高级人工智能

沈华伟

中国科学院计算技术研究所

2022.12.1

大纲

- 强化学习
- 多臂赌博机
- 马尔科夫决策过程

强化学习

The idea that we **learn by interacting with our environment** is probably the first to occur to us when we think about the nature of learning.

Learning from interaction is a foundational idea underlying nearly all theories of learning and intelligence.

Reinforcement learning is a **computational approach** to learning from interaction, which is a kind of **goal-directed** learning.

Richard S. Sutton and Andrew G. Barto. Reinforcement Learning: An Introduction (Second edition). The MIT Press, Cambridge, Massachusetts

强化学习

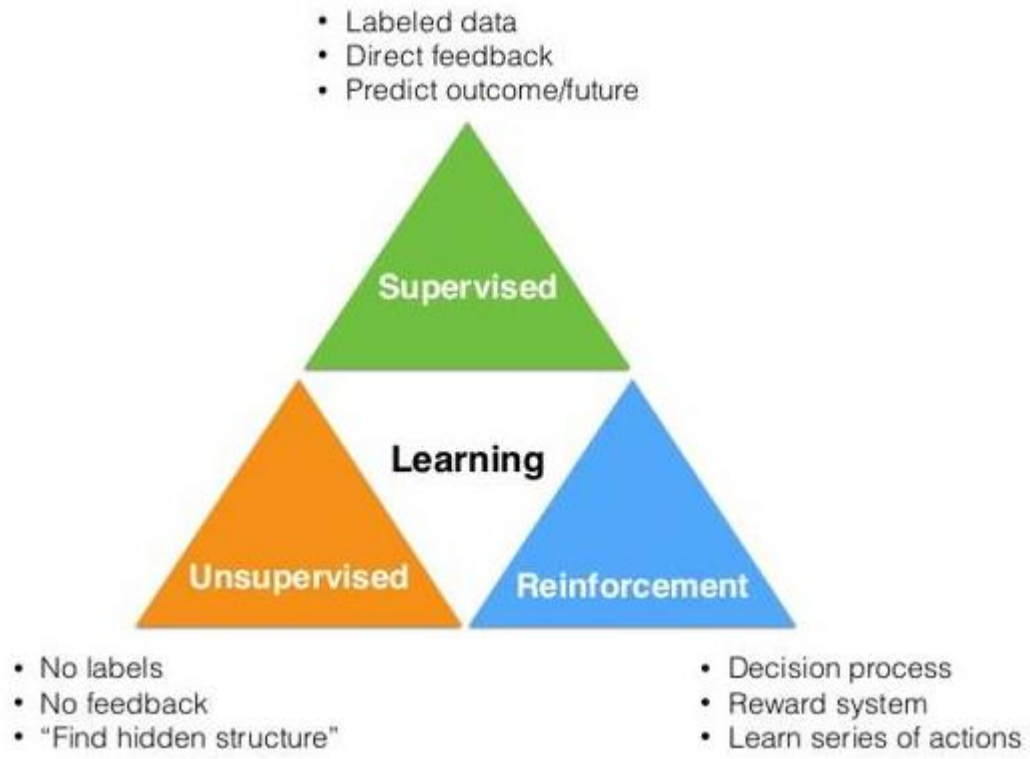
- 强化学习：Reinforcement learning
  - 目标：学习从环境状态到行为的映射(即策略），智能体选择能够获得环境最大奖赏的行为，使得外部环境对学习系统在某种意义下的评价为最佳

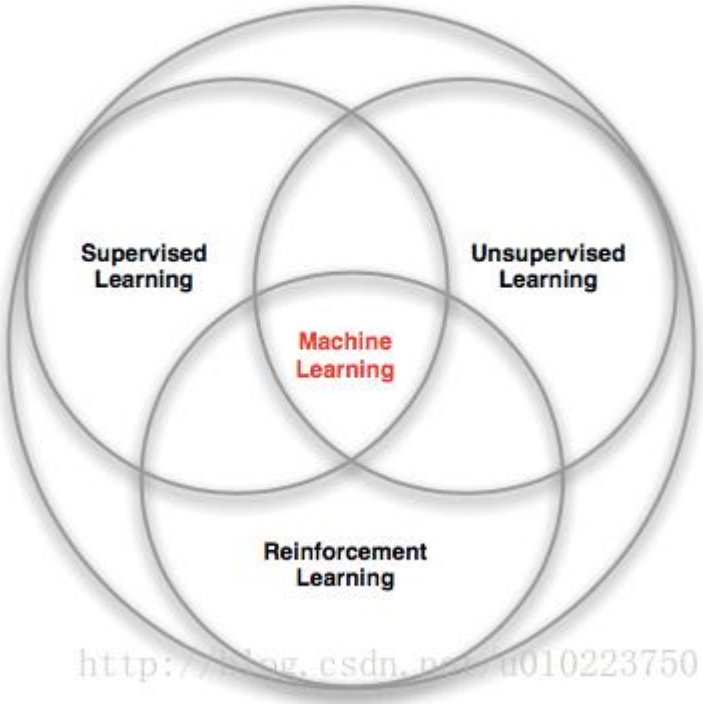
- 区别于监督学习
  - 监督学习是从标注中学习
    - Learning from a training set of labeled examples provided by a knowledgeable external supervisor
    - Focusing on generalization capacity
  - 强化学习是从交互中学习
    - Learning from interactions with environment

两种反馈

- 评价性反馈：Evaluative feedback
  - 当智能体采取某个行为时，对该行为给出一个评价，但并不知道哪个行为是最好的
  - 强化学习经常面临的是评价性反馈
- 指导性反馈：Instructive feedback
  - 直接给出某个状态下的正确或最好行为
  - 独立于智能体当前采取的行为
  - 监督学习使用的是指导性反馈

强化学习 vs. 监督学习 vs. 无监督学习





三者目标和手段均不同，又可以相互结合和促进

强化学习的两大特性

- 试错搜索
  - Trial-and-error search
- 延迟奖励
  - Delayed reward

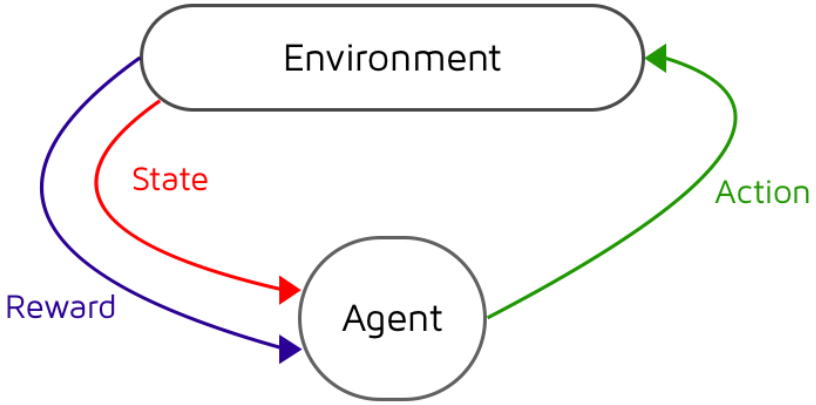
用于判断某一问题是否适用于强化学习求解

强化学习需要应对的挑战

- Exploration-exploitation dilemma
  - Exploitation：利用/开采
    - Agent exploits what it has already experienced in order to obtain reward
  - Exploration：探索/勘探
    - Agent explores in order to make better action selections in the future
- Focusing on the target problem as a whole rather than focusing on many isolated sub-problems
  - Generally start with a complete, interactive, goal-seeking agent
  - 例子：下棋的目标是胜负，而不是尽可能多地吃掉对方的棋子

强化学习的要素

- 主体：智能体和环境
  - 状态、行为和奖励



- 要素
  - 策略：policy
    - 状态到行为的映射，包括确定策略和随机策略两种
  - 奖励：reward
    - 关于状态和行为的函数，通常具有一定的不确定性
  - 价值：value
    - 累积奖励或长期目标
  - 环境模型：model of environment
    - 刻画环境对行为的反馈

强化学习发展历程简要回顾（1/2）

- 1911年，Thorndike提出效果律（Law of effect），从心理学的角度探讨了强化思想：动物感到舒服的行为会被强化，动物感到不舒服的行为会被弱化
- 1954年，马文·明斯基（Marvin Minsky）在其博士论文中实现了计算上的试错学习
- 1957年，Bellman提出求解最优控制问题的动态规划方法，并提出了最优控制问题的随机离散版本，即著名的马尔科夫决策过程
- 1960年，Howard提出马尔科夫决策过程的策略迭代方法
- 1961年，明斯基在其论文“Steps toward artificial intelligence”中首次使用“Reinforcement learning”一词

*1969年，明斯基因在人工智能领域的贡献获得图灵奖

强化学习发展历程简要回顾（2/2）

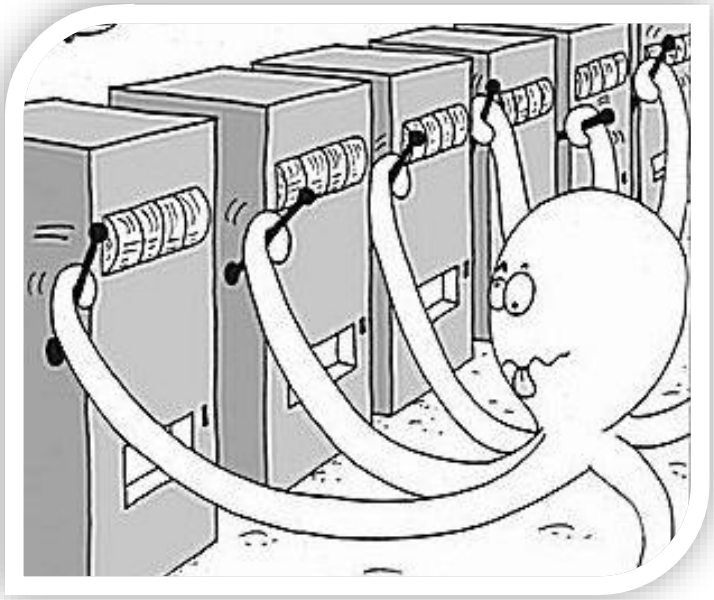
- 1989年，Watkins提出了Q-learning，将动态规划、时序差分、蒙特卡洛模拟三条线结合在了一起
- 1992年，Tesauro将强化学习成功应用到西洋双陆棋
- ……
- 2015年，强化学习和深度学习结合：AlphaGo
- 2017年，AlphaGo Zero
- ……

大纲

- 强化学习
- 多臂赌博机
- 马尔科夫决策过程

多臂赌博机 (Multi-armed bandit)

- 一台赌博机有多个摇臂，每个摇臂摇出的奖励（reward）大小不确定，玩家希望摇固定次数的臂所获得的期望累积奖励最大



问题形式化

- 行为：摇哪个臂
- 奖励：每次摇臂获得的奖金
- A_t表示第t轮的行为，R_t表示第t轮获得的奖励
- 第t轮采取行为a的期望奖励为：

$$q_*(a) \doteq \mathbb{E}[R_t | A_t = a]$$

Question:

假如摇臂T次，那么按照什么策略摇臂，才能使期望累积奖励最大呢？

Answer:

当$q_*(a)$已知时，每次都选择$q_*(a)$最大的a（贪心策略）

贪心策略

- 一般情况下， $q_*(a)$ 对于玩家而言是未知的或具有不确定性，此时该采用什么样的策略呢？
- 玩家在第t轮时只能依赖于当时对$q_*(a)$的估值$Q_t(a)$进行选择
- 此时，贪心策略在第t轮选择$Q_t(a)$最大的a

在$q_*(a)$未知时，贪心策略还是最好的策略吗？

利用和探索

- 利用：Exploitation
  - 按照贪心策略进行选择，即选择$Q_t(a)$最大的行为a
  - 优点：最大化即时奖励
  - 缺点：由于$Q_t(a)$只是对$q_*(a)$的估计，估计的不确定性导致按照贪心策略选择的行为不一定是$q_*(a)$最大的行为

- 探索：Exploration
  - 选择贪心策略之外的行为（non-greedy actions）
  - 缺点：短期奖励会比较低
  - 优点：长期奖励会比较高，通过探索可以找出奖励更大的行为，供后续选择

✓ 每步选择在“利用”和“探索”中二选一

✓ 如何平衡“利用” 和“探索” 是关键

贪心策略和ε贪心策略

- 贪心策略形式化地表示为

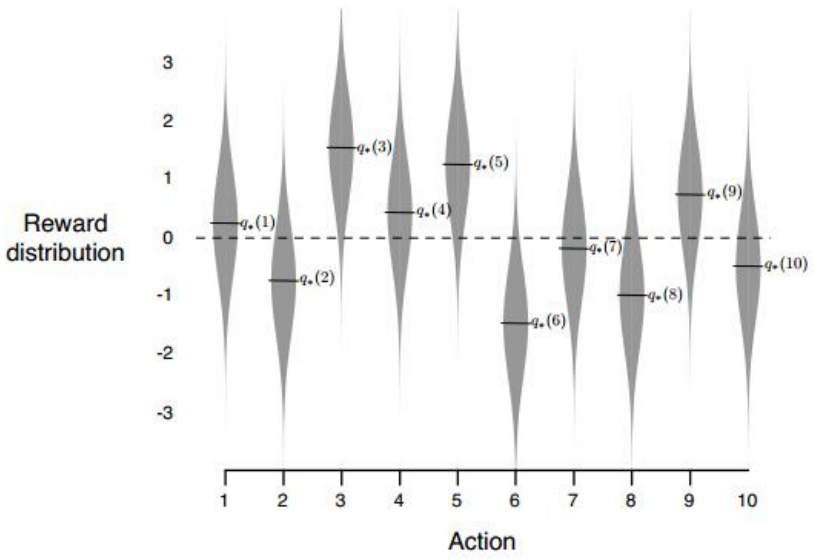
$$A_t \doteq \arg\max_a Q_t(a)$$

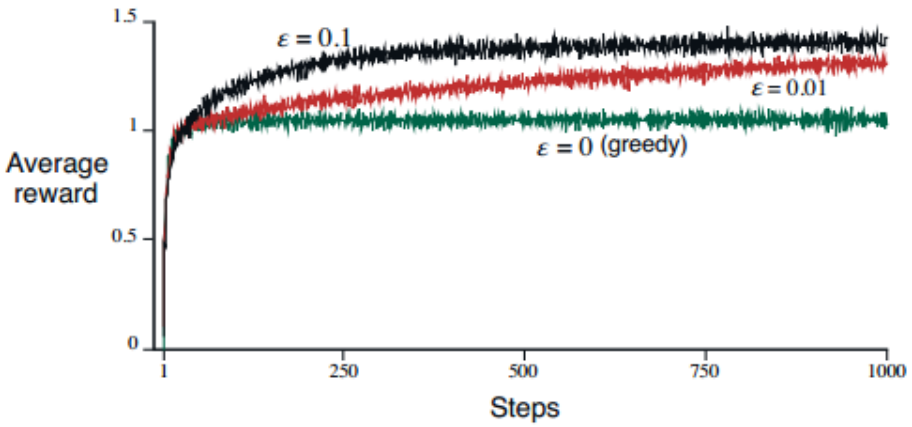
当有多个行为的$Q_t(a)$同时为最大时，随机选择一个

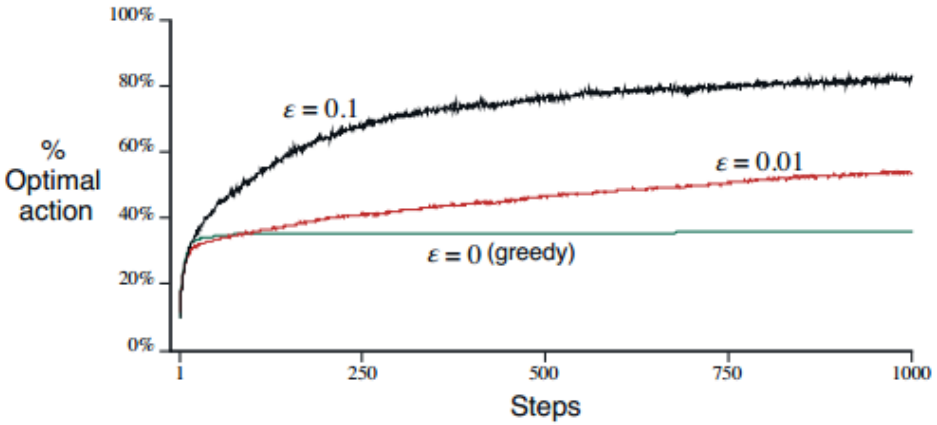
- ε贪心策略
  - 以概率$1-\varepsilon$按照贪心策略进行行为选择——Exploitation
  - 以概率ε在所有行为中随机选择一个——Exploration
  - ε的取值取决于$q_*(a)$的方差，方差越大ε取值应越大

例子

- 假设有10个臂，每个臂的期望奖励$q_*(a)$从一个均值为0、方差为1的正态分布中采样得到
- 在第t轮选择行为A_t所得到的奖励服从均值为$q_*(A_t)$、方差为1的正态分布







行为估值方法

- 根据历史观测样本的均值对$q_*(a)$进行估计

$$Q_t(a) \doteq \frac{\text{sum of rewards when } a \text{ taken prior to } t}{\text{number of times } a \text{ taken prior to } t} = \frac{\sum_{i=1}^{t-1} R_i \cdot \mathbb{1}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{1}_{A_i=a}}$$

✓ 约定：当分母等于0时，$Q_t(a)$=0

✓ 当分母趋于无穷大时，$Q_t(a)$收敛到$q_*(a)$

行为估值的增量式实现

- 行为估值时，一个行为被选择了n次后的估值记为

$$Q_n \doteq \frac{R_1 + R_2 + \cdots + R_{n-1}}{n-1}$$

该估值方式需要记录$n-1$个奖励值

- 增量式实现

$$\begin{aligned}
Q_{n+1} &= \frac{1}{n}\sum_{i=1}^{n} R_i \\
&= \frac{1}{n}\left(R_n + \sum_{i=1}^{n-1} R_i\right) \\
&= \frac{1}{n}\left(R_n + (n-1)\frac{1}{n-1}\sum_{i=1}^{n-1} R_i\right) \\
&= \frac{1}{n}\Big(R_n + (n-1)Q_n\Big) \\
&= \frac{1}{n}\Big(R_n + nQ_n - Q_n\Big) \\
&= Q_n + \frac{1}{n}\Big[R_n - Q_n\Big],
\end{aligned}$$

✓ 只需要记录Q_n和n两个值

贪心策略的算法

- 伪代码

A simple bandit algorithm

Initialize, for $a = 1$ to k:
$Q(a) \leftarrow 0$
$N(a) \leftarrow 0$

Repeat forever:
$A \leftarrow \begin{cases} \arg\max_a Q(a) & \text{with probability } 1-\varepsilon \quad \text{(breaking ties randomly)} \\ \text{a random action} & \text{with probability } \varepsilon \end{cases}$
$R \leftarrow bandit(A)$
$N(A) \leftarrow N(A) + 1$
$Q(A) \leftarrow Q(A) + \frac{1}{N(A)}\left[R - Q(A)\right]$

bandit(A)：采取行动*A*，返回这次行动获得的奖励

更具一般性的行为估值

- 更新公式

$$NewEstimate \leftarrow OldEstimate + StepSize \left[Target - OldEstimate\right]$$

Target – OldEstimate 表示估计误差

- 对于贪心策略的增量实现而言
  - 步长（StepSize）: 1/n

- 更一般的情况
  - 步长用参数α或$\alpha_t(a)$表示

非平稳问题

- 平稳问题
  - $q_*(a)$是稳定的，不随时间而变化
  - 随着观测样本的增加，平均值估计方法最终收敛于$q_*(a)$

- 非平稳问题
  - $q_*(a)$是关于时间的函数
  - 对$q_*(a)$的估计需要更关注最近的观测样本

非平稳情形下的行为估值

- 行为估值的更新公式

$$Q_{n+1} \doteq Q_n + \alpha\Big[R_n - Q_n\Big]$$

- 递推得到

$$\begin{aligned}
Q_{n+1} &= Q_n + \alpha\Big[R_n - Q_n\Big] \\
&= \alpha R_n + (1-\alpha)Q_n \\
&= \alpha R_n + (1-\alpha)\left[\alpha R_{n-1} + (1-\alpha)Q_{n-1}\right] \\
&= \alpha R_n + (1-\alpha)\alpha R_{n-1} + (1-\alpha)^2 Q_{n-1} \\
&= \alpha R_n + (1-\alpha)\alpha R_{n-1} + (1-\alpha)^2\alpha R_{n-2} + \\
&\qquad \cdots + (1-\alpha)^{n-1}\alpha R_1 + (1-\alpha)^n Q_1 \\
&= (1-\alpha)^n Q_1 + \sum_{i=1}^{n} \alpha(1-\alpha)^{n-i} R_i .
\end{aligned}$$

对应于指数“新近”带权平均

更新步长的选择

- 并不是所有的步长选择$\{\alpha_n(a)\}$都保证收敛
  - $\alpha_n(a) = \frac{1}{n}$收敛
  - $\alpha_n(a) = \alpha$不收敛

- 收敛条件

$$\sum_{n=1}^{\infty} \alpha_n(a) = \infty \qquad \text{and} \qquad \sum_{n=1}^{\infty} \alpha_n^2(a) < \infty$$

  - 第一个条件保证步长足够大，克服初值或随机扰动的影响
  - 第二个条件保证步长最终会越来越小，小到保证收敛

行为选择策略

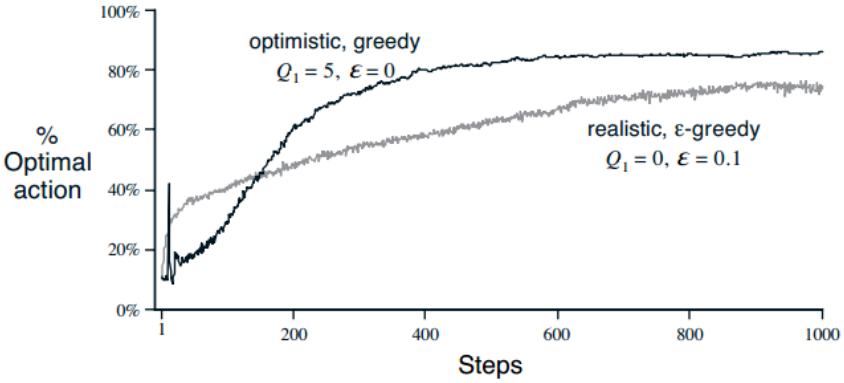
- 如何制定合适的行为选择策略？
  - 贪心策略：选择当前估值最好的行为
  - ε贪心策略：以一定的概率随机选择非贪心行为（non-greedy actions），但是对于非贪心行为不加区分

- 行为选择策略
  - 平衡exploitation和exploration，应对行为估值的不确定性
  - 关键：确定每个行为被选择的概率

乐观初值法

- 行为的初始估值
  - 前述贪心策略中，每个行为的初始估值为0
  - 每个行为的初始估值可以帮助我们引入先验知识
  - 初始估值还可以帮助我们平衡exploitation和exploration

- 乐观初值法：Optimistic Initial Values
  - 为每个行为赋一个高的初始估值
  - 好处：初期每个行为都有较大机会被explore



UCB行为选择策略

- UCB：Upper-Confidence-Bound

$$A_t \doteq \arg\max_a \left[Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}} \right]$$

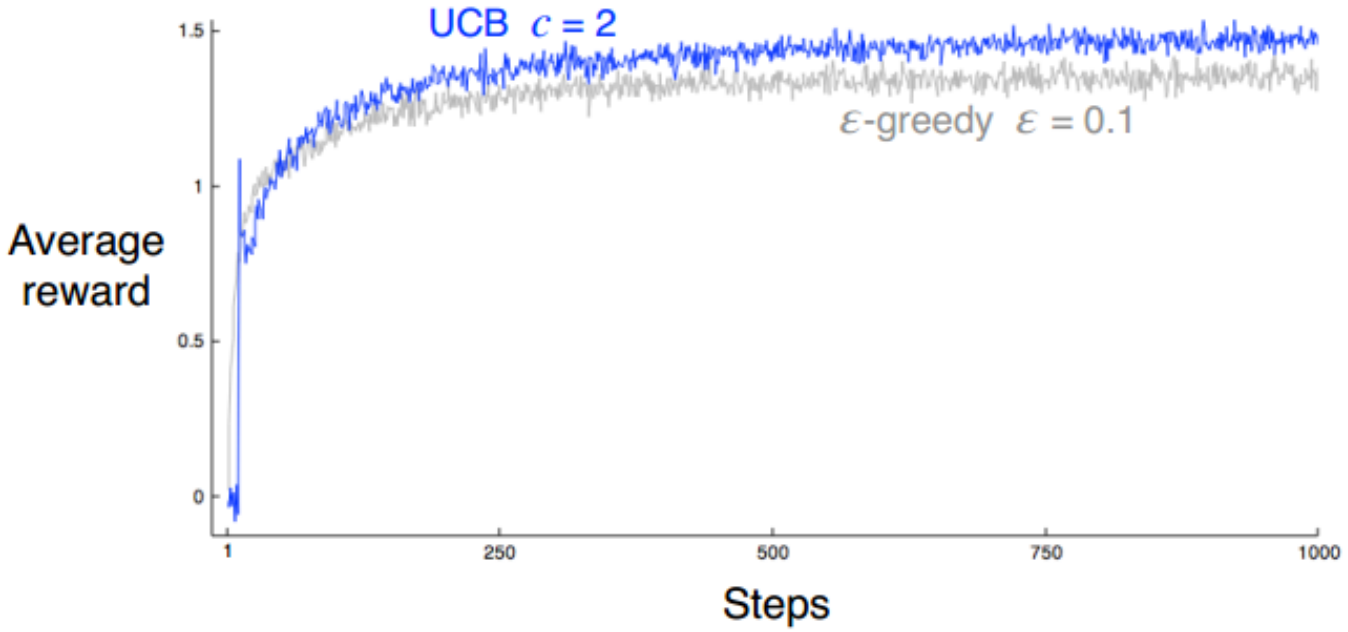
$N_t(a)$表示时刻t之前行为a被选择的次数

公式解读

- ✓ 选择潜力大的行为：依据估值的置信上界进行行为选择
- ✓ 第一项表示当前估值要高，e.g., 接近greedy action
- ✓ 第二项表示不确定性要高，e.g., 被选择的次数少
- ✓ 参数c用来控制exploration的程度

UCB策略 vs. ε贪心策略

UCB策略一般会优于ε贪心策略，不过最初几轮相对较差



UCB策略实现起来比ε贪心策略要复杂，在多臂赌博机之外的强化学习场景中使用较少

梯度赌博机算法

- 和前面的确定策略不同，梯度赌博机是一种随机策略
  - 使用$H_t(a)$表示在第t轮对行为a的偏好程度
  - 根据行为选择后获得的奖励大小更新$H_t(a)$
- 在第t轮选择行为a的概率为

$$\Pr\{A_t = a\} \doteq \frac{e^{H_t(a)}}{\sum_{b=1}^{k} e^{H_t(b)}} \doteq \pi_t(a)$$

- 更新公式

$$H_{t+1}(A_t) \doteq H_t(A_t) + \alpha\big(R_t - \bar{R}_t\big)\big(1 - \pi_t(A_t)\big), \qquad \text{and}$$
$$H_{t+1}(a) \doteq H_t(a) - \alpha\big(R_t - \bar{R}_t\big)\pi_t(a), \qquad \text{for all } a \neq A_t$$

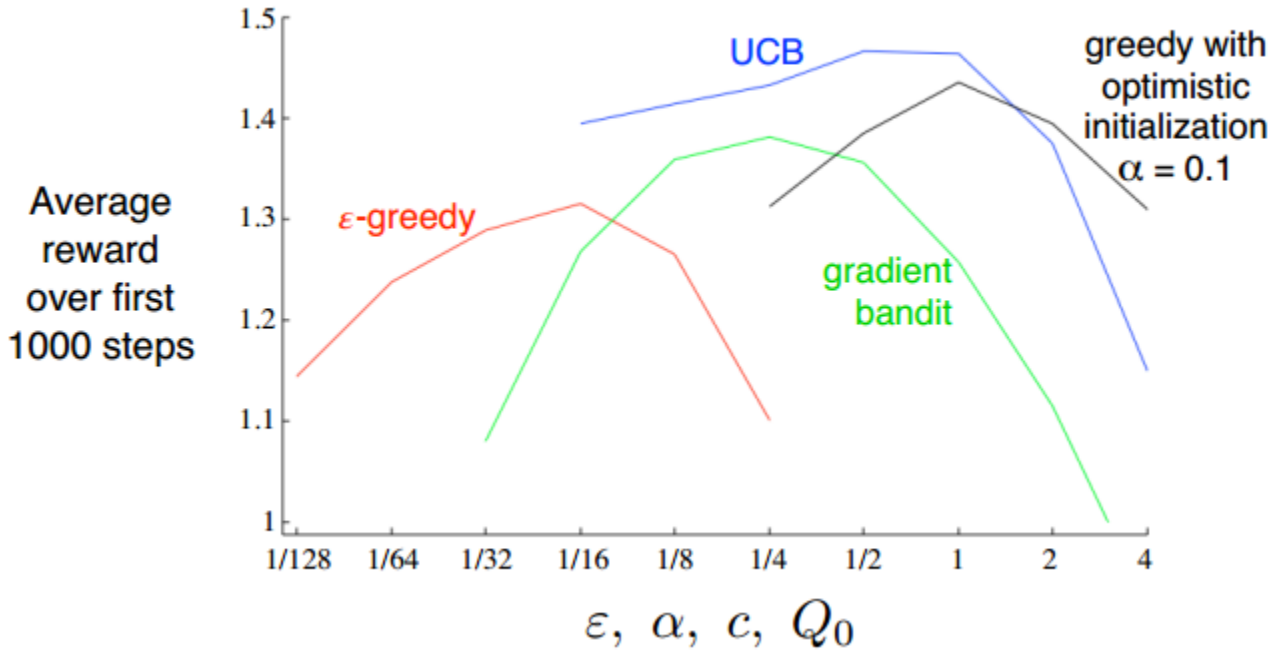
  等价于随机梯度上升

- 优化目标：第t轮的期望奖励大小

$$\mathbb{E}[R_t] = \sum_b \pi_t(b) q_*(b)$$

不同策略的对比

- UCB策略表现相对较好



多臂赌博机的应用案例

- 手机客户端的新闻（或广告）投放
  - 行为a：投放新闻a
  - 新闻a的被点击的可能性记为$q_*(a)$
  - 目标：使得多次投放新闻累计获得的点击数最高

小结

- 多臂赌博机是强化学习的一个简化场景
  - 行为和状态之间没有关联关系
- 扩展情形
  - 有上下文的多臂赌博机（Contextual bandit）
    - 存在多个多臂赌博机，状态表示赌博机
    - 学习状态到行为的映射
    - 但行为不改变状态
- 更一般的情形
  - 马尔科夫决策过程

课间休息

大纲

- 强化学习
- 多臂赌博机
- 马尔科夫决策过程

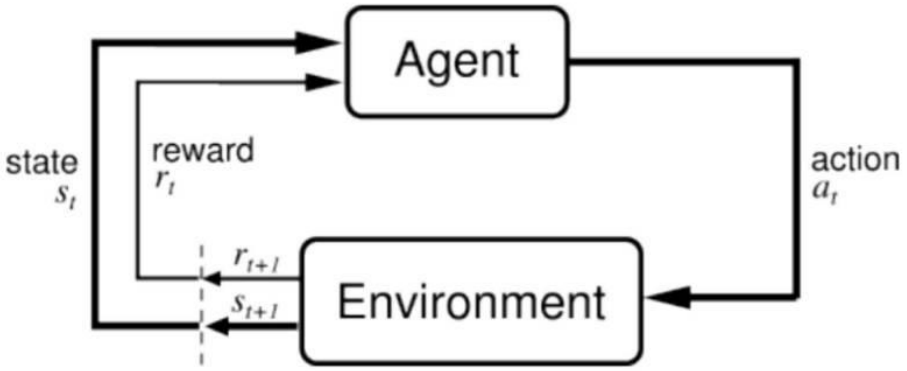
马尔科夫决策过程

- Markov Decision Process: MDP
- 常用于建模序列化决策过程
- 行为不仅获得即时奖励，还能改变状态，从而影响长期奖励
- 学习状态到行为的映射——策略
  - 多臂赌博机学习$q_*(a)$
  - MDP学习$q_*(s,a)$或$\boldsymbol{v}_*(s)$

马尔科夫决策过程的要素

- 智能体和环境按照离散的时间步进行交互

  $t = 0,1,2,3,\cdots$



- 形式化记号
  - 智能体在时间步t所处的状态记为$S_t \in S$
  - 智能体在时间步t所采取的行为记为$A_t \in A(s)$
  - 采取行为A_t后智能体转到状态S_{t+1}，并获得奖励$R_{t+1} \in R$
  - 马尔科夫决策过程产生的序列记为

$$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \ldots$$

(有限)马尔科夫决策过程的建模

- 模型

$$p(s', r \mid s, a) \doteq \Pr\{S_t = s', R_t = r \mid S_{t-1} = s, A_{t-1} = a\}$$

这里 $p : \mathcal{S} \times \mathcal{R} \times \mathcal{S} \times \mathcal{A} \to [0, 1]$ 满足

$$\sum_{s' \in \mathcal{S}} \sum_{r \in \mathcal{R}} p(s', r \mid s, a) = 1, \text{ for all } s \in \mathcal{S}, a \in \mathcal{A}(s)$$

一些常见推导

- 状态转移概率

$$p(s'|s,a) \doteq \Pr\{S_t=s' \mid S_{t-1}=s, A_{t-1}=a\} = \sum_{r\in\mathcal{R}} p(s',r|s,a)$$

- “状态-行为”对的期望奖励

$$r(s,a) \doteq \mathbb{E}[R_t \mid S_{t-1}=s, A_{t-1}=a] = \sum_{r\in\mathcal{R}} r \sum_{s'\in\mathcal{S}} p(s',r|s,a)$$

- “状态-行为-下一状态”的奖励

$$r(s,a,s') \doteq \mathbb{E}[R_t \mid S_{t-1}=s, A_{t-1}=a, S_t=s'] = \sum_{r\in\mathcal{R}} r \frac{p(s',r|s,a)}{p(s'|s,a)}$$

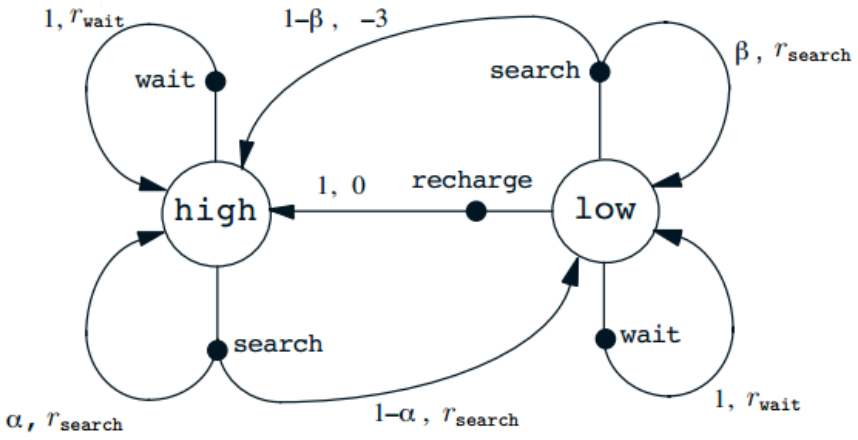
MDP例子

- 垃圾回收机器人
  - 状态（电压高低）：$\mathcal{S} = \{\texttt{high}, \texttt{low}\}$
  - 行为：寻找垃圾、等待、充电

$$\begin{aligned} \mathcal{A}(\texttt{high}) &\doteq \{\texttt{search}, \texttt{wait}\} \\ \mathcal{A}(\texttt{low}) &\doteq \{\texttt{search}, \texttt{wait}, \texttt{recharge}\} \end{aligned}$$

  - 模型：

s	a	s'	$p(s'\|s,a)$	$r(s,a,s')$
high	search	high	α	$r_{\texttt{search}}$
high	search	low	$1-\alpha$	$r_{\texttt{search}}$
low	search	high	$1-\beta$	-3
low	search	low	β	$r_{\texttt{search}}$
high	wait	high	1	$r_{\texttt{wait}}$
high	wait	low	0	$r_{\texttt{wait}}$
low	wait	high	0	$r_{\texttt{wait}}$
low	wait	low	1	$r_{\texttt{wait}}$
low	recharge	high	1	0
low	recharge	low	0	0.



奖励假设

- 目标和奖励
  - 目标：长期的或最终的
  - 奖励：即时的
- 奖励假设：reward hypothesis

> That all of what we mean by goals and purposes can be well thought of as the maximization of the expected value of the cumulative sum of a received scalar signal (called reward)

- 如何设置奖励呢？
  - 奖励是我们与智能体进行交互的途径

奖励设置

- 设置奖励是希望智能体能达到我们期望的目标
  - 下围棋
    - 目标：赢棋
    - 奖励需要是能够实现赢棋这一目标才合适
      - 吃子多少？占领棋盘的中心？
  - 迷宫
    - 目标：尽快走出去
    - 奖励：每走一步，奖励为-1（相当于惩罚）
  - 垃圾回收机器人
    - 目标：尽可能少的人工干预的情况下回收尽可能多的垃圾
    - 奖励：回收一个垃圾奖励+1（等待和主动寻找获得奖励的概率不同），人工干预一次奖励-3
  - 教育孩子？员工管理？

累积奖励

- 累积奖励（也叫回报：return）

$$G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_T$$

T表示最后一步，对应的状态被称为终止态

- 具有终止态的马尔科夫决策过程被称为“多幕式”任务

- 没有终止态的任务称为连续式任务，其累积奖励记为

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

$0 \leq \gamma \leq 1$表示折扣率

累积奖励的递推公式

- 递推公式

$$\begin{aligned} G_t &\doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \cdots \\ &= R_{t+1} + \gamma\left(R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} + \cdots\right) \\ &= R_{t+1} + \gamma G_{t+1} \end{aligned}$$

- 求和公式

$$G_t \doteq \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$$

注意：$\gamma = 1$和$T = \infty$不能同时出现

策略和状态估值函数

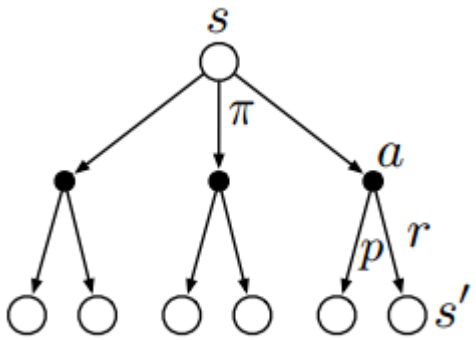
- 策略
  - 状态到行为的映射
  - 随机式策略：$\pi(a|s)$
  - 确定式策略：$a = \pi(s)$
- 给定策略π，状态估值函数(state-value function)定义为

$$v_\pi(s) \doteq \mathbb{E}_\pi[G_t \mid S_t = s] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s\right], \text{ for all } s \in \mathcal{S}$$

- 给定策略π，行为估值函数(action-value function)定义为

$$q_\pi(s,a) \doteq \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s, A_t = a\right]$$

贝尔曼方程



Backup diagram for v_π

- 状态估值函数的贝尔曼方程

$$
\begin{aligned}
v_\pi(s) &\doteq \mathbb{E}_\pi[G_t \mid S_t = s] \\
&= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s] \\
&= \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r \mid s, a) \Big[r + \gamma \mathbb{E}_\pi[G_{t+1} | S_{t+1} = s'] \Big] \\
&= \sum_a \pi(a|s) \sum_{s', r} p(s', r \mid s, a) \Big[r + \gamma v_\pi(s') \Big], \quad \text{for all } s \in \mathcal{S}
\end{aligned}
$$

Tip!

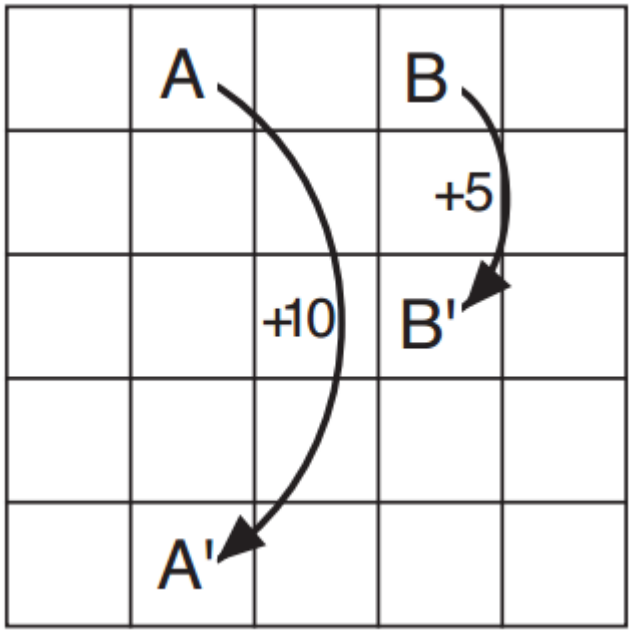
Bellman equation expresses a relationship between the value of a state and the values of its successor states

贝尔曼方程的作用

- **贝尔曼方程**定义了状态估值函数的依赖关系
  - 给定策略下，每个状态的估值视为一个变量
  - 所有状态（假如有n个）的估值根据贝尔曼方程形成了一个具有n个方程和n个变量的线性方程组
  - 求解该方程组即可得到该策略下每个状态的估值

例子：Gridworld

- 游戏规则：正常移动一次的奖励为0；出界时回到当前位置，奖励为-1；A位置选择任意方向都到达A‘，奖励为+10；B位置选择任意方向都到达B’，奖励为+5；折扣率$\gamma = 0.9$。
- 假定策略为：从每个格子等概率选择四个移动方向（行为）



游戏规则

策略

3.3	8.8	4.4	5.3	1.5
1.5	3.0	2.3	1.9	0.5
0.1	0.7	0.7	0.4	-0.4
-1.0	-0.4	-0.4	-0.6	-1.2
-1.9	-1.3	-1.2	-1.4	-2.0

策略对应的状态估值函数

注意：这里的计算精度为小数点后一位

最优策略

- 状态估值函数定义了策略空间上的一个偏序
  - 给定两个策略π和π'，如果对于所有状态s，都有$v_\pi(s) \geq v_{\pi'}(s)$，则我们说$\pi \geq \pi'$

- 最优策略
  - 最优策略π_*对应于如下状态估值函数

$$v_*(s) \doteq \max_\pi v_\pi(s)$$

  - 最优策略可以有多个，所对应的状态估值函数都一样
  - 最优策略对应的行为估值函数

$$q_*(s,a) \doteq \max_\pi q_\pi(s,a)$$

贝尔曼最优性方程

- 状态估值函数的贝尔曼最优性方程

$$
\begin{aligned}
v_*(s) &= \max_{a \in \mathcal{A}(s)} q_{\pi_*}(s,a) \\
&= \max_a \mathbb{E}_{\pi_*}[G_t \mid S_t = s, A_t = a] \\
&= \max_a \mathbb{E}_{\pi_*}[R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t = a] \\
&= \max_a \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) \mid S_t = s, A_t = a] \\
&= \max_a \sum_{s',r} p(s', r \mid s, a)\big[r + \gamma v_*(s')\big].
\end{aligned}
$$

- 行为估值函数的贝尔曼最优性方程

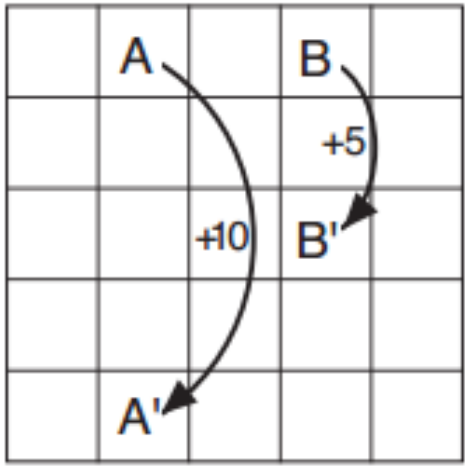
$$
\begin{aligned}
q_*(s,a) &= \mathbb{E}\Big[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') \;\Big|\; S_t = s, A_t = a\Big] \\
&= \sum_{s',r} p(s', r \mid s, a)\Big[r + \gamma \max_{a'} q_*(s', a')\Big].
\end{aligned}
$$

寻找最优策略

- 基于状态估值函数的贝尔曼最优性方程
  - 第一步：求解状态估值函数的贝尔曼最优性方程得到最优策略对应的状态估值函数
  - 第二步：根据状态估值函数的贝尔曼最优性方程，进行一步搜索找到每个状态下的最优行为
    - 注意：最优策略可以存在多个
    - 贝尔曼最优性方程的优势，可以采用贪心局部搜索即可得到全局最优解
- 基于行为估值函数的贝尔曼最优性方程
  - 直接得到最优策略

例子：Gridworld

- 最优状态估值函数和最优策略

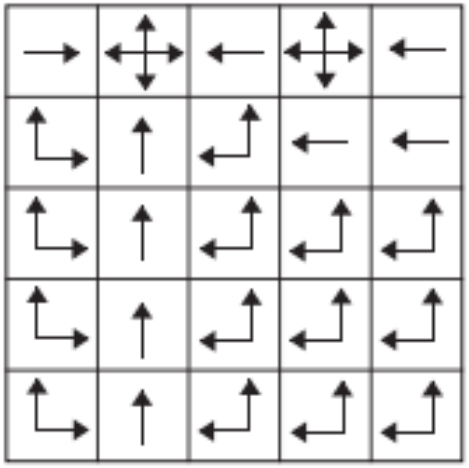


Gridworld

22.0	24.4	22.0	19.4	17.5
19.8	22.0	19.8	17.8	16.0
17.8	19.8	17.8	16.0	14.4
16.0	17.8	16.0	14.4	13.0
14.4	16.0	14.4	13.0	11.7

v_*

最优策略对应的状态估值函数



π_*

最优策略

寻找最优策略小结

- 求解贝尔曼最优性方程寻找最优策略的局限性
  - 需要知道环境模型

$$p(s', r|s, a)$$

  - 需要高昂的计算代价和内存（存放估值函数）
  - 依赖于马尔科夫性

- 下一节课讲解实际应用中寻找最优策略的方法
  - 动态规划方法、蒙特卡洛方法、时序查分方法、参数化方法

下课