强化学习

第五讲：无模型控制学习

教师：赵冬斌 朱圆恒 张启超

中国科学院大学
中国科学院自动化研究所

March 31, 2022

上节回顾

无模型预测学习

- 蒙特卡洛方法
- 时间差分学习
- TD(λ)

预测 vs 控制

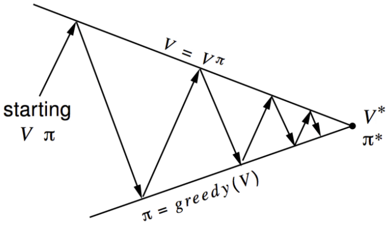
- 预测问题 prediction
  - 计算 MDP 某一策略的价值函数
- 控制问题 control
  - 计算 MDP 的最优策略/最优价值函数

预测 vs 控制

- 预测问题 prediction
  - 计算 MDP 某一策略的价值函数
- 控制问题 control
  - 计算 MDP 的最优策略/最优价值函数
- 无模型控制学习 model-free control learning
  - 要么 MDP 本身的模型未知，只能根据观测的经验数据学习，e.g. 参数未知的机械臂
  - 要么 MDP 太复杂，状态空间过大，根据在线样本训练更有效，e.g. 围棋

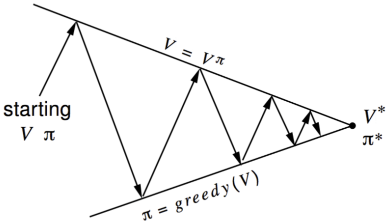
蒙特卡洛控制

策略迭代 (回顾)



- 策略评估: 估计 V_π
  1. 矩阵求解贝尔曼期望方程
  2. 迭代策略评估
- 策略提升: 生成新的 $\pi' \geq \pi$
  - 贪心策略提升

策略迭代 (回顾)



- 策略评估: 估计 V_π (蒙特卡洛学习/时间差分学习)
  1. 矩阵求解贝尔曼期望方程
  2. 迭代策略评估
- 策略提升: 生成新的 $\pi' \geq \pi$
  - 贪心策略提升

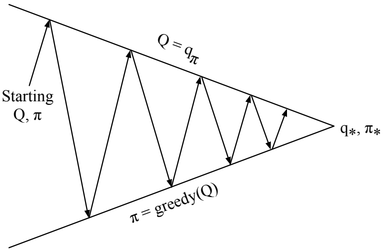
动作 -价值函数

- 基于 $V(s)$ 进行贪心策略提升需要 MDP 的模型信息

$$\pi'(s) = \mathcal{G}(V) = \arg\max_{a\in\mathcal{A}} \left(\mathcal{R}_s^a + \gamma \sum_{s'\in\mathcal{S}} \mathcal{P}_{ss'}^a V(s')\right)$$

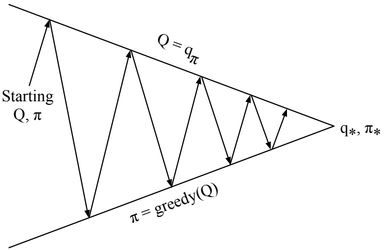
- 基于 $Q(s,a)$ 进行贪心策略提升是无模型的

$$\pi'(s) = \mathcal{G}(Q) = \arg\max_{a\in\mathcal{A}} Q(s,a)$$



- 无模型策略评估: MC 学习 $Q = Q_\pi$
- 无模型策略提升: 贪心策略提升 $\pi' = \mathcal{G}(Q)$ 确定性策略

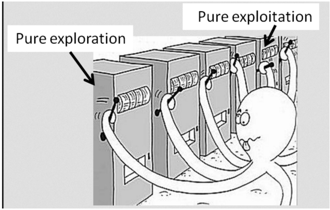
蒙特卡洛学习 + 动作 -价值函数的策略迭代



- 无模型策略评估: MC 学习 $Q = Q_\pi$
- 无模型策略提升: 贪心策略提升 $\pi' = \mathcal{G}(Q)$ 确定性策略

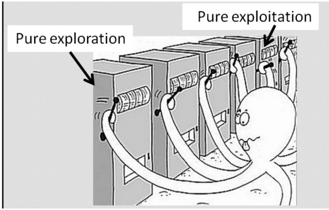
- 但这样无模型的策略迭代一定能学到最优策略吗?

举例：贪心动作选择



- 假定你现在在玩一组老虎机，每个机子的收益满足一定随机概率分布，不同的机子概率分布不同且未知
- 第 1 次策略是每个机子都试玩下，得到收益率是 1.1, 1.2, 0.9, 0.8, 1.05, 0.85

举例：贪心动作选择



- 假定你现在在玩一组老虎机，每个机子的收益满足一定随机概率分布，不同的机子概率分布不同且未知
- 第 1 次策略是每个机子都试玩下，得到收益率是 1.1, 1.2, 0.9, 0.8, 1.05, 0.85
- 第二次再玩你的策略是什么？
  - 是把钱都放在第 2 个机子上（确定性策略, exploitability）
  - 还是分出一部分钱放在别的机子上看看这次收益会是什么样（随机性策略, explorability）

- 假设 MDPs 是确定型的, $\mathcal{R}$ 和 $\mathcal{P}$ 都是确定性的
- 对一个 确定性 (deterministic) 的策略 π 使用 MC 方法评估它的 Q 函数会有什么问题?

$$s_0, a_0 = \pi(s_0), s_1, a_1 = \pi(s_1), \ldots$$

- 假设 MDPs 是确定型的, $\mathcal{R}$ 和 $\mathcal{P}$ 都是确定性的
- 对一个确定性 (deterministic) 的策略 π 使用 MC 方法评估它的 Q 函数会有什么问题?

$$s_0, a_0 = \pi(s_0), s_1, a_1 = \pi(s_1), \ldots$$

- 在状态 s_0 下只能观测到动作 $a_0 = \pi(s_0)$ 对应的回报
- 无法产生其它 $(s_0, a), a \neq \pi(s_0)$ 下的轨迹
- Q 函数只和状态有关, $Q(s_0, a_0) = Q(s_0, \pi(s_0))$
- 所以需要在策略中加入探索动作增加轨迹的多样性, 使得 Q 函数更精确, 完成下一步的策略提升

ϵ-贪心探索

- 在线学习需要具有探索性的策略
- 保证获得尽可能全面的模型观测数据
- ϵ-贪心策略是最简单的一种探索策略
- 所有动作都有非零概率被选中执行

$$
\begin{aligned}
\pi(s) =& \epsilon\text{-greedy}(Q)(s) \\
=& \begin{cases} \arg\max_{a\in\mathcal{A}} Q(s,a) & \text{with } 1-\varepsilon \text{ probability} \\ \text{random } a \in \mathcal{A} & \text{with } \epsilon \text{ probability} \end{cases}
\end{aligned}
$$

ϵ-贪心策略提升

定理

给定一个 ϵ-贪心策略 π 和它的动作 -价值 Q_π，策略提升得到的新 ϵ-贪心策略 π' 比之前的策略 π 要好，即 $V_{\pi'}(s) \geq V_\pi(s)$

ϵ-贪心策略提升

定理

给定一个 ϵ-贪心策略 π 和它的动作 -价值 Q_π，策略提升得到的新 ϵ-贪心策略 π' 比之前的策略 π 要好，即 $V_{\pi'}(s) \geq V_\pi(s)$

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

ϵ-贪心策略提升

定理

给定一个 ϵ-贪心策略 π 和它的动作 -价值 Q_π，策略提升得到的新 ϵ-贪心策略 π' 比之前的策略 π 要好，即 $V_{\pi'}(s) \geq V_\pi(s)$

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

$$Q_\pi(s, \pi'(s)) = \mathcal{T}_{\pi'}(V_\pi) \geq V_\pi(s)$$

ϵ-贪心策略提升

定理

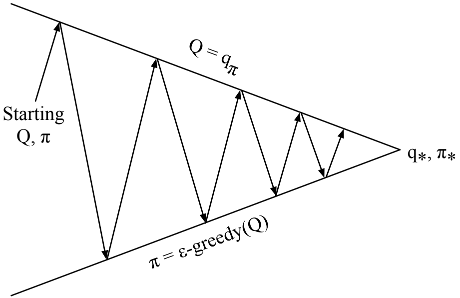
给定一个 ϵ-贪心策略 π 和它的动作 -价值 Q_π，策略提升得到的新 ϵ-贪心策略 π' 比之前的策略 π 要好，即 $V_{\pi'}(s) \geq V_\pi(s)$

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

$$
Q_\pi(s, \pi'(s)) = \mathcal{T}_{\pi'}(V_\pi) \geq V_\pi(s)
$$

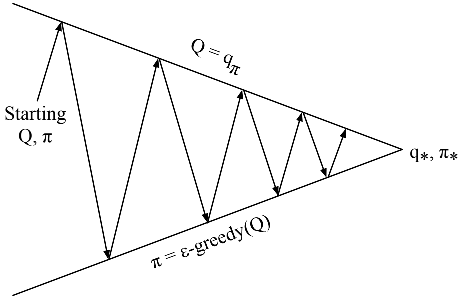
以此类推，$V_\pi(s) \leq \ldots \mathcal{T}_{\pi'}^{\infty} \leq (V_\pi) = V_{\pi'}(s)$

MC 策略迭代



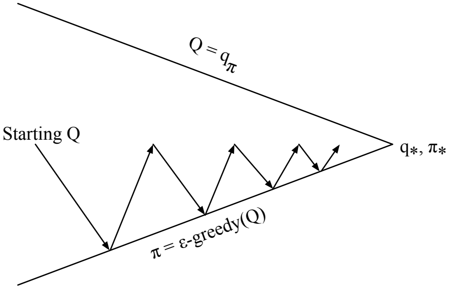
- 策略评估：MC 学习 $Q = Q_\pi$
- 策略提升：ϵ- 贪心策略提升

MC 策略迭代



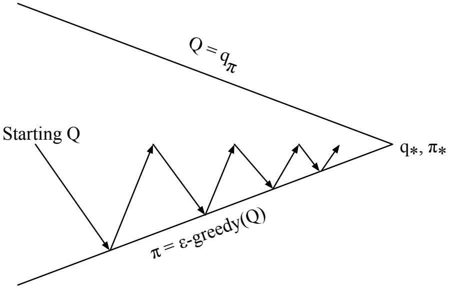
- 策略评估: MC 学习 $Q = Q_\pi$
  - (每次需要多条轨迹完成精确的评估后才进行策略提升)
- 策略提升: ϵ- 贪心策略提升

MC 控制学习



- 能否每生成一个轨迹:
  1. 策略评估: MC 学习 $Q \simeq Q_\pi$
  2. 策略提升: ϵ-贪心策略提升
- 然后再生成下一条轨迹继续迭代

MC 控制学习



- 能否每生成一个轨迹:
  1. 策略评估: MC 学习 $Q \simeq Q_\pi$
  2. 策略提升: ϵ-贪心策略提升
- 然后再生成下一条轨迹继续迭代

每次都是不精确的策略评估 (一次轨迹的 MC 评估) 能否收敛到最优策略?

定义

无限探索，无穷时刻收敛为贪心策略(Greedy in the Limit with Infinite Exploration, GLIE) 的含义是

- 智能体无限次数地探索所有的状态 -动作对

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

- 策略在无穷时刻收敛到贪心策略

$$\lim_{k\to\infty} \pi_k(a|s) = \mathcal{I}(a = \arg\max_{a'\in\mathcal{A}} Q_k(s,a'))$$

- 如果 ϵ-贪心策略的探索率 ϵ 随时间衰减到零 $\lim_{k\to\infty}\epsilon_k = 0$，那么它就满足 GLIE
- e.g. $\epsilon_k = \frac{1}{k}$

```
初始化 Q(s,a) = 0, N(s,a) = 0, ε = 1, k = 1
π_k = ε-greedy(Q)
loop
    使用策略 π_k 采集第 k 个轨迹 {s_0, a_0, r_1, s_1, ..., s_T}
    for t = 0, ... T do
        if (s_t, a_t) 是 k-次轨迹上首次访问 then
            G_t = r_{t+1} + γ r_{t+2} + ...
            N(s_t, a_t) = N(s_t, a_t) + 1
            Q(s_t, a_t) = Q(s_t, a_t) + 1/N(s_t, a_t) (G_t − Q(s_t, a_t))
        end if
    end for
    k = k + 1, ε = 1/k
    π_k = ε-greedy(Q)
end loop
```

$$Q(s_t, a_t) = Q(s_t, a_t) + \frac{1}{N(s_t, a_t)}\left(G_t - Q(s_t, a_t)\right)$$

定理

GLIE 蒙特卡洛控制是收敛到最优动作 -价值函数,
$Q(s, a) \to Q_*(s, a)$

定理

GLIE 蒙特卡洛控制是收敛到最优动作 -价值函数, $Q(s,a) \to Q_*(s,a)$

- 但实际算法运行时, ϵ 更多是使用一个固定的常值, 保证新观测的数据对更新的有效性

定理

GLIE 蒙特卡洛控制是收敛到最优动作 -价值函数，
$Q(s, a) \to Q_*(s, a)$

- 但实际算法运行时，ϵ 更多是使用一个固定的常值，保证新观测的数据对更新的有效性
- 那么能否和时间差分学习结合呢?

MC vs TD 控制

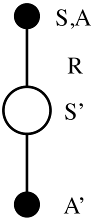
- 时间差分 (TD) 学习和蒙特卡洛 (MC) 方法相比有如下优势
  - 低方差
  - 在线学习
  - 不要求完整的轨迹
- 很自然的想法: 在控制学习中使用 TD 替代 MC
  - 基于 TD 学习 $Q(s, a)$
  - 使用 ϵ-贪心方法进行策略提升
  - 每个时刻都利用观测量对 Q 函数更新

Sarsa

- Q 函数的贝尔曼期望方程

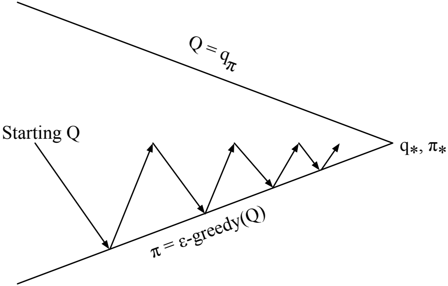
$$Q_\pi(s,a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a \sum_{a' \in \mathcal{A}} \pi(a'|s') Q_\pi(s',a')$$

- 利用样本代表奖励 $\mathcal{R}$ 和模型 $\mathcal{P}$ 函数



- 智能体在线学习时每个时刻都能观察到的一段轨迹 (s,a,r,s',a')
- TD 学习 Q 函数

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



MC 控制

- 每生成一个轨迹:
  1. 策略评估: MC 学习 $Q \simeq Q_\pi$
  2. 策略提升: ϵ-贪心策略提升

Sarsa 控制

- 在每一时刻:
  1. 策略评估: 使用 TD 学习 $Q \approx Q_\pi$
  2. 策略提升: ϵ- 贪心策略提升

Sarsa 算法

初始化 $Q(s,a)$, $\pi = \epsilon\text{-greedy}(Q)$, $t=0$, 初始状态 $s_t = s_0$
采样动作 $a_t \sim \pi(s_t)$
loop
  执行动作 a_t 后获得观测量 (r_{t+1}, s_{t+1})
  采样动作 $a_{t+1} \sim \pi(s_{t+1})$
  根据 $(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1})$ 更新 Q

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

  策略更新

$$\pi = \epsilon\text{-greedy}(Q)$$

  $t = t+1$
end loop

定理

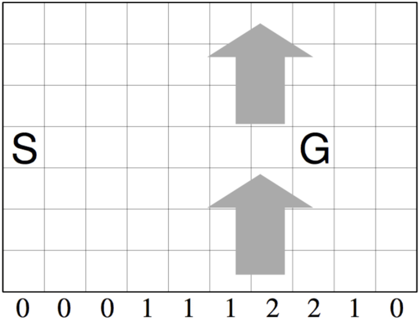
当如下条件件满足时 Sarsa 算法是收敛到最优动作 -价值函数，$Q(s, a) \rightarrow Q_*(s, a)$

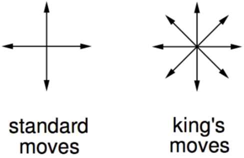
- 策略序列 $\pi_t(a|s)$ 是 GLIE 的
- 更新步长 α_t 满足 Robbins-Monro 序列要求

$$\sum_{t=1}^{\infty} \alpha_t = \infty, \qquad \sum_{t=1}^{\infty} \alpha_t^2 < \infty$$

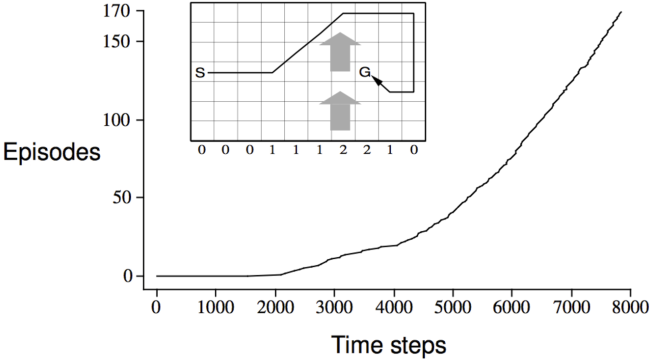
- e.g. $\alpha_t = 1/t$
- 证明参考 [Singh, S., Jaakkola, T., Littman, M. L., & Szepesvári, C. (2000). Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine learning*, 38(3), 287-308.]

举例：有风的迷宫





- 在到达目标点之前每一步都有 $R = -1$
- 无折扣因子，$\gamma = 1$



- 曲线斜率的增加代表智能体每个 episode 到达目标的速度越来越快

- 相比于 MC, Sarsa 每一步都在学习, 很快就会发现比较差的动作, 然后转向其它动作
- MC 则是在一段轨迹结束后 (达到终止状态, 或是轨迹序列太长), 再更新 Q, 效率低

- 考虑 n-步的回报, $n = 1, 2, \infty$:

$$\begin{aligned}
&\color{red}{n = 1} \quad &\color{red}{(Sarsa)} \quad & q_t^{(1)} = r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) \\
&\color{red}{n = 2} & & q_t^{(2)} = r_{t+1} + \gamma r_{t+2} + \gamma^2 Q(s_{t+2}, a_{t+2}) \\
&\vdots & & \\
&\color{red}{n = \infty} & \color{red}{(MC)} \quad & q_t^{(\infty)} = r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{t+T-1} r_T
\end{aligned}$$

- 定义 n-步的 Q-回报

$$q_t^{(n)} = r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^n Q(s_{t+n}, a_{t+n})$$

- n-步 Sarsa 算法更新 $Q(s, a)$, 向 n-步 Q-回报逼近

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\left(\boxed{q_t^{(n)}} - Q(s_t, a_t)\right)$$

- 考虑 n-步的回报, $n = 1, 2, \infty$:

$$\begin{array}{lll} n = 1 & (Sarsa) & q_t^{(1)} = r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) \\ n = 2 & & q_t^{(2)} = r_{t+1} + \gamma r_{t+2} + \gamma^2 Q(s_{t+2}, a_{t+2}) \\ \vdots & & \\ n = \infty & (MC) & q_t^{(\infty)} = r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{t+T-1} r_T \end{array}$$

- 定义 n-步的 Q-回报

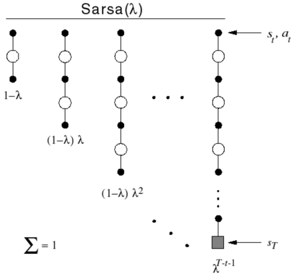
$$q_t^{(n)} = r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^n Q(s_{t+n}, a_{t+n})$$

- n-步 Sarsa 算法更新 $Q(s, a)$, 向 n-步 Q-回报逼近

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\left(q_t^{(n)} - Q(s_t, a_t)\right)$$

- n-步回报的好处是?

前向 Sarsa(λ)



- q^λ 回报包含了所有的 n- 步 Q 回报 $q_t^{(n)}$
- 使用 $(1-\lambda)\lambda^{n-1}$ 对回报加权
  (无穷长轨迹)

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

(轨迹在 T 时刻终止)

$$q_t^\lambda = (1-\lambda)\sum_{n=1}^{T-t-1}\lambda^{n-1}q_t^{(n)} + \lambda^{T-t-1}q_t$$

- 前向 Sarsa(λ) 算法

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\left(q_t^\lambda - Q(s_t, a_t)\right)$$

后向 Sarsa(λ)

- 和 TD(λ) 算法一样，将资格迹引入到在线控制算法中
- 但是 Sarsa(λ) 算法为每个状态 - 动作对都定义资格迹 (vs 之前的 TD(λ) 只定义状态的资格迹)

$$E_0(S, A) = 0$$
$$E_t(S, A) = \gamma\lambda E_{t-1}(S, A) + \mathcal{I}(S = s_t, A = a_t)$$

- 每一时刻对所有的状态和动作的 $Q(s, a)$ 更新
- 更新量与 TD 误差 δ_t 和资格迹 $E_t(S, A)$ 呈正比

$$\delta_t = r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)$$

$$Q(S, A) \leftarrow Q(S, A) + \alpha\delta_t E_t(S, A), \quad \forall S, A$$

Sarsa(λ) 算法

初始化 $Q(s, a)$, $\pi = \epsilon\text{-greedy}(Q)$

repeat {在每个 episode:}

    令 $E(S, A) = 0, \forall S, A$

    初始化 s, 选择 $a \sim \pi(s)$

    repeat {对 episode 的每一步}

        执行 a 后观察 r, s'

        选择动作 $a' \sim \pi(s')$

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

        $E(s, a) \leftarrow E(s, a) + 1$

        for all $S \in \mathcal{S}, A \in \mathcal{A}$ **do**

            $Q(S, A) \leftarrow Q(S, A) + \alpha \delta E(S, A)$

            $E(S, A) \leftarrow \gamma \lambda E(S, A)$

        end for

        $s \leftarrow s'$, $a \leftarrow a'$

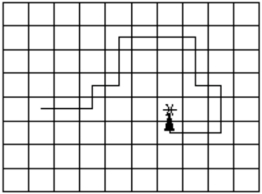
    until s 是终止状态

until

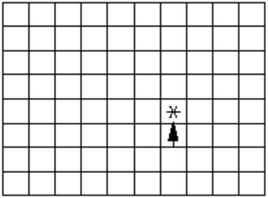
举例: 迷宫问题

- 智能体走迷宫, 到达目标点获得奖励 +1, 其它时刻奖励为 0

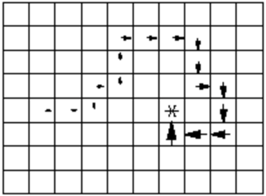
智能体的某一次行走路径



1 步 Sarsa 算法对应的动作价值更新



Sarsa(λ) 在 $\lambda = 0.9$ 时的动作价值更新



- 图 1 是智能体的某次行走轨迹
- 图 2 和图 3 分别展示 1 步 Sarsa 和 Sarsa(λ) 根据轨迹对 (s, a) 的更新
- 箭头代表 $Q(s, a)$ 的更新量

一步 Sarsa vs Sarsa(λ)

- 1 步 Sarsa 只对获得奖励的 最后一步状态 -动作 强化它的 Q 值
- 资格迹方法能够对 整条轨迹上的状态 -动作 强化它们的 Q 值
  - 强化的幅度 (箭头大小) 随着离奖励时刻的距离增加而衰减
  - 衰减率等于 $\gamma\lambda$
  - $0 < \gamma \leq 1,\ 0 < \lambda \leq 1$

重要性采样

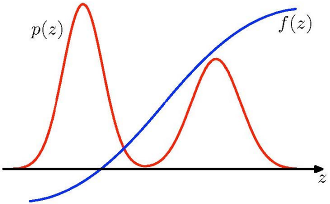
在策略 (on-policy) 和离策略 (off-policy) 学习

- 在策略学习
  - 根据策略 π 产生的样本来学习关于 π 的相关知识
  - e.g. 皇帝通过微服出巡，亲自下凡了解百姓生活 (On-policy)
- 离策略学习
  - 根据另一个策略 μ 产生的样本来学习关于 π 的相关知识
  - e.g. 派巡查官员去了解地方情况，皇帝本人则躺在皇宫里收听百官情报即可 (Off-policy)

- MC/TD 学习中智能体对策略的评估 V_π 使用的是执行策略 π 后产生的数据
- 但是，智能体能否用另一个策略 μ 产生的数据学习 V_π
- 这种学习方式是有意义的:
  - 有时智能体需要观察人类或别的智能体的行为去学习
  - 有时需要重复利用旧策略 $\pi_1, \pi_2, \ldots, \pi_{t-1}$ 产生的经验去学习
  - 执行**探索性的**策略去学习**最优**策略
  - 执行**单一的**策略去学习**多个**策略

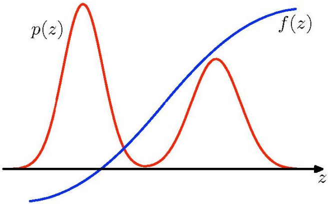
简单的例子

- 变量 z 满足分布 $p(z)$, 定义一个关于 z 的函数 $f(z)$
- 我们想要计算 $p(z)$ 分布下 $f(z)$ 的期望



简单的例子

- 变量 z 满足分布 $p(z)$, 定义一个关于 z 的函数 $f(z)$
- 我们想要计算 $p(z)$ 分布下 $f(z)$ 的期望

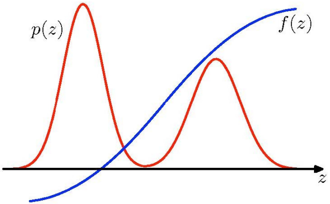


- 从 $p(z)$ 采样的一组独立样本 $\{z^1, \ldots, z^N\}$

$$\mathbb{E}[f] = \int f(z)p(z)\,dz$$
$$\approx \frac{1}{N}\sum_{n=1}^{N} f(z^n) = \hat{f}$$

简单的例子

- 变量 z 满足分布 $p(z)$，定义一个关于 z 的函数 $f(z)$
- 我们想要计算 $p(z)$ 分布下 $f(z)$ 的期望



- 从 $p(z)$ 采样的一组独立样本 $\{z^1, \ldots, z^N\}$

$$\mathbb{E}[f] = \int f(z)p(z)\,dz$$

$$\approx \frac{1}{N}\sum_{n=1}^{N} f(z^n) = \hat{f}$$

- 基于样本的计算是真实结果的无偏估计

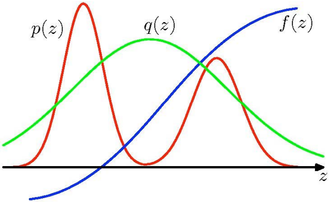
$$\mathbb{E}[f] = \mathbb{E}[\hat{f}]$$

- 方差满足

$$\mathrm{var}[\hat{f}] = \frac{1}{N}\mathbb{E}[(f - \mathbb{E}[f])^2]$$

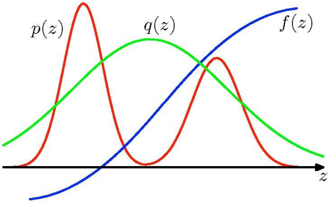
重要性采样 importance sampling

- 假定有另外一组容易采样的样本 $\{z^1, z^2, \ldots, z^N\}$, 满足分布 $q(z)$



重要性采样 importance sampling

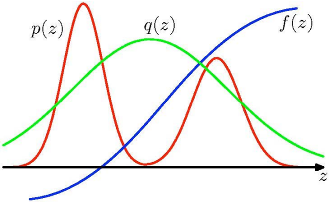
- 假定有另外一组容易采样的样本 $\{z^1, z^2, \ldots, z^N\}$, 满足分布 $q(z)$



$$
\begin{aligned}
\mathbb{E}[f] &= \int f(z)p(z)\,dz \\
&= \int f(z)\frac{p(z)}{q(z)}q(z)\,dz \\
&\approx \frac{1}{N}\sum_{n=1}^{N}\boxed{\frac{p(z^n)}{q(z^n)}}f(z^n), \quad z^n \sim q(z)
\end{aligned}
$$

重要性采样 importance sampling

- 假定有另外一组容易采样的样本 $\{z^1, z^2, \ldots, z^N\}$, 满足分布 $q(z)$



$$
\begin{aligned}
\mathbb{E}[f] &= \int f(z)p(z)\,dz \\
&= \int f(z)\frac{p(z)}{q(z)}q(z)\,dz \\
&\approx \frac{1}{N}\sum_{n=1}^{N}\boxed{\frac{p(z^n)}{q(z^n)}}f(z^n), \quad z^n \sim q(z)
\end{aligned}
$$

- 比值 $w^n = p(z^n)/q(z^n)$ 称为 重要性权重 importance weights
- 要求: $q(z) > 0$ if $p(z) > 0$

另一种形式

- 如果数据的分布是未归一化的，$\tilde{p}(z) = \mathcal{Z}_p p(z)$，$\tilde{q}(z) = \mathcal{Z}_q q(z)$
- 使用 $\tilde{q}(z)$，$\tilde{p}(z)$ 计算 $f(z)$ 的期望

$$\mathbb{E}[f] = \int f(z)p(z)\,dz = \int f(z)\frac{p(z)}{q(z)}q(z)\,dz = \frac{\mathcal{Z}_q}{\mathcal{Z}_p}\int f(z)\frac{\tilde{p}(z)}{\tilde{q}(z)}q(z)\,dz$$

$$\approx \frac{\mathcal{Z}_q}{\mathcal{Z}_p}\frac{1}{N}\sum_n \frac{\tilde{p}(z^n)}{\tilde{q}(z^n)}f(z^n) = \frac{\mathcal{Z}_q}{\mathcal{Z}_p}\frac{1}{N}\sum_n w^n f(z^n)$$

- 同时 $\mathcal{Z}_p/\mathcal{Z}_q$ 也可以用样本估计

$$\frac{\mathcal{Z}_p}{\mathcal{Z}_q} = \frac{1}{\mathcal{Z}_q}\int \tilde{p}(z)\,dz = \int \frac{\tilde{p}(z)}{\tilde{q}(z)}q(z)\,dz \approx \frac{1}{N}\sum_n \frac{\tilde{p}(z^n)}{\tilde{q}(z^n)} = \frac{1}{N}\sum_n w^n$$

- 因此

$$\mathbb{E}[f] \approx \sum_{n=1}^{N} \frac{w^n}{\sum_{m=1}^{N} w^m} f(z^n), \quad z^n \sim q(z)$$

- 智能体在 行为策略 μ 作用下产生一条轨迹
  $\tau = \{s_t, a_t, s_{t+1}, \ldots, s_T | a_{t:T-1} \sim \mu\}$
  和相应的回报 $G_t = r_{t+1} + \gamma r_{t+2} + \ldots$
- 在 μ 下轨迹的概率是

$$p(\tau|\mu) = \mu(a_t|s_t)p(s_{t+1}|s_t, a_t)\mu(a_{t+1}|s_{t+1})\ldots p(s_T|s_{T-1}, a_{T-1})$$

- 在 目标策略 π 作用下智能体产生同一条轨迹的概率是

$$p(\tau|\pi) = \pi(a_t|s_t)p(s_{t+1}|s_t, a_t)\pi(a_{t+1}|s_{t+1})\ldots p(s_T|s_{T-1}, a_{T-1})$$

- 基于重要性采样计算智能体在 π 下的回报

$$G_t^{\pi/\mu} = \frac{p(\tau|\pi)}{p(\tau|\mu)} G_t = \underbrace{\prod_{k=t}^{T-1} \frac{\pi(a_k|s_k)}{\mu(a_k|s_k)}}_{\rho_t} G_t$$

ρ_t 也称为重要性采样比率 importance sampling ratio

基于重要性采样的离策略 MC 方法

- 基于行为策略 μ 的数据，对 π 的 MC 预测学习变成

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

- 但是这种方法要求 $\mu(a|s) > 0, \forall a \in \mathcal{A}$
- 同时使用重要性采样会进一步加大 MC 方法的方差

- 智能体执行策略 μ 生成数据 $(s_t, a_t, r_{t+1}, s_{t+1})$
- π 的 TD 目标的重要性权重等于 $\pi(a_t|s_t)/\mu(a_t|s_t)$
- TD 学习变成

$$V(s_t) \leftarrow V(s_t) + \alpha \left(\frac{\pi(a_t|s_t)}{\mu(a_t|s_t)} (r_{t+1} + \gamma V(s_{t+1})) - V(s_t) \right)$$

- 方差要比基于重要性采样的 MC 方法小很多
- 而且是在线学习

Q-学习

- Sarsa 算法是在策略学习

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

  - 生成数据的行为策略ϵ-greedy(Q)
  - 和评估 Q 的目标策略ϵ-greedy(Q) 是一致的

$$a_{t+1} \sim \epsilon\text{-greedy}(Q)(s_{t+1})$$

$$\begin{aligned} Q_\pi(s,a) =& \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a \sum_{a' \in \mathcal{A}} \pi(a'|s') Q_\pi(s', a') \\ \approx& r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) \end{aligned}$$

- ϵ-贪心策略始终不如完全贪心策略，能否基于离策略方法以贪心策略作为目标策略，评估它的 Q 函数？

Q-学习

- 以贪心策略$\pi(s) = \arg\max_a Q(s,a)$ 的 Q 值作为 TD 目标

$$
\begin{aligned}
Q_\pi(s,a) =& \mathcal{R}_s^a + \gamma \sum_{s'} \mathcal{P}_{ss'}^a \sum_{a'} \pi(a'|s') Q_\pi(s',a') \\
=& \mathcal{R}_s^a + \gamma \sum_{s'} \mathcal{P}_{ss'}^a \max_{a'} Q_\pi(s',a') \\
\approx& r_{t+1} + \gamma \max_{a'} Q(s_{t+1},a')
\end{aligned}
$$

- 对 Q 的更新

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

- 智能体产生数据 $(s_t, a_t, r_{t+1}, s_{t+1})$ 的动作依然是根据ϵ-贪心策略: $a_t \sim \epsilon\text{-greedy}(Q)$
  - 离策略学习: 目标策略 (greedy(Q)) 和行为策略 (ϵ-greedy(Q)) 不同

- 智能体产生数据 $(s_t, a_t, r_{t+1}, s_{t+1})$ 的动作依然是根据ϵ-贪心策略: $a_t \sim \epsilon\text{-greedy}(Q)$
  - 离策略学习: 目标策略 $(\text{greedy}(Q))$ 和行为策略 $(\epsilon\text{-greedy}(Q))$ 不同
- 但更新公式没有使用重要性采样权重:
  - 因为计算的 TD 目标是基于已知的 r_{t+1}, s_{t+1} 和确定的 $a' = \arg\max Q(s_{t+1}, a')$, 没有动作选择的随机过程

- 智能体产生数据 $(s_t, a_t, r_{t+1}, s_{t+1})$ 的动作依然是根据ϵ-贪心策略: $a_t \sim \epsilon\text{-greedy}(Q)$
  - 离策略学习: 目标策略 (greedy(Q)) 和行为策略 (ϵ-greedy(Q)) 不同
- 但更新公式没有使用重要性采样权重:
  - 因为计算的 TD 目标是基于已知的 r_{t+1}, s_{t+1} 和确定的 $a' = \arg\max Q(s_{t+1}, a')$, 没有动作选择的随机过程
- 思考: 基于重要性采样的 Sarsa 算法更新公式会是什么样的?

- 智能体产生数据 $(s_t, a_t, r_{t+1}, s_{t+1})$ 的动作依然是根据ϵ-贪心策略: $a_t \sim \epsilon\text{-greedy}(Q)$
  - 离策略学习: 目标策略 (greedy(Q)) 和行为策略 (ϵ-greedy(Q)) 不同
- 但更新公式没有使用重要性采样权重:
  - 因为计算的 TD 目标是基于已知的 r_{t+1}, s_{t+1} 和确定的 $a' = \arg\max Q(s_{t+1}, a')$, 没有动作选择的随机过程
- 思考: 基于重要性采样的 Sarsa 算法更新公式会是什么样的?

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\left(r_{t+1} + \gamma \frac{\pi(a_{t+1}|s_{t+1})}{\mu(a_{t+1}|s_{t+1})} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right)$$

Q 学习算法

初始化 $Q(s,a)$, $t=0$, 初始状态 $s_t = s_0$
定义策略 $\pi_b = \epsilon\text{-greedy}(Q)$
loop
    采样动作 $a_t \sim \pi_b(s_t)$ 并执行
    获得观测量 r_{t+1}, s_{t+1}
    根据 $(s_t, a_t, r_{t+1}, s_{t+1})$ 更新 Q

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

    更新策略 $\pi_b = \epsilon\text{-greedy}(Q)$
    $t = t + 1$
end loop

收敛条件

- Q-学习中 $Q(s,a)$ 收敛到最优 $Q^*(s,a)$ 的条件?

  所有的状态-动作对 (s,a) 被无穷次遍历 (ϵ-贪心策略)

- Q-学习中的行为策略 π_b 收敛到最优 π^* 的条件?

  除了 $Q(s,a)$ 收敛到 $Q^*(s,a)$ 外, ϵ 还要逐渐衰减到 0

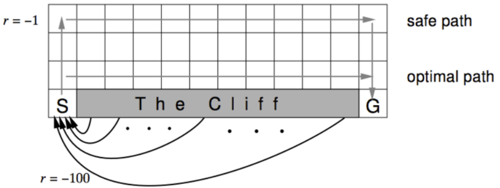
Q 学习和价值迭代

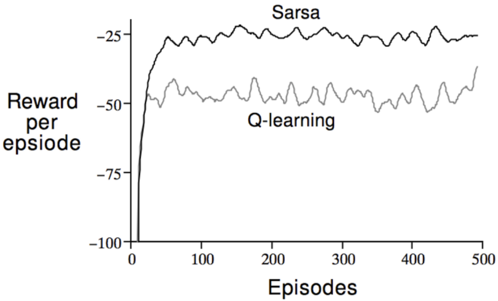
- 也有人认为 Q-学习是价值迭代的在线形式

$$
\begin{aligned}
Q_{k+1}(s,a) = & \mathcal{R}_s^a + \gamma \sum_{s'} \mathcal{P}_{ss'}^a \max_{a'} Q_k(s',a') \qquad \text{(Q 函数的价值迭代)} \\
\approx & r_{t+1} + \gamma \max_{a'} Q(s_{t+1},a') \qquad \text{(基于样本的目标)}
\end{aligned}
$$

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

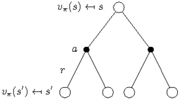
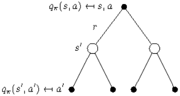
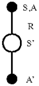
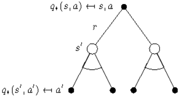
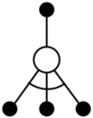
举例: 沿着悬崖行走


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

- Q-学习正确地学到了 最优路径，即沿着悬崖的边缘行走
- 但是由于 ϵ-贪心策略会采取一定概率的随机动作，有时智能体跌入悬崖
- Sarsa 学到的是 安全路径，即沿着网格的最顶端行走
- Sarsa 在学习过程中考虑了随机探索动作的影响
- 即使 Sarsa 学到的安全路径比 Q- 学习的最优路径行走步数要长，但是每个 episode 获得的奖励和却比 Q-学习的高

动态规划和时间差分学习的关系 (1)

	全体后继更新 (DP) Full Backup	样本更新 (TD) Sample Backup
$V_\pi(s)$ 的贝尔曼 期望方程	 迭代策略评估	 TD 学习
$Q_\pi(s,a)$ 的贝尔曼 最优方程	 Q-策略迭代	 Sarsa
$Q_\pi(s,a)$ 的贝尔曼 最优方程	 Q-价值迭代	 Q-学习

动态规划和时间差分学习的关系 (2)

全体后继更新 (DP)	样本更新 (TD)
迭代策略评估 $V(s) \leftarrow \mathbb{E}[R + \gamma V(s') \mid s]$	TD 学习 $V(s) \overset{\alpha}{\leftarrow} r + \gamma V(s')$
Q-策略迭代 $Q(s,a) \leftarrow \mathbb{E}[R + \gamma Q(s',a') \mid s,a]$	Sarsa $Q(s,a) \overset{\alpha}{\leftarrow} r + \gamma Q(s',a')$
Q-价值迭代 $Q(s,a) \leftarrow \mathbb{E}\left[R + \gamma \max_{a' \in \mathcal{A}} Q(s',a') \mid s,a\right]$	Q-学习 $Q(s,a) \overset{\alpha}{\leftarrow} r + \gamma \max_{a' \in \mathcal{A}} Q(s',a')$

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

Q(λ)

- 资格迹描述的是目标策略的轨迹上状态或状态 -动作的强化权重
- Q 学习是离策略方法，ϵ-greedy 行为策略产生的轨迹会有探索动作
- 与目标策略的 greedy 动作不一致
- 因此 Q(λ) 需要额外处理行为策略和目标策略之间的不同

Watkins's Q(λ)

初始化 $Q(S,A)$, $E(S,A)=0, \forall S,A$
repeat {对每个 episode:}
    初始化 s, a
    repeat {对 episode 的每一步:}
        执行动作 a, 观测 r, s'
        根据 Q 提取的策略 (e.g. ϵ-greedy) 对 s' 选择动作 a'
        $a^* \leftarrow \arg\max_b Q(s', b)$
        $\delta \leftarrow r + \gamma Q(s', a^*) - Q(s, a)$
        $E(s,a) \leftarrow E(s,a) + 1$
        for all S, A: **do**
            $Q(S,A) \leftarrow Q(S,A) + \alpha\delta E(S,A)$
            if $a' = a^*$ **then**
                $E(S,A) \leftarrow \gamma\lambda E(S,A)$
            else
                $E(S,A) \leftarrow 0$
            end if
        end for
        $s \leftarrow s', a \leftarrow a'$
    until s 是终止状态
until

- Watkin's Q(λ) 只累积轨迹上连续采用 greedy 动作的资格迹
- 每遇到 ϵ 探索动作或轨迹到达终止状态, 资格迹重新置 0
- 缺点: 由于频繁的探索动作, 资格迹会经常置 0, 起不到信息积累作用
- 其它改进算法: Peng's Q(λ), Naïve Q(λ)
- 但是普遍 Q(λ) 算法不如 Sarsa(λ) 效果明显

Double Q 学习

- 考虑这样的 MDP: 1 个状态 ($|\mathcal{S}| = 1$), 2 个动作, 每个动作获得均值 0 的奖励 ($\mathbb{E}[r|a = a_1] = \mathbb{E}[r|a = a_2] = 0$)
- 那么 $Q(s, a_1) = Q(s, a_2) = 0 = V(s)$

最大化偏差 Maximization Bias

- 考虑这样的 MDP: 1 个状态 ($|\mathcal{S}| = 1$), 2 个动作, 每个动作获得均值 0 的奖励 ($\mathbb{E}[r|a = a_1] = \mathbb{E}[r|a = a_2] = 0$)
- 那么 $Q(s, a_1) = Q(s, a_2) = 0 = V(s)$
- 假设先前有执行动作 a_1 和 a_2 的样本
- 基于这些有限的样本计算 $\hat{Q}(s, a_1), \hat{Q}(s, a_2)$ 作为对 Q 的估计
- 假设我们用了一种无偏的方法估计 Q, e.g. $\hat{Q}(s, a_1) = \frac{1}{n(s,a_1)} \sum_{i=0}^{n(s,a_1)} r_i(s, a_1)$

最大化偏差 Maximization Bias

- 考虑这样的 MDP: 1 个状态 ($|\mathcal{S}| = 1$), 2 个动作, 每个动作获得均值 0 的奖励 ($\mathbb{E}[r|a = a_1] = \mathbb{E}[r|a = a_2] = 0$)
- 那么 $Q(s, a_1) = Q(s, a_2) = 0 = V(s)$
- 假设先前有执行动作 a_1 和 a_2 的样本
- 基于这些有限的样本计算 $\hat{Q}(s, a_1), \hat{Q}(s, a_2)$ 作为对 Q 的估计
- 假设我们用了一种无偏的方法估计 Q, e.g. $\hat{Q}(s, a_1) = \frac{1}{n(s,a_1)} \sum_{i=0}^{n(s,a_1)} r_i(s, a_1)$
- 基于估计的 $\hat{Q}$ 定义贪心策略 $\hat{\pi} = \arg\max_a \hat{Q}(s, a)$
- 尽管每个 $Q(s, a)$ 的估计是无偏的, 但是以此得到的策略 $\hat{\pi}$ 对应的价值的期望 $\hat{V}^{\hat{\pi}}$ 是有偏的

最大化偏差 Maximization Bias

- 考虑这样的 MDP: 1 个状态 ($|\mathcal{S}|=1$), 2 个动作, 每个动作获得均值 0 的奖励 ($\mathbb{E}[r|a=a_1]=\mathbb{E}[r|a=a_2]=0$)
- 那么 $Q(s,a_1)=Q(s,a_2)=0=V(s)$
- 假设先前有执行动作 a_1 和 a_2 的样本
- 基于这些有限的样本计算 $\hat{Q}(s,a_1),\hat{Q}(s,a_2)$ 作为对 Q 的估计
- 假设我们用了一种无偏的方法估计 Q, e.g. $\hat{Q}(s,a_1)=\frac{1}{n(s,a_1)}\sum_{i=0}^{n(s,a_1)} r_i(s,a_1)$
- 基于估计的 $\hat{Q}$ 定义贪心策略 $\hat{\pi}=\arg\max_a \hat{Q}(s,a)$
- 尽管每个 $Q(s,a)$ 的估计是无偏的, 但是以此得到的策略 $\hat{\pi}$ 对应的价值的期望 $\hat{V}^{\hat{\pi}}$ 是有偏的

$$
\begin{aligned}
\hat{V}^{\hat{\pi}} &= \mathbb{E}[\max(\hat{Q}(s,a_1),\hat{Q}(s,a_2))] && (V=\max Q \text{ 的期望}) \\
&\geq \max(\mathbb{E}[\hat{Q}(s,a_1)],\mathbb{E}[\hat{Q}(s,a_2)]) && (\text{Jensen's inequality}) \\
&= \max(0,0)=0=V^*
\end{aligned}
$$

Double learning

- 基于有限样本估计的 Q 函数，生成的贪心策略会导致 最大化偏差
- 偏差产生的原因在于 use *max of estimates* as *estimate of max of true values*
- 解决办法：将样本分成两组，分别定义两个独立的估计 $\hat{Q}_1(s, a_i)$, $\hat{Q}_2(s, a_i)$
  1. 使用一个 Q 函数计算 max 动作：$a^* = \arg\max_a \hat{Q}_1(s, a)$
  2. 使用另一个 Q 函数估计 a^* 的价值：$\hat{Q}_2(s, a^*)$
  3. 获得无偏的估计：$\mathbb{E}[\hat{Q}_2(s, a^*)] = Q(s, a^*)$
- 为什么是无偏估计？
  - 假定在第 1 步中选择 $a^* = a_1$ 的概率是 $p(a_1)$，选择 $a^* = a_2$ 的概率是 $p(a_2)$，并且 $p(a_1) + p(a_2) = 1$
  - $\hat{Q}_2(s, a)$ 是真实 $Q(s, a)$ 的无偏估计

$$\begin{aligned}\mathbb{E}[\hat{Q}_2(s, a^*)] =& \mathbb{E}[p(a_1)\hat{Q}_2(s, a_1) + p(a_2)\hat{Q}_2(s, a_2)] \\ =& p(a_1)Q(s, a_1) + p(a_2)Q(s, a_2) = 0 = V^*\end{aligned}$$

Double Q-Learning

初始化 $Q_1(s,a)$, $Q_2(s,a)$, 初始化状态 $s_t = s_0$

loop

  在 $\pi(s) = \arg\max_a \big(Q_1(s,a) + Q_2(s,a)\big)$ 基础上选择 ϵ-greedy 动作 a_t

  获得观测量 r_{t+1}, s_{t+1}

  if with 0.5 probability **then**

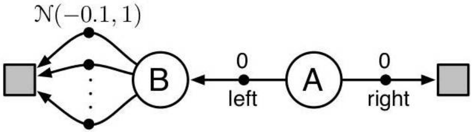
$$Q_1(s_t,a_t) \leftarrow Q_1(s_t,a_t) + \alpha\Big(r_{t+1} + \gamma Q_1\big(s_{t+1}, \arg\max_{a'} Q_2(s_{t+1},a')\big) - Q_1(s_t,a_t)\Big)$$

  else

$$Q_2(s_t,a_t) \leftarrow Q_2(s_t,a_t) + \alpha\Big(r_{t+1} + \gamma Q_2\big(s_{t+1}, \arg\max_{a'} Q_1(s_{t+1},a')\big) - Q_2(s_t,a_t)\Big)$$

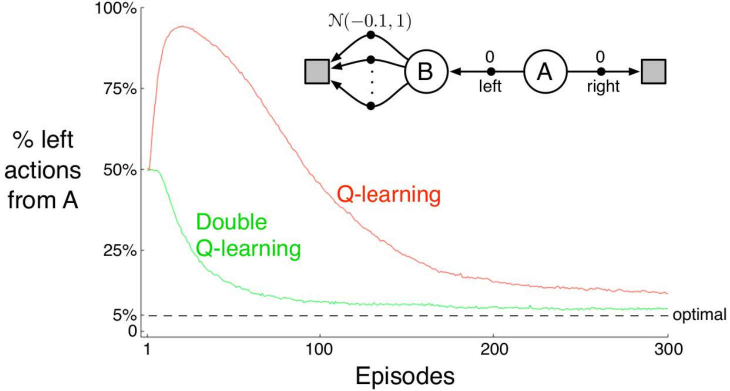
  end if

end loop



–Sutton, R.S., & Barto, A.G. (2011). Reinforcement Learning: An Introduction.

The MDP has two non-terminal states A and B. Episodes always start in A with a choice between two actions, left and right. The right action transitions immediately to the terminal state with a reward and return of zero. The left action transitions to B, also with a reward of zero, from which there are many possible actions all of which cause immediate termination with a reward drawn from a normal distribution with mean -0.1 and variance 1.0. Thus, the expected return for any trajectory starting with left is-0.1, and thus taking left in state A is always a mistake.



Nevertheless, Q-learning method may favor left because of maximization bias making B appear to have a positive value. Figure 6.7 shows that Q-learning with ϵ-greedy action selection initially learns to strongly favor the left action on this example. Even at asymptote, Q-learning takes the left action about 5% more often than is optimal at our parameter settings ($\epsilon = 0.1$, $\alpha = 0.1$, and $\gamma = 1$).

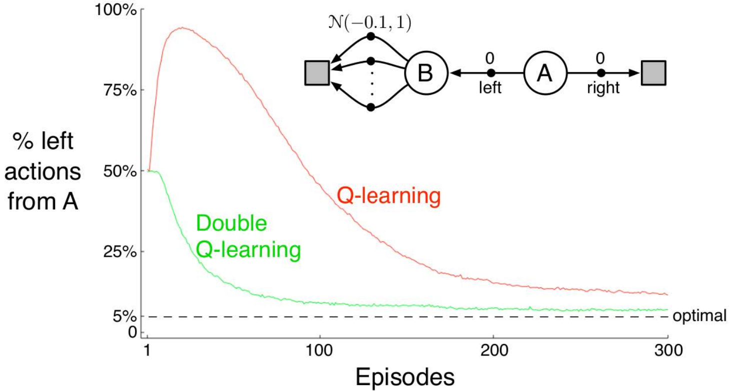


Figure 6.7: Comparison of Q-learning and Double Q-learning on a simple episodic MDP (shown inset). Q-learning initially learns to take the left action much more often than the right action, and always takes it significantly more often than the 5% minimum probability enforced by ϵ-greedy action selection with ϵ= 0.1. In contrast, Double Q-learning is essentially unaffected by maximization bias. These data are averaged over 10,000 runs. The initial action-value estimates were zero. Any ties in ϵ-greedy action selection were broken randomly.

探索与利用

探索 (exploration) 与利用 (exploitation)

- 在线决策过程存在的一个关键问题是:

  利用: 根据当前的信息做出最佳的决策

  探索: 采样更多的信息

- 想要做出长期的最佳决策有时需要牺牲当前短期的部分利益
- 收集更多的信息从而做出整体最佳的决策

举例

- 选择餐馆
  - 利用：去你平时最喜欢去的餐馆
  - 探索：尝试一个之前没去过的
- 网站植入广告
  - 利用：植入当前收益最大的广告
  - 探索：尝试一个不同的广告
- 石油钻井
  - 利用：沿着当前已知的最好的点向下挖
  - 探索：在一个新的点钻井
- 玩游戏
  - 利用：使用你认为是最优的打法
  - 探索：尝试一种新打法

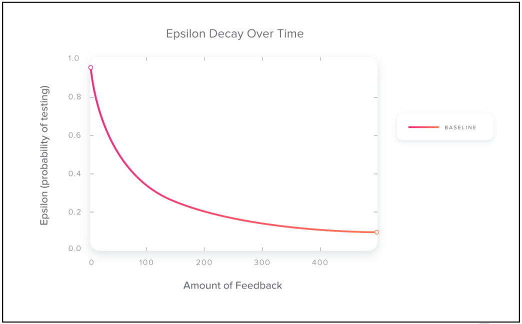
- 如果 Q-学习没有探索，即 $\epsilon = 0$，学习结果容易陷入局部最优解，无法找到真正最优策略
- 如果 ϵ 很大也不好
  1. 状态，动作空间很大时，想要探索整个空间是费时费力的，对在线学习过程不利
  2. 随机的动作会降低智能体的实际回报

衰减 ϵ-探索

$$\pi_t(s) = \begin{cases} \arg\max\limits_{a \in \mathcal{A}} Q_t(s,a), & \text{with } 1-\epsilon_t \text{ probability} \\ \text{random} a \in \mathcal{A}, & \text{with } \epsilon_t \text{ probability} \end{cases}$$

- ϵ-贪心策略的探索率随时间衰减

$$\epsilon_t = \max(\epsilon_0 \cdot d^t, \epsilon_{\min})$$



非均匀探索概率

- ϵ-贪心策略除了最优动作，其它动作 以等概率随机选择
- 假如：有两个动作看起来不错，而其它的动作完全不可行
- 那么比较合理的探索方式是选择那些看起来不错的动作
  - 从不错的动作中选出最优的动作
  - 而不是将精力浪费在看起来不可行的动作
- 根据动作 -价值决定动作被选中的概率

波尔兹曼探索, Boltzmann exploration

$$\pi(a|s) = \frac{e^{Q(s,a)/T}}{\sum_{a' \in \mathcal{A}} e^{Q(s,a')/T}}$$

- a 被选中的概率与 $e^{Q(s,a)/T}$ 呈正比 ($Q(s,a) \geq 0, \forall s, a$)
- 温度系数 $T > 0$ 决定了策略的随机性能
  - 如果 T 很大, 所有动作几乎以等概率选择 (探索)
  - 如果 T 很小, 高价值的动作更容易被选中 (利用)
  - 极限情况 $T \to 0$, 只选择最优动作

总结

蒙特卡洛控制

Sarsa

重要性采样

Q-学习

Double Q 学习

探索与利用