强化学习原理
——无模型(MC&TD)强化学习算法

杨博渊

yby@nankai.edu.cn

2023.03.17

提纲

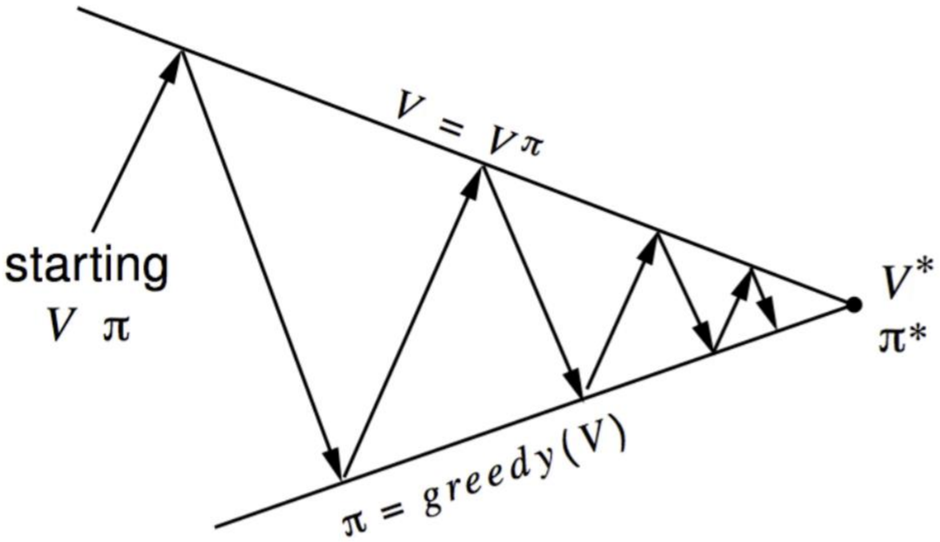
- 无模型强化学习概述
- 无模型预测
  - 蒙特卡洛
  - 时间差分
  - TD(λ)
- 无模型控制
  - On-policy蒙特卡洛
  - On-policy时间差分
  - Off-policy学习
- 总结

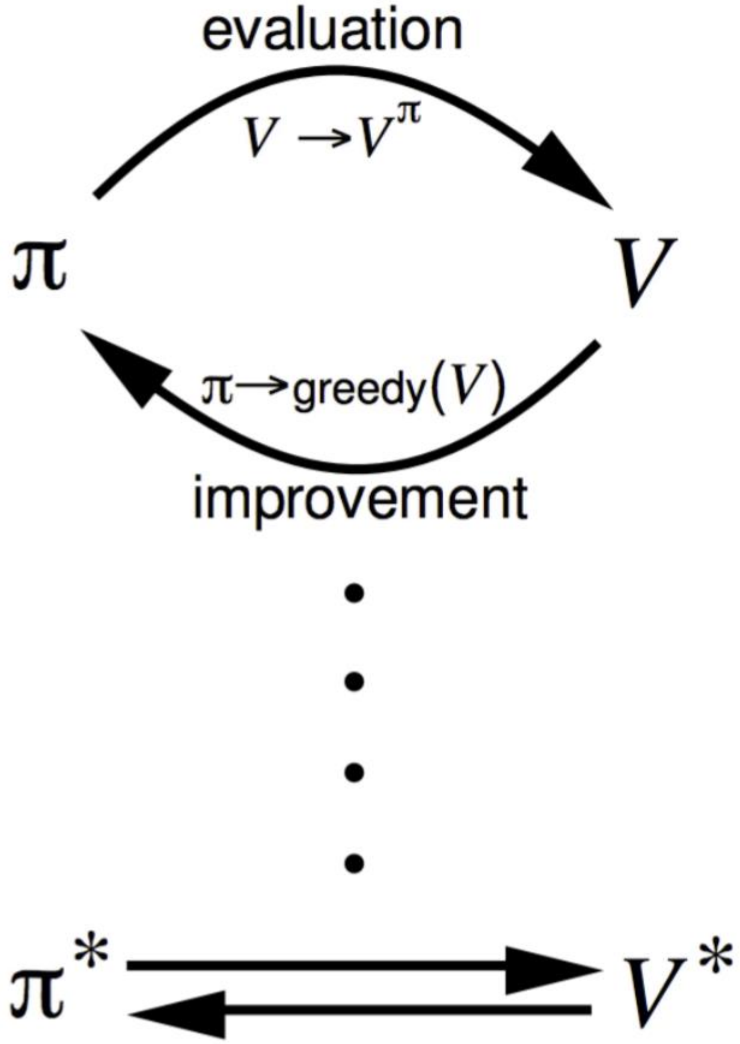
同策略与异策略学习

- On-policy学习
  - 在任务中学习
  - 从策略π的样本经验中学习策略π

- Off-policy学习
  - 在“别人的肩膀上”学习
  - 从策略μ的样本经验中学习策略π

广义策略迭代





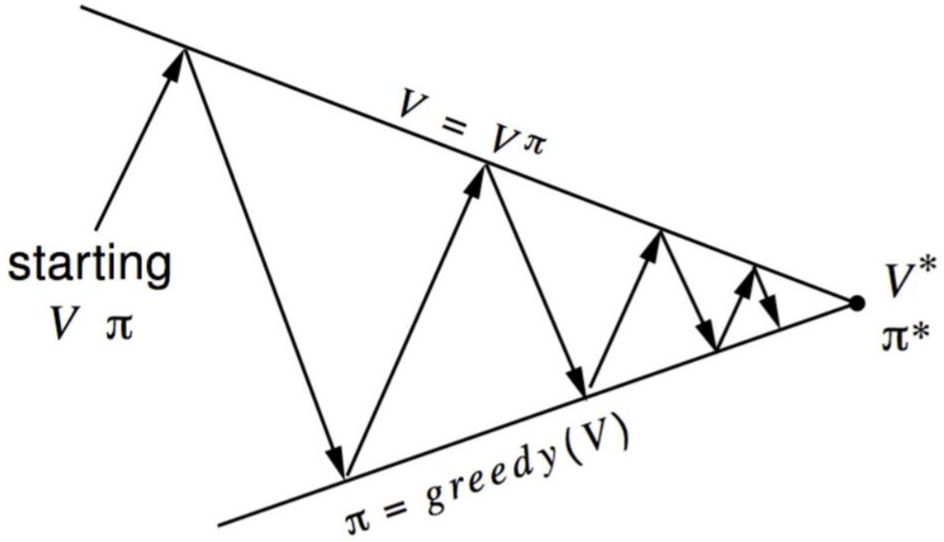
策略评估 估计v_π

例如迭代策略评估

策略改进 生成$\pi' > \pi$

例如贪婪策略改进

蒙特卡洛评估的广义策略迭代



策略评估 蒙特卡洛策略评估，$\mathrm{V} = v_\pi$?

策略改进 贪婪策略改进?

利用行为值函数策略迭代

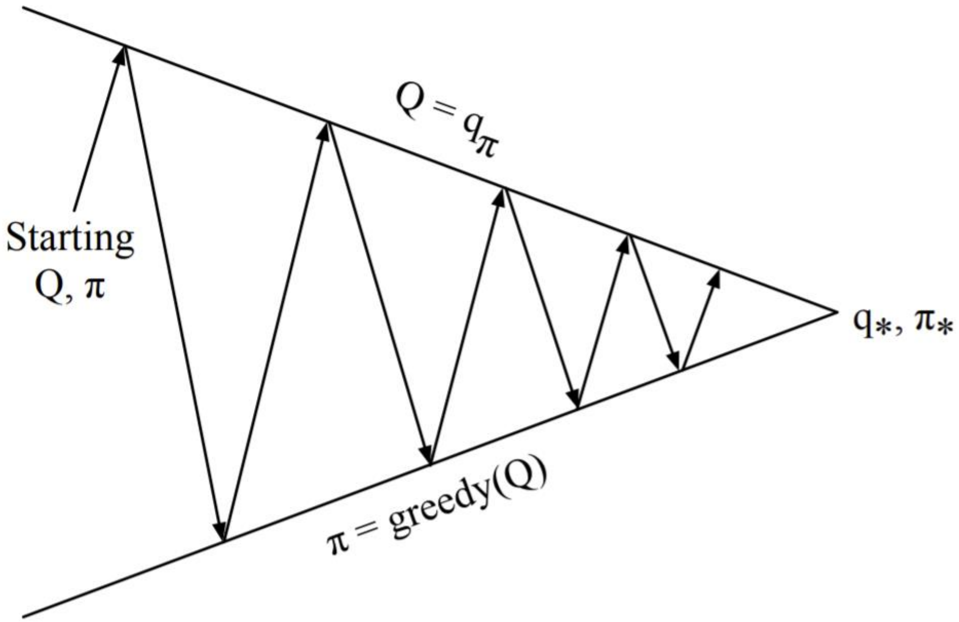
- MDP模型的贪婪策略改进

$$\pi'(s) = \underset{a \in \mathcal{A}}{\operatorname{argmax}}\ \mathcal{R}_s^a + \mathcal{P}_{ss'}^a V(s')$$

- 无模型的贪婪策略改进

$$\pi'(s) = \underset{a \in \mathcal{A}}{\operatorname{argmax}}\ Q(s, a)$$

蒙特卡洛评估的广义策略迭代



策略评估 蒙特卡洛策略评估，$\mathrm{Q} = q_\pi$

策略改进 贪婪策略改进?

贪婪行为选择示例

你打开左侧门得到即时奖励为 0： $V(left) = 0$；

你打开右侧门得到即时奖励 1： $V(right) = +1$；

在使用贪婪算法时，接下来你将会继续打开右侧的门，而不会尝试打开左侧门

你打开右侧门得到即时奖励 + 3： $V(right) = +2$；

你打开右侧门得到即时奖励 + 2： $V(right) = +2$；

...

- 最优选择？

ε-greedy探索

$$\pi(a|s) = \begin{cases} \epsilon/m + 1 - \epsilon & \text{if } a^* = \underset{a \in \mathcal{A}}{\operatorname{argmax}}\ Q(s,a) \\ \epsilon/m & \text{otherwise} \end{cases}$$

- 保证持续探索的简单想法
- 全部m个动作都有几率被选择
- 1-ε概率选择贪婪动作
- ε概率随机在动作集合中选择一个

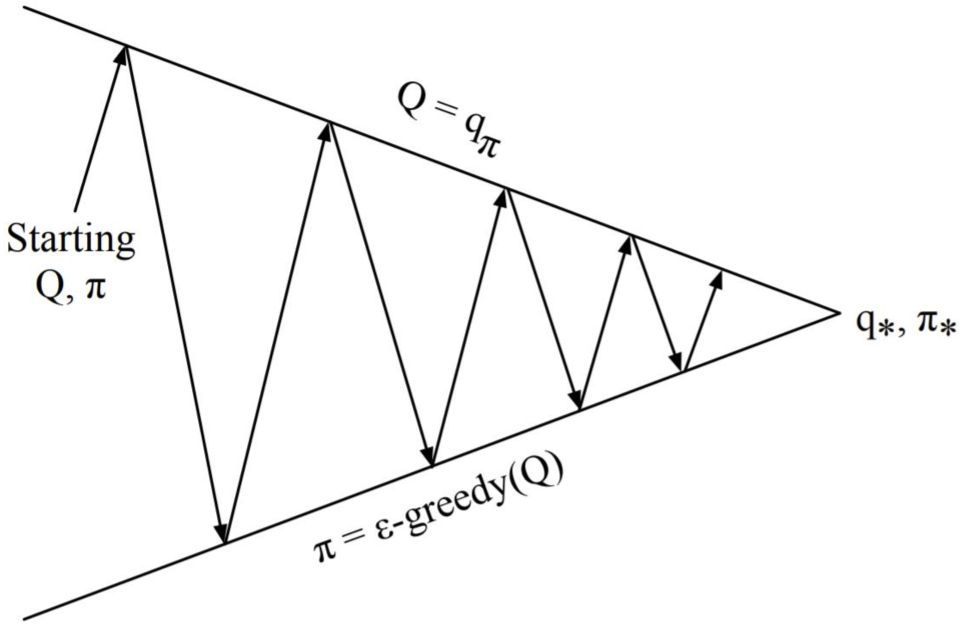
ε-greedy策略改进

■ 定理

对于任意的ε-greedy策略π，如果ε-greedy策略π'的q_π是改进的，那么$v_{\pi'}(s) \geq v_\pi(s)$

$$
\begin{aligned}
q_\pi(s,\pi'(s)) &= \sum_{a\in\mathcal{A}} \pi'(a|s) q_\pi(s,a) \\
&= \epsilon/m \sum_{a\in\mathcal{A}} q_\pi(s,a) + (1-\epsilon)\max_{a\in\mathcal{A}} q_\pi(s,a) \\
&\geq \epsilon/m \sum_{a\in\mathcal{A}} q_\pi(s,a) + (1-\epsilon)\sum_{a\in\mathcal{A}} \frac{\pi(a|s)-\epsilon/m}{1-\epsilon} q_\pi(s,a) \\
&= \sum_{a\in\mathcal{A}} \pi(a|s) q_\pi(s,a) = v_\pi(s)
\end{aligned}
$$

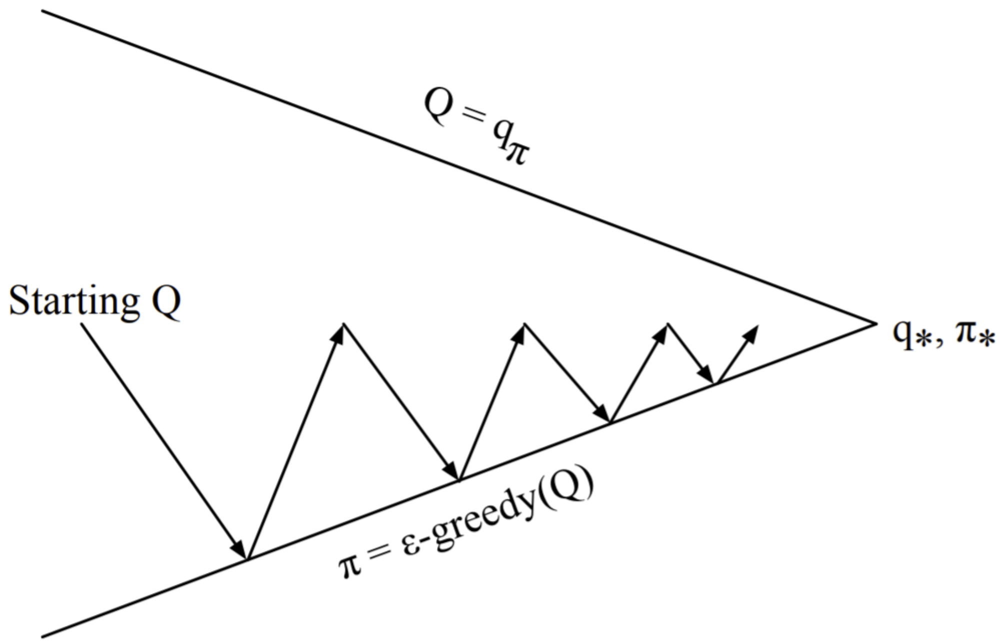
蒙特卡洛策略迭代



策略评估 蒙特卡洛策略评估，$\mathrm{Q} = q_\pi$

策略改进 ε-greedy策略改进

蒙特卡洛控制



每一episode

策略评估 蒙特卡洛策略评估，$Q \approx q_\pi$

策略改进 ε-greedy策略改进

GLIE

■ 定义

GLIE： Greedy in the Limit with Infinite Exploration

所有已经经历的状态行为对会被无限次探索

$$\lim_{k\to\infty} N_k(s,a) = \infty$$

策略收敛至一个贪婪的策略

$$\lim_{k\to\infty} \pi_k(a|s) = \mathbf{1}(a = \operatorname*{argmax}_{a'\in\mathcal{A}} Q_k(s,a'))$$

GLIE蒙特卡洛控制

对于给定策略 π， 采样第 k 个Episode: $\{S_1, A_1, R_2, \ldots, S_T\} \sim \pi$

对于该Episode里出现的每一个状态行为对 S_t 和 A_t ,更其计数和Q函数:

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

$$Q(S_t, A_t) \leftarrow + \frac{1}{N(S_t, A_t)}(G_t - Q(S_t, A_t))$$

基于新的Q函数改善以如下方式改善策略:

$$\epsilon \leftarrow 1/k$$

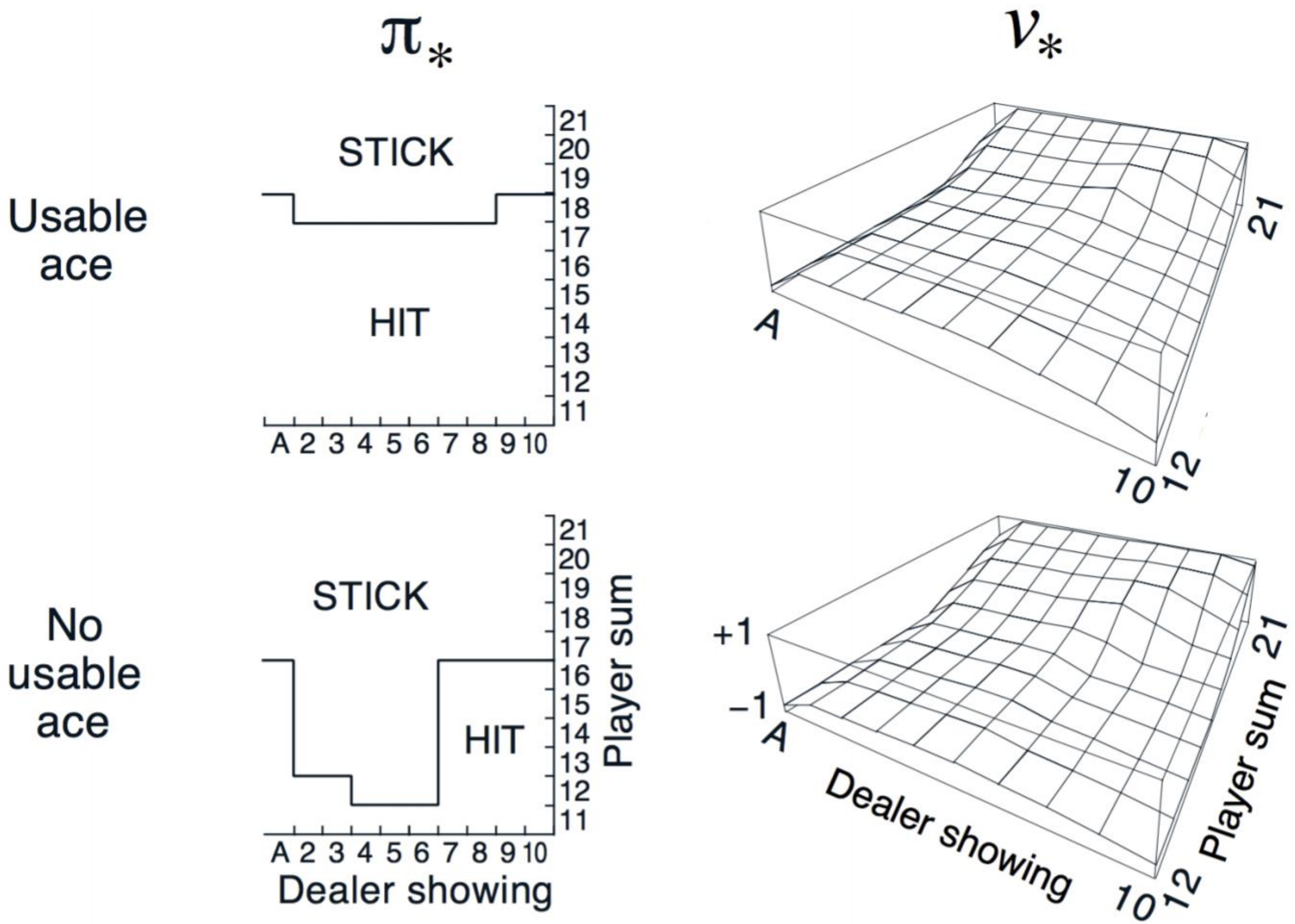
$$\pi \leftarrow \epsilon - greedy(Q)$$

- 定理

GLIE蒙特卡洛控制能收敛至最优的状态行为值函数 $Q(s, a) \rightarrow q_*(s, a)$

21点示例

21点示例



π_*
v_*
Usable ace
STICK
HIT
21 20 19 18 17 16 15 14 13 12 11
A 2 3 4 5 6 7 8 9 10
A
10
12
21
No usable ace
STICK
HIT
Player sum
Dealer showing
+1
−1
A
Dealer showing
10
12
Player sum
21

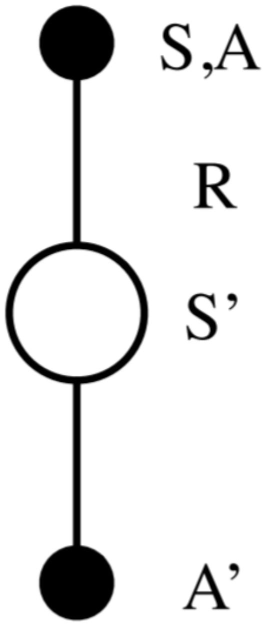
提纲

- 无模型强化学习概述
- 无模型预测
  - 蒙特卡洛
  - 时间差分
  - TD(λ)
- 无模型控制
  - On-policy蒙特卡洛
  - On-policy时间差分
  - Off-policy学习
- 总结

MC vs. TD 控制

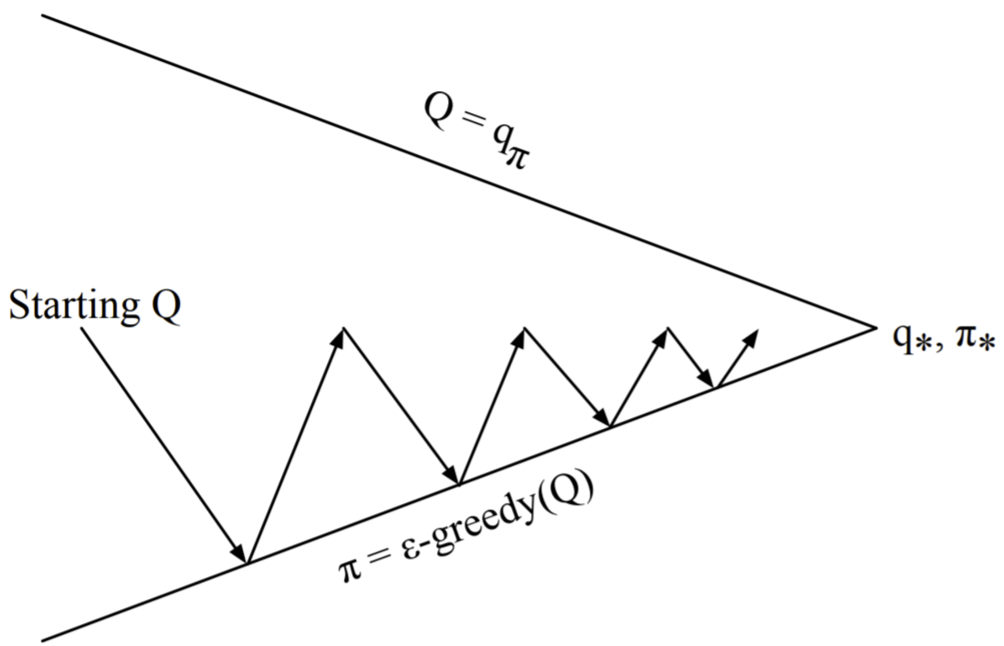
- 相比于MC，TD具有以下多种优势
  - 更小的方差
  - 可以在线实时学习
  - 可以学习不完整Episode
- 一个自然的想法：在控制回路中用TD代替MC
  - TD目标应用到Q(S,A)
  - 应用ε-greedy策略改进
  - 每个时间步都进行更新

Sarsa：动作值函数更新



$$Q(S,A) \leftarrow Q(S,A) + \alpha \left(R + \gamma Q(S',A') - Q(S,A)\right)$$

Sarsa：同策略控制



每一时间步

策略评估 Sarsa，$Q \approx q_\pi$

策略改进 ε-greedy策略改进

Sarsa：算法

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

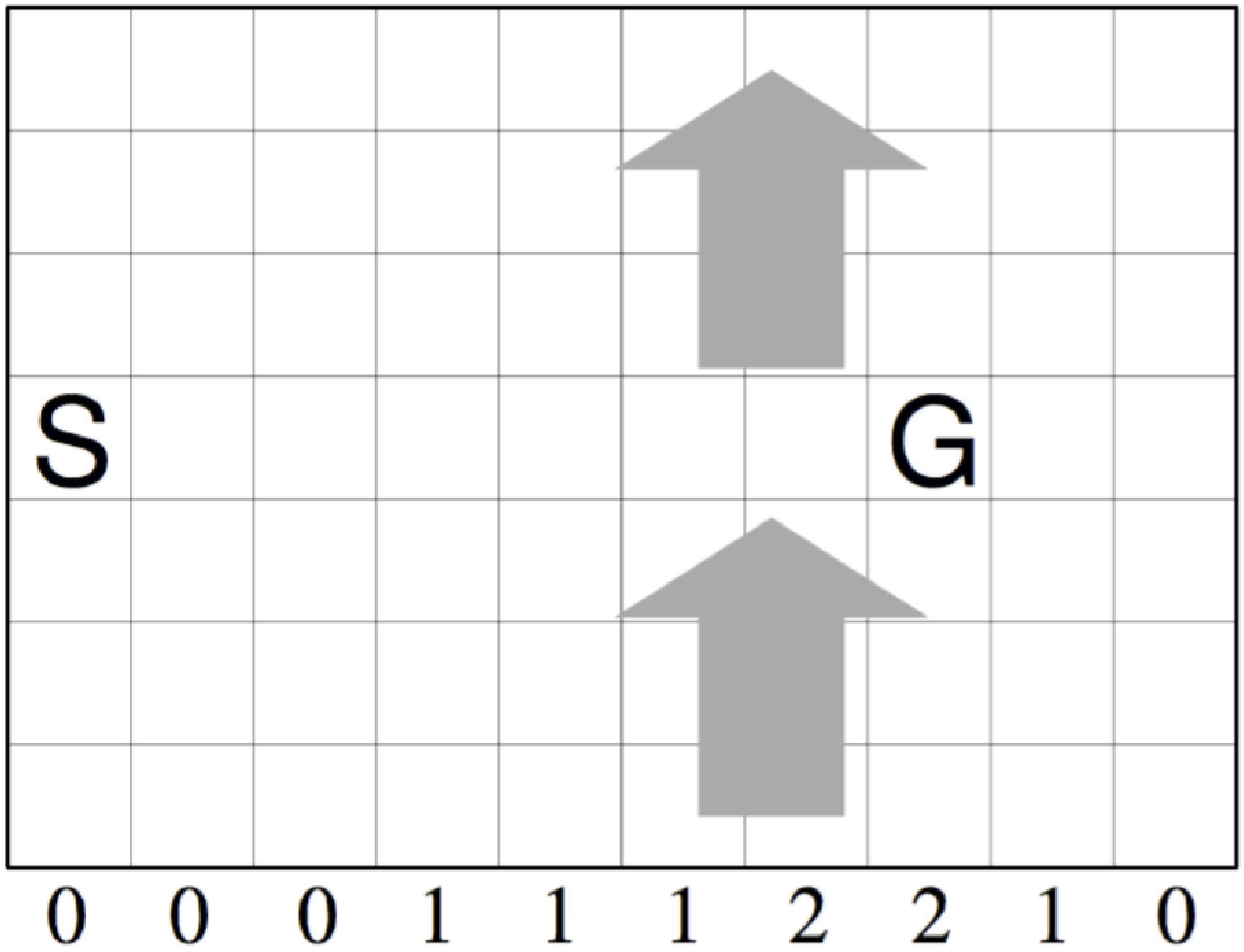
Sarsa：收敛性

定理：满足如下两个条件时，Sarsa算法将收敛至最优动作值函数

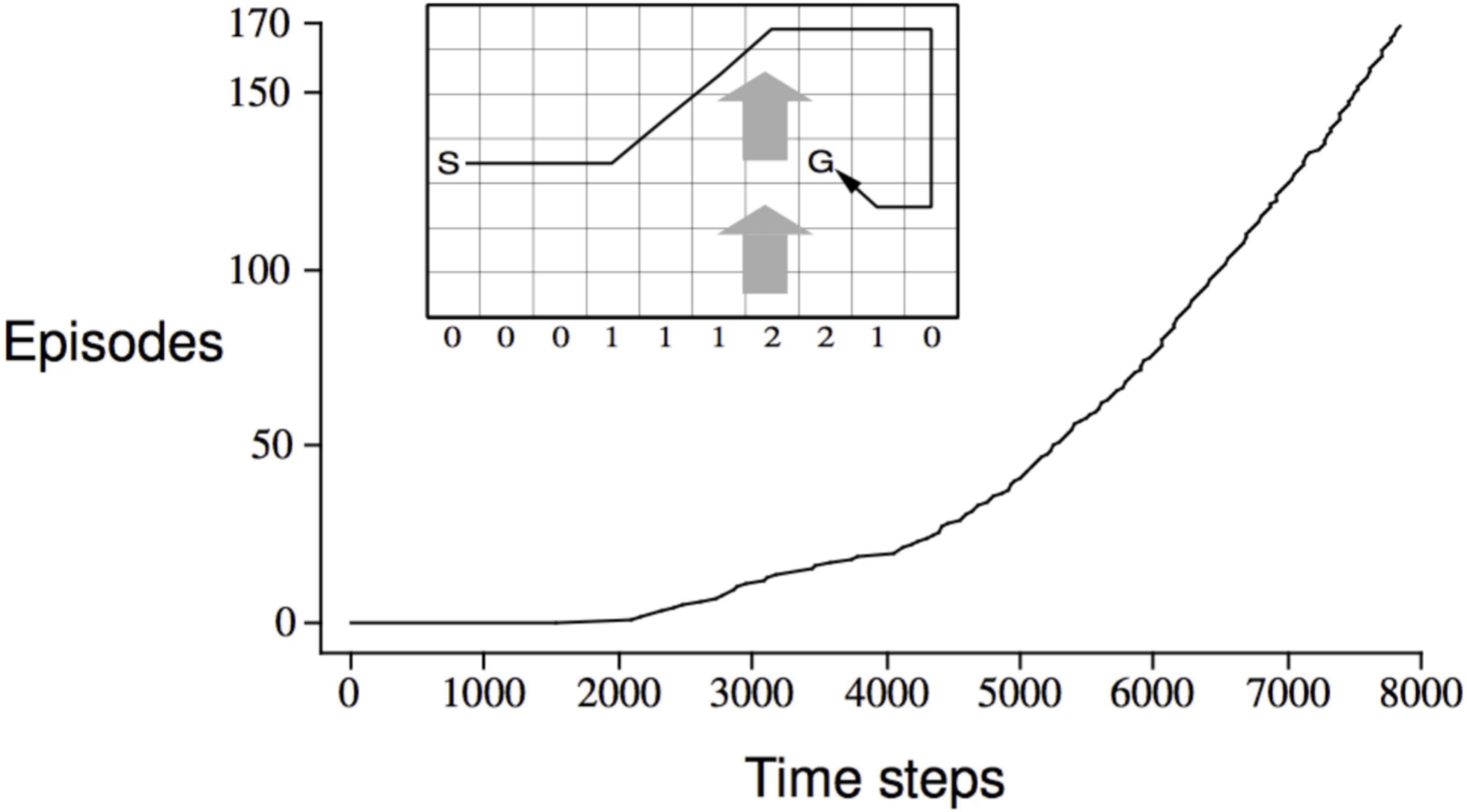
条件一：任何时候的策略 $\pi_t(a|s)$ 符合GLIE特性;

条件二：步长系数αt满足：$\sum_{t=1}^{\infty} a_t = \infty$ 且 $\sum_{t=1}^{\infty} a_t^2 < \infty$

示例：有风格子世界



示例：有风格子世界


170
150
100
Episodes
50
0
S
G
0 0 0 1 1 1 2 2 1 0
0 1000 2000 3000 4000 5000 6000 7000 8000
Time steps

提纲

- 无模型强化学习概述
- 无模型预测
  - 蒙特卡洛
  - 时间差分
  - TD(λ)
- 无模型控制
  - On-policy蒙特卡洛
  - On-policy时间差分
  - Off-policy学习
- 总结

Off-policy学习

- 评估目标策略$\pi(a|s)$，计算$v_\pi(s)$或$q_\pi(s,a)$
- 遵循行为策略$\mu(a|s)$

$$\{S_1, A_1, R_2, ..., S_T\} \sim \mu$$

- 可以通过观测人类或其它智能体学习
- 可以重复利用旧策略生成的经验
- 遵循探索策略学习出最优策略
- 遵循同一策略学习出多种策略

重要性采样

- 估计不同分布的期望

$$
\begin{aligned}
\mathbb{E}_{X\sim P}[f(X)] &= \sum P(X)f(X) \\
&= \sum Q(X)\frac{P(X)}{Q(X)}f(X) \\
&= \mathbb{E}_{X\sim Q}\left[\frac{P(X)}{Q(X)}f(X)\right]
\end{aligned}
$$

MC重要性采样

- 利用μ生成的回报评估π
- 利用策略相似度加权回报G_t
- 重要性采样率连乘

$$G_t^{\pi/\mu} = \frac{\pi(A_t|S_t)}{\mu(A_t|S_t)} \frac{\pi(A_{t+1}|S_{t+1})}{\mu(A_{t+1}|S_{t+1})} \cdots \frac{\pi(A_T|S_T)}{\mu(A_T|S_T)} G_t$$

- 修正回报更新值函数

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

TD重要性采样

- 利用μ生成的TD目标评估π
- 利用策略相似度加权TD目标
- 重要性采样修正

$$V(S_t) \leftarrow V(S_t) + \alpha \left(\frac{\pi(A_t|S_t)}{\mu(A_t|S_t)} (R_{t+1} + \gamma V(S_{t+1})) - V(S_t) \right)$$

Q-Learning

- 考虑动作值函数$Q(s,a)$的off-policy学习

- 下一动作通过行为策略选择 $A_{t+1} \sim \mu(\cdot|S_t)$

- 但同时考虑可替代的后续动作 $A' \sim \pi(\cdot|S_t)$

- 更新行为值函数

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left(R_{t+1} + \gamma Q(S_{t+1}, A') - Q(S_t, A_t) \right)$$

Q-Learning

- 行为策略和目标策略都进行更新
- 目标策略π是贪婪的

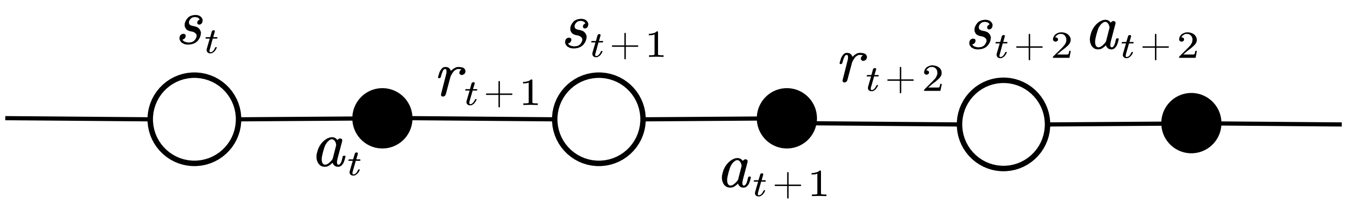
$$\pi(S_{t+1}) = \underset{a'}{\operatorname{argmax}}\ Q(S_{t+1}, a')$$

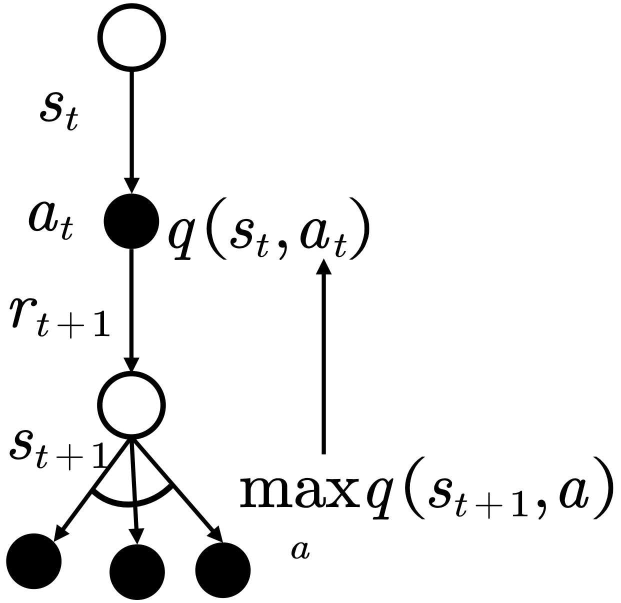
- 行为策略μ是ε-贪婪的
- Q-learning目标：

$$\begin{aligned} & R_{t+1} + \gamma Q(S_{t+1}, A') \\ = & R_{t+1} + \gamma Q(S_{t+1}, \underset{a'}{\operatorname{argmax}}\ Q(S_{t+1}, a')) \\ = & R_{t+1} + \max_{a'} \gamma Q(S_{t+1}, a') \end{aligned}$$

Q-Learning

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：$q_*(s, a)$

Q-Learning

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

  Repeat (for each step of episode):

    Choose A from S using policy derived from Q (e.g., ε-greedy)

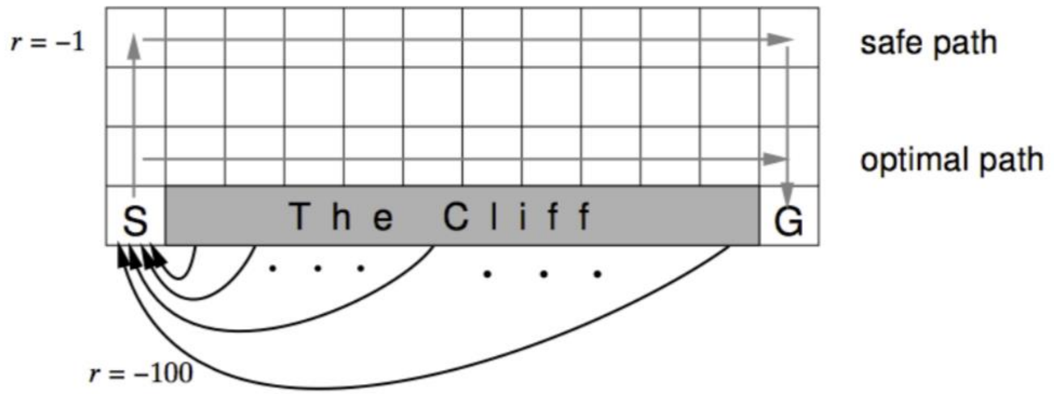
    Take action A, observe R, S'

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

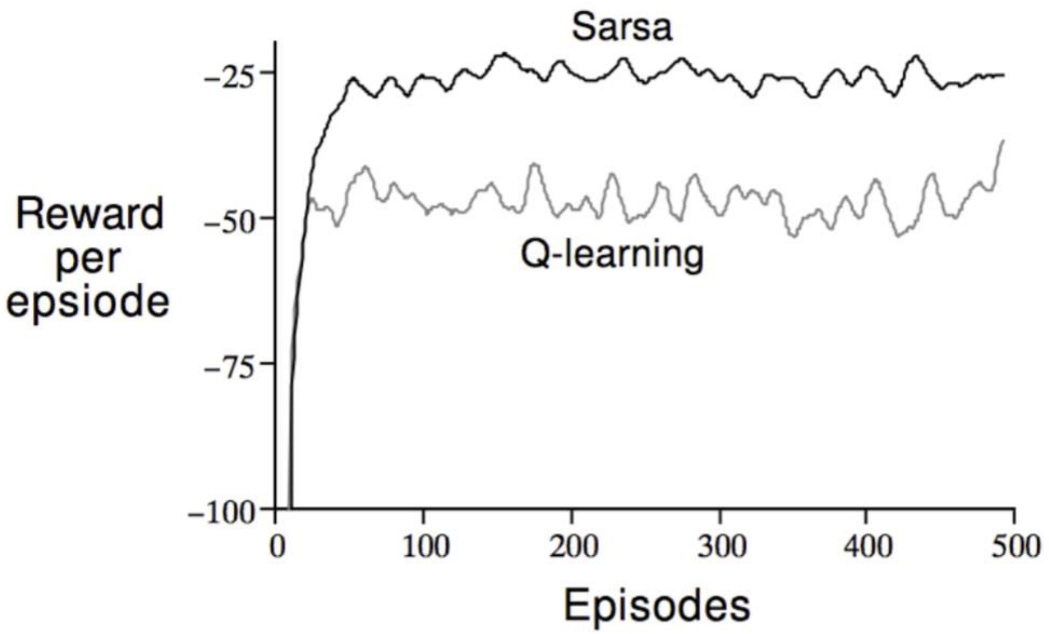
    $S \leftarrow S'$;

  until S is terminal

示例：悬崖行走



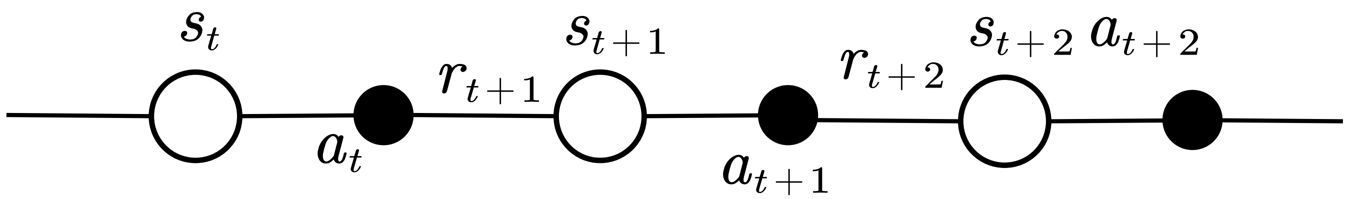
r = −1
safe path
optimal path
S
The Cliff
G
r = −100



Sarsa
Q-learning
Reward per epsiode
−25
−50
−75
−100
0
100
200
300
400
500
Episodes

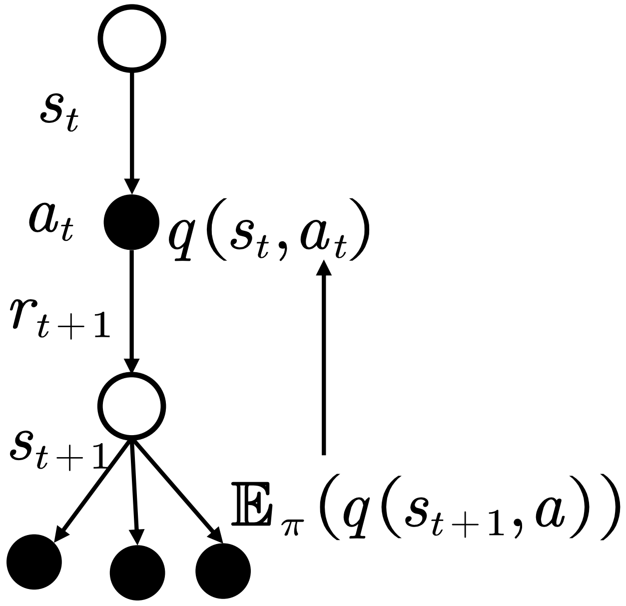
Expected Sarsa

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 Q-Learning

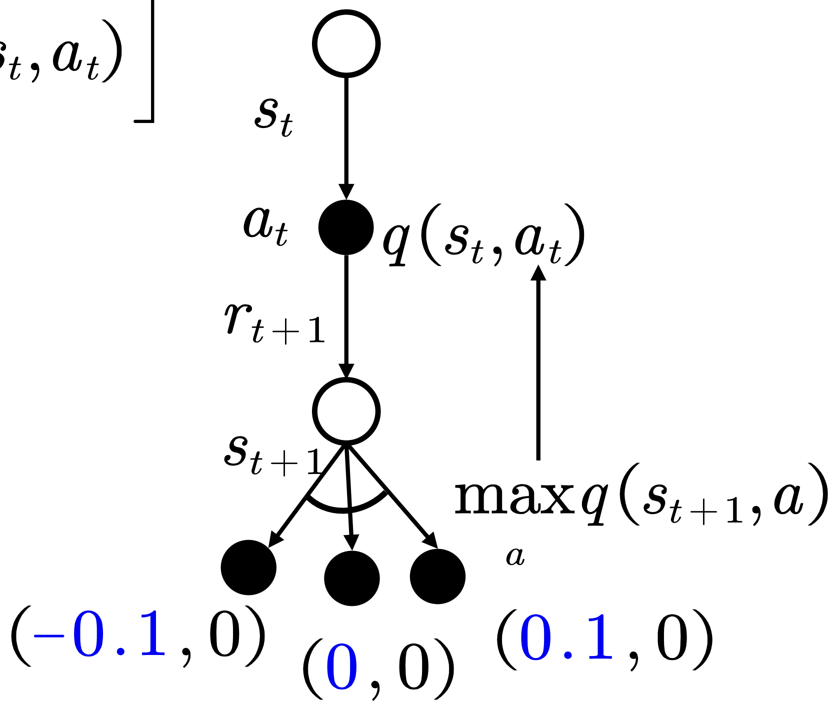
Qlearning更新：

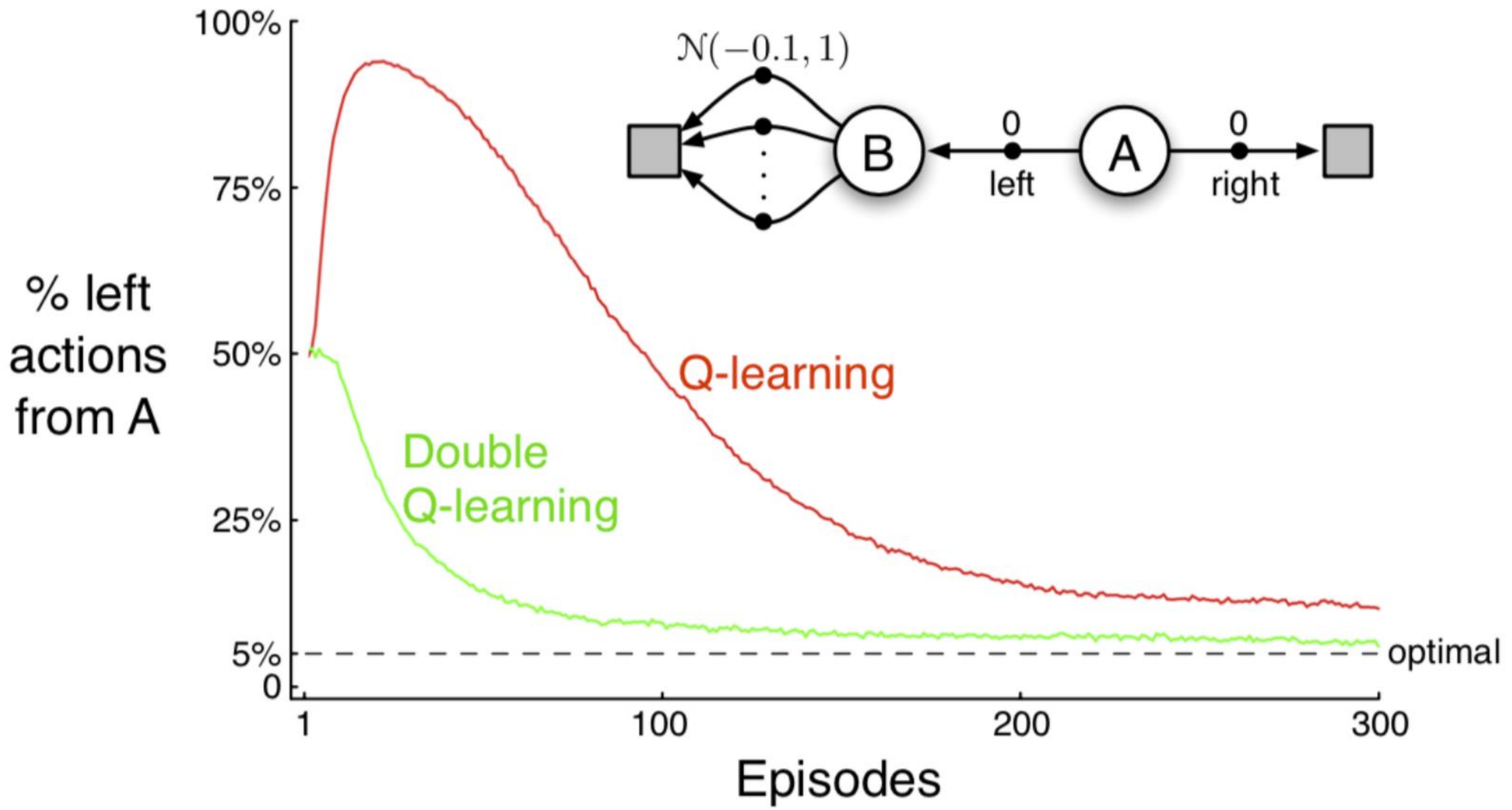
$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\left[r_{t+1} + \gamma \max_a Q(s_{t+1},a) - Q(s_t,a_t)\right]$$

最优化操作，容易导致偏差，称为最大化偏差

Double-Qlearning 更新：

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





100%
75%
50%
25%
5%
0
% left actions from A
$\mathcal{N}(-0.1, 1)$
B
0
left
A
0
right
Q-learning
Double Q-learning
optimal
1
100
200
300
Episodes

Double Q-Learning

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:

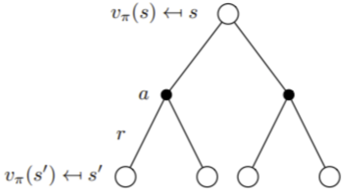
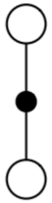
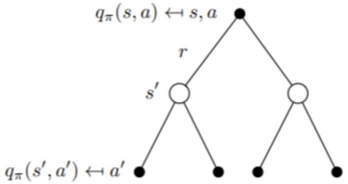
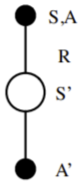
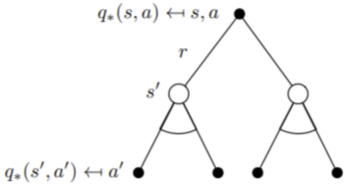
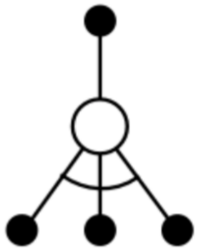
$$Q_1(S, A) \leftarrow Q_1(S, A) + \alpha\Big(R + \gamma Q_2\big(S', \arg\max_a Q_1(S', a)\big) - Q_1(S, A)\Big)$$

        else:

$$Q_2(S, A) \leftarrow Q_2(S, A) + \alpha\Big(R + \gamma Q_1\big(S', \arg\max_a Q_2(S', a)\big) - Q_2(S, A)\Big)$$

        $S \leftarrow S'$
    until S is terminal

DP与TD

	Full Backup (DP)	*Sample Backup (TD)*
Bellman Expectation Equation for $v_\pi(s)$	 Iterative Policy Evaluation	 TD Learning
Bellman Expectation Equation for $q_\pi(s, a)$	 Q-Policy Iteration	 Sarsa
Bellman Optimality Equation for $q_*(s, a)$	 Q-Value Iteration	 Q-Learning

DP与TD

Full Backup (DP)	*Sample Backup (TD)*
Iterative Policy Evaluation	TD Learning
$V(s) \leftarrow \mathbb{E}[R + \gamma V(S') \mid s]$	$V(S) \xleftarrow{\alpha} R + \gamma V(S')$
Q-Policy Iteration	Sarsa
$Q(s,a) \leftarrow \mathbb{E}[R + \gamma Q(S',A') \mid s,a]$	$Q(S,A) \xleftarrow{\alpha} R + \gamma Q(S',A')$
Q-Value Iteration	Q-Learning
$Q(s,a) \leftarrow \mathbb{E}\left[R + \gamma \max_{a' \in \mathcal{A}} Q(S',a') \mid s,a\right]$	$Q(S,A) \xleftarrow{\alpha} R + \gamma \max_{a' \in \mathcal{A}} Q(S',a')$

其中 $x \xleftarrow{\alpha} y \equiv x \leftarrow x + \alpha(y - x)$

第四次作业

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

2. 阅读MC方法和TD方法的实现

3. 利用MC方法和TD方法实现你自己的小游戏