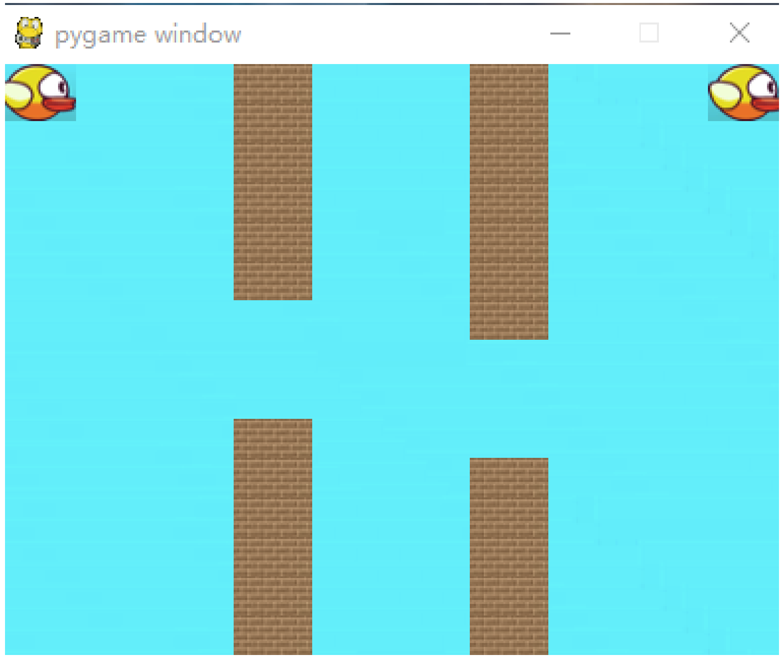
$$Q_2(S, A) \leftarrow Q_2(S, A) + \alpha\Big(R + \gamma Q_1\big(S', \operatorname{arg\,max}_a Q_2(S', a)\big) - Q_2(S, A)\Big)$$

        $S \leftarrow S'$
    until S is terminal

第四次作业

1.阅读《Reinforcement Learning： An Introduction》第五、六章

2. 利用MC方法和TD方法实现右图游戏



3. 利用MC方法和TD方法实现你自己的小游戏