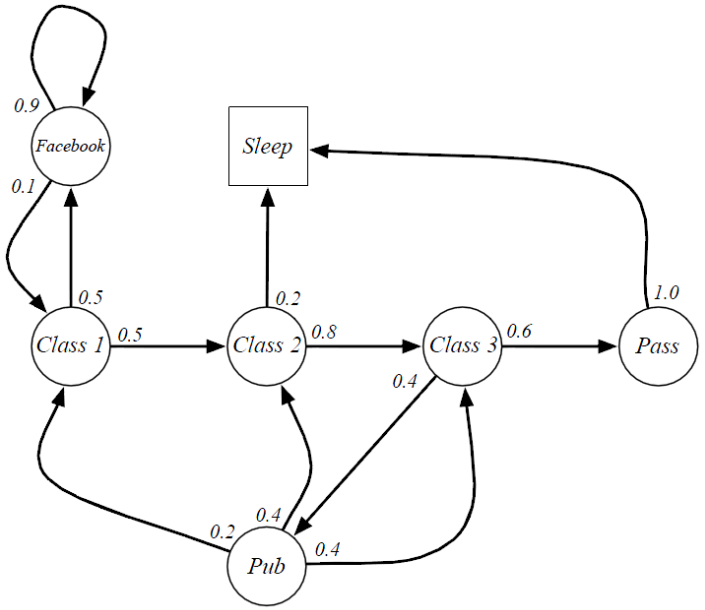
《机器学习及其在化学中的应用》2025年课程

Sec. 12

强化学习2：马尔科夫决策过程



刘志荣 (LiuZhiRong@pku.edu.cn)

北京大学化学学院

2025.12.1

内容提要

- 有限马尔科夫决策过程
- 目标、策略、价值函数
- 贝尔曼方程与最优策略

马尔科夫过程（Markov process）

- 马尔科夫性：未来只受现在影响，与过去无关。

$$p(s_{t+1}|s_1, s_2, \dots, s_t) = p(s_{t+1}|s_t)$$

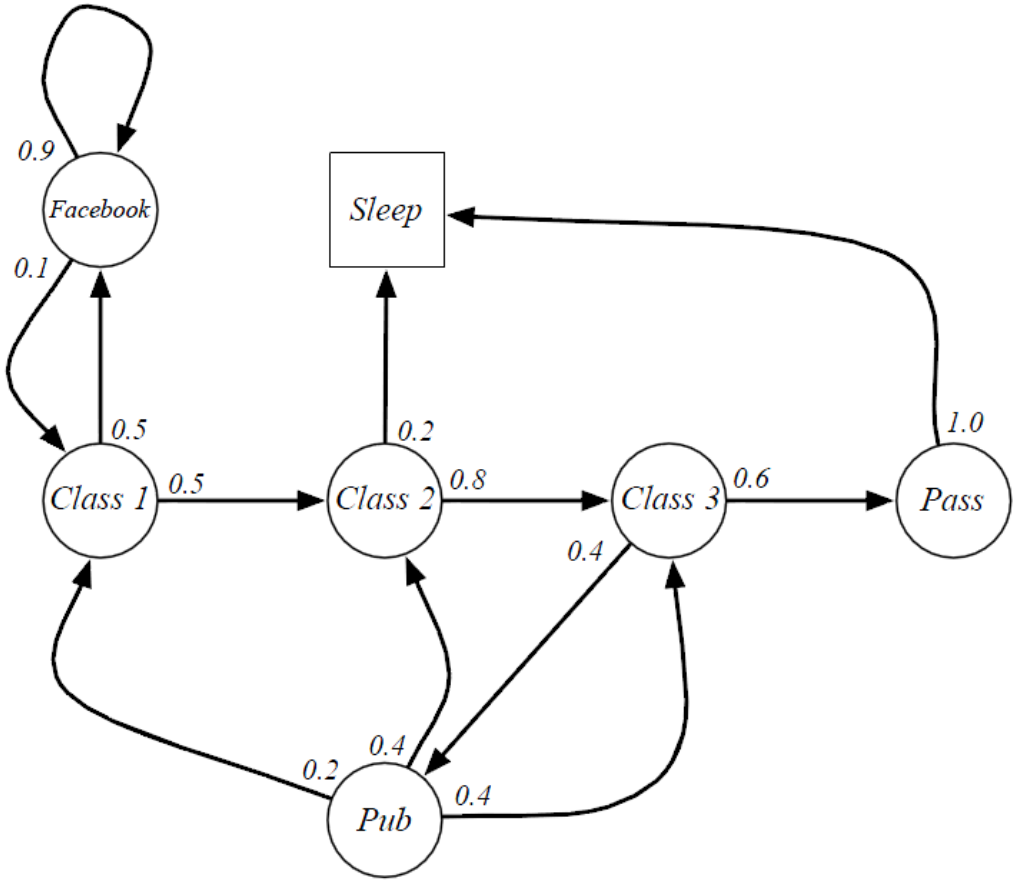
  - 当前状态包含了历史所有能影响未来的信息。
- 马尔科夫过程：满足马尔科夫性的随机过程。
- 如果状态数目有限（分立），则可用状态转移矩阵描述：

$$p(s_{t+1}|s_t) = p(s'|s) = \begin{bmatrix} P_{11} & \cdots & P_{1n} \\ \vdots & \ddots & \vdots \\ P_{n1} & \cdots & P_{nn} \end{bmatrix}$$

满足$\sum_{j=1}^{n} P_{ij} = 1$

行（号）：转移前状态；
列（号）：转移后状态。

例子: Student Markov Chain...



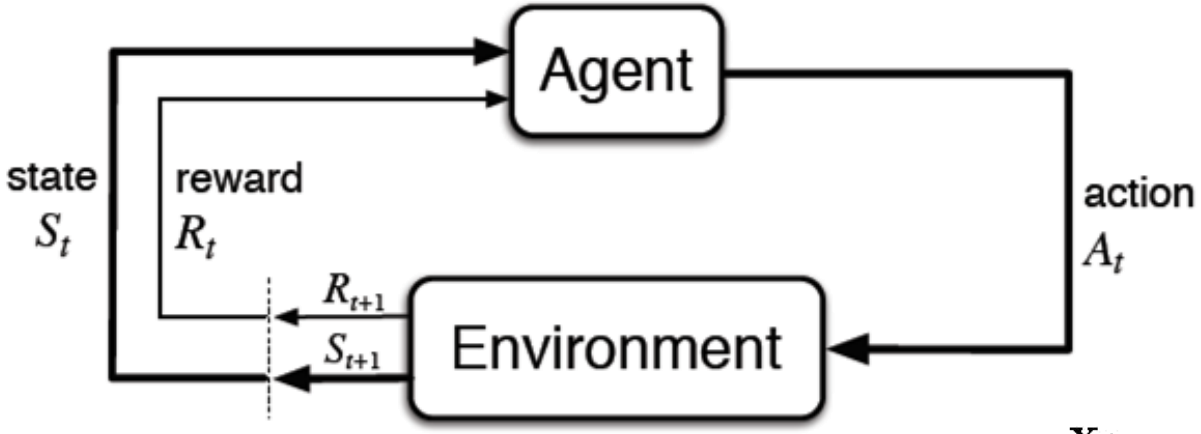
$$\mathcal{P} = \begin{array}{c} \\ C1 \\ C2 \\ C3 \\ Pass \\ Pub \\ FB \\ Sleep \end{array} \begin{array}{c} \begin{array}{ccccccc} C1 & C2 & C3 & Pass & Pub & FB & Sleep \end{array} \\ \begin{bmatrix} & 0.5 & & & & 0.5 & \\ & & 0.8 & & & & 0.2 \\ & & & 0.6 & 0.4 & & \\ & & & & & & 1.0 \\ 0.2 & 0.4 & 0.4 & & & & \\ 0.1 & & & & & 0.9 & \\ & & & & & & 1 \end{bmatrix} \end{array}$$

一些可能的轨迹:

- C1 C2 C3 Pass Sleep
- C1 FB FB C1 C2 Sleep
- C1 C2 C3 Pub C2 C3 Pass Sleep
- C1 FB FB C1 C2 C3 Pub C1 FB FB FB C1 C2 C3 Pub C2 Sleep

有限马尔科夫决策过程

- Finite Markov decision process (MDP)
- 动力学行为决定于概率$p(s', r|s, a)$
- 在前面的多臂老虎机问题中，我们考虑行动a的价值$q_*(a)$，而在MDPs中我们同时考虑状态s与行动a的影响，$q_*(s, a)$



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

基于$p(s',r|s,a)$，可计算其它的一些重要量：

- 状态转移概率

$$p(s'|s,a)=\sum_{r\in\mathcal{R}}p(s',r|s,a)$$

- 状态-行动的回报期望值

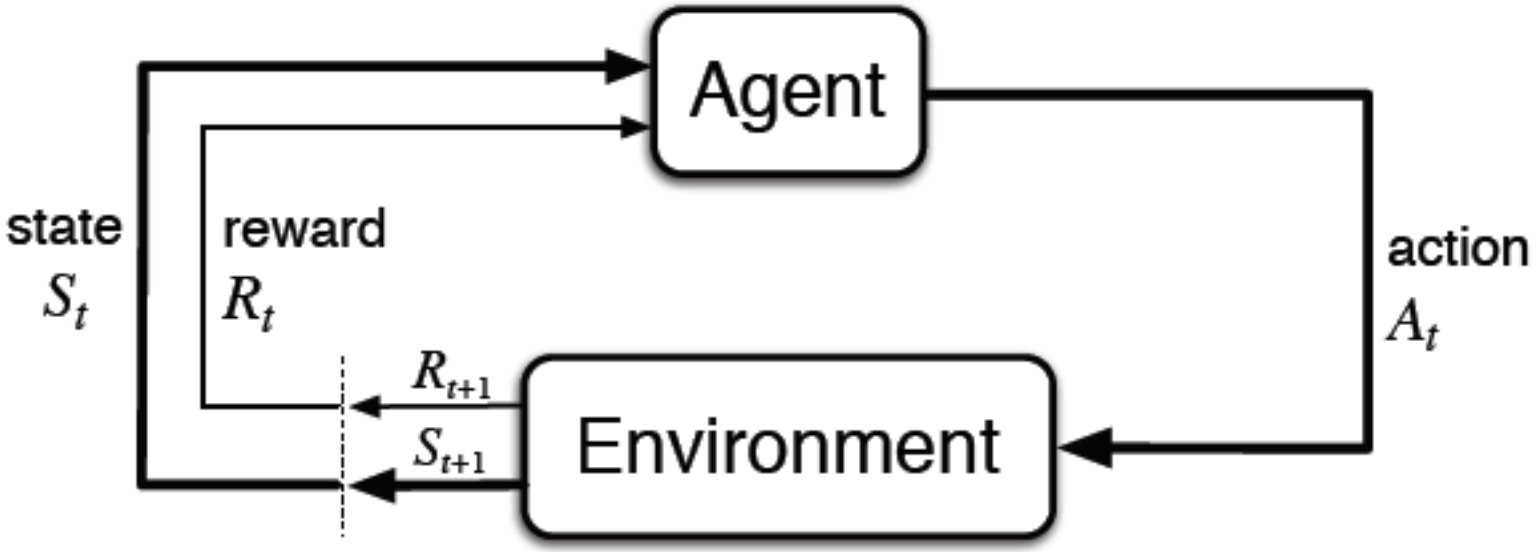
$$r(s,a)=\sum_{r\in\mathcal{R},s'\in\mathcal{S}}rp(s',r|s,a)$$

- 状态-行动-后继状态的回报期望值

$$r(s,a,s')=\sum_{r\in\mathcal{R}}rp(r|s,a,s')=\sum_{r\in\mathcal{R}}r\frac{p(s',r|s,a)}{p(s'|s,a)}$$

MDP框架把目标导向行为的强化学习概括成智能体与环境之间来回传递的三种信号：

- 一个信号用来表示智能体做出的选择（行动） a；
- 一个信号表示做出该选择的基础（状态） s；
- 还有一个信号定义了智能体的目标（回报） r。



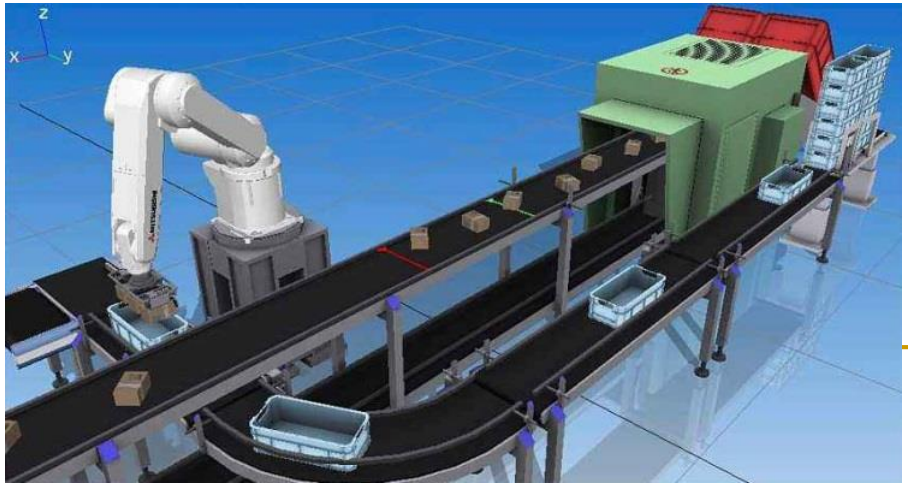
3.1: The agent–environment interaction in a Markov decision process.

例子：生物反应器

- 行动：加热（加热元件）
  与搅拌（电动机）。
- 状态：传感器读数、化学组分。
- 回报：目标化合物产生速率。

例子：拣放机器人 Pick-and-Place Robot

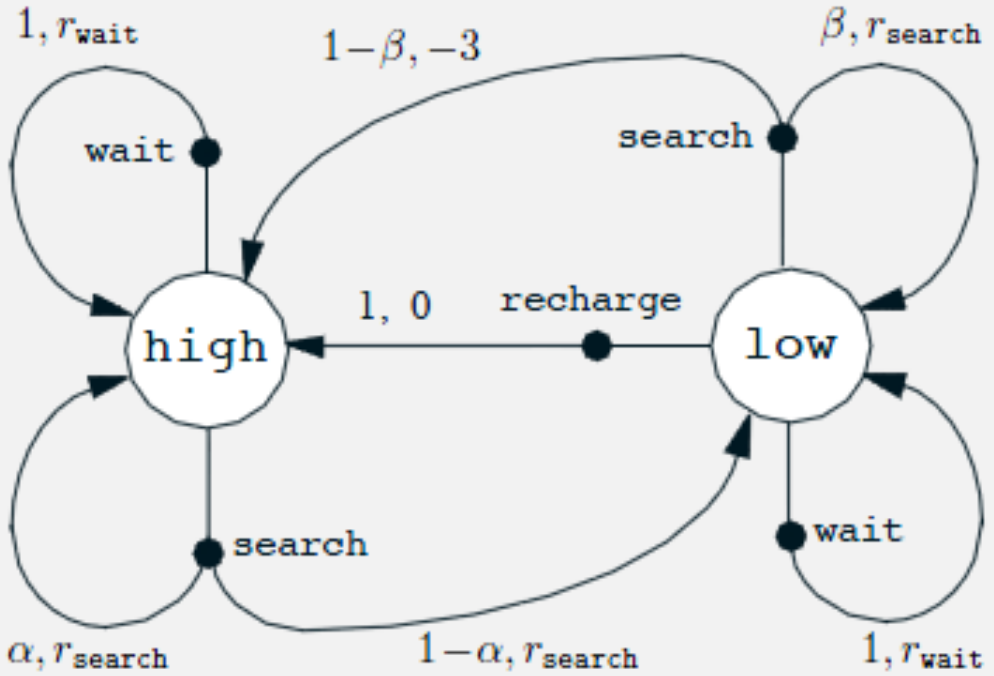
- 行动：电动机电压。
- 状态：连接处位置与速度。
- 回报：成功拣放时+1。



例子：易拉罐回收机器人

- 状态：电量高与低。
- 动作：主动搜索、原地等待、去充电。
- 回报：得到一个易拉罐时+1。

s	a	s'	$p(s' \mid s, a)$	$r(s, a, s')$
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	-
low	wait	high	0	-
low	wait	low	1	$r_{\texttt{wait}}$
low	recharge	high	1	0
low	recharge	low	0	-



目标函数

- 在t时刻的决策目标，是使之后接收的回报最大化。在最简单的情况下，定义目标函数（Goal，有时称为return）

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

  - 其中T是终止时刻。这适合回合制（分幕制，episodes）的任务。

- 有时候涉及无穷长时间的任务，此时一般引入贬值（折扣、贴现）的观点，定义

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

- 合起来，可写成：$G_t \doteq \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$

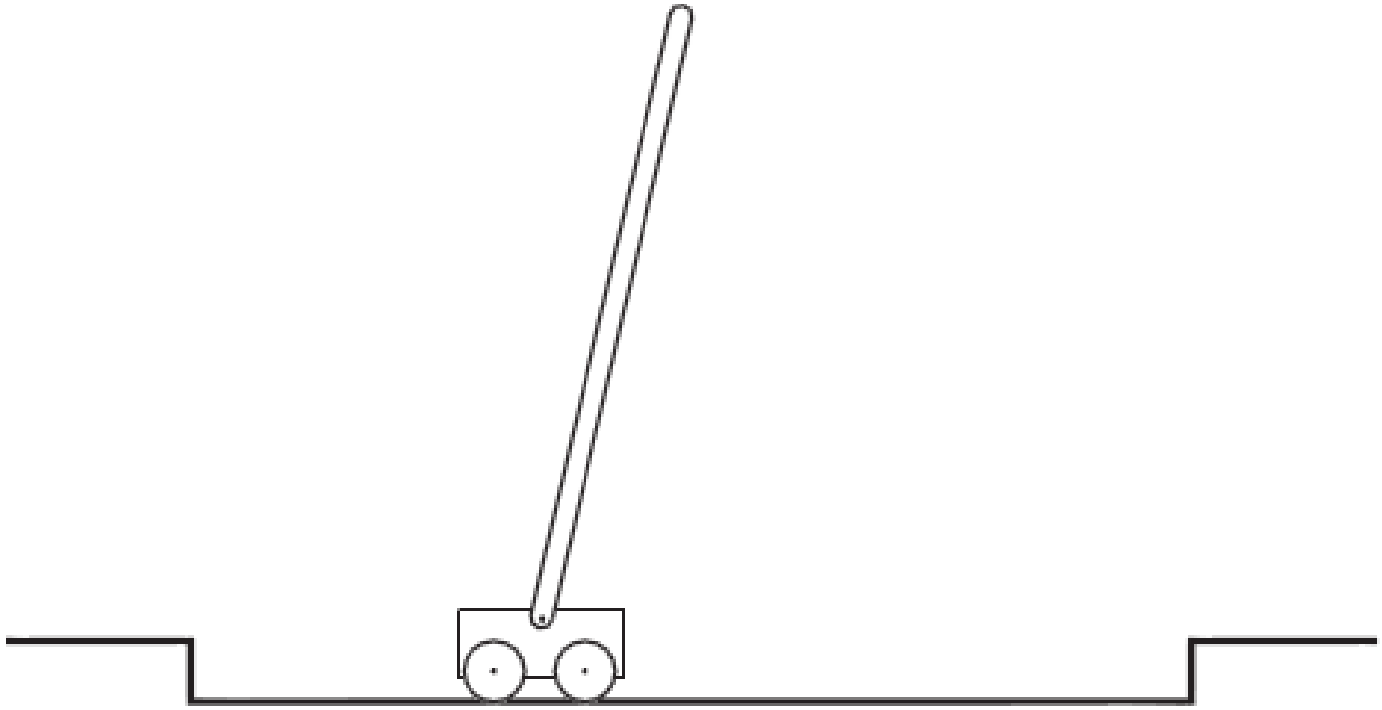
满足

$$G_t = R_{t+1} + \gamma G_{t+1}$$

例子：小车上的杆平衡

- 方案一：回合制，每步得分$+1$
- 方案二：无穷长时间，倒下时得分-1，其它0。

最终效果是等价的。



策略（policy）与价值（value）函数

- 策略函数给出了当状态为s时采取行动a的概率：

$$\pi(a|s)$$

- 状态s在策略π下的价值函数是其最后达到目标的期望值：

$$v_\pi(s) \doteq \mathbb{E}_\pi[G_t|S_t = s]$$

  - 称为策略π的状态价值函数（state-value function）。

- 类似地，在策略π下对状态s采取行动a的价值函数是

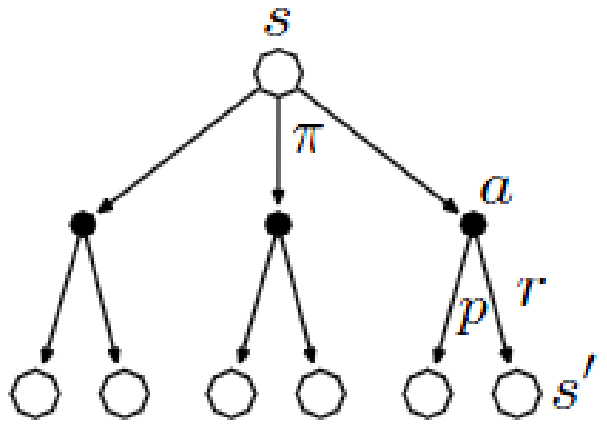
$$q_\pi(s,a) \doteq \mathbb{E}_\pi[G_t|S_t = s, A_t = a]$$

先采取行动a，
其后再遵循π

  - 称为策略π的行动价值函数（action-value function）。

- 因此 $v_\pi(s) = \sum_a q_\pi(s,a)\pi(a|s)$

贝尔曼（Bellman）方程



Backup diagram for v_π

$$v_\pi(s) \doteq \mathbb{E}_\pi[G_t|S_t = s]$$

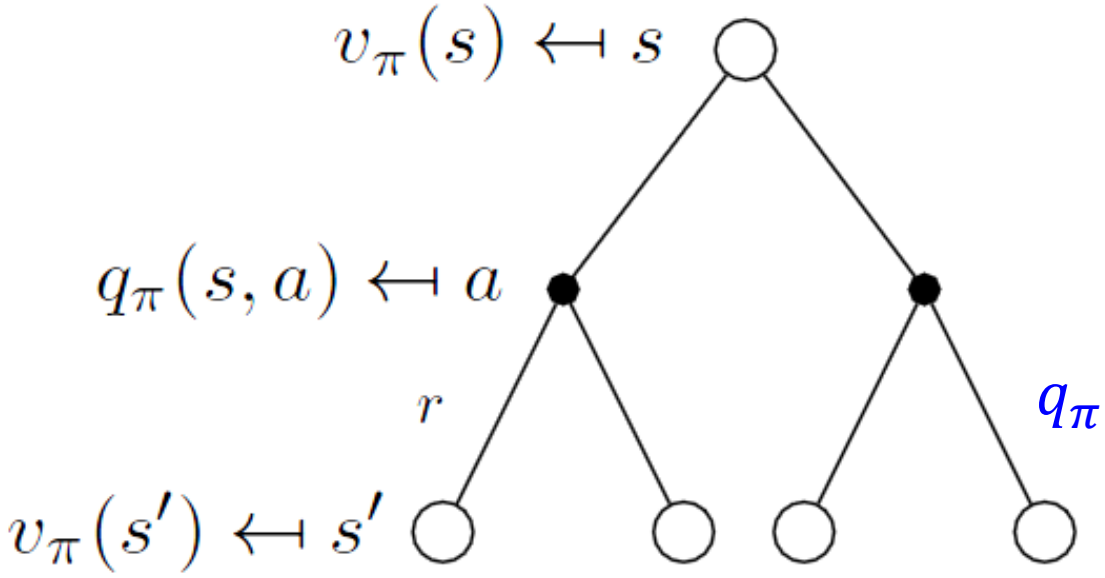
$$= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s]$$

$$= \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\left[r + \gamma \mathbb{E}_\pi[G_{t+1}|S_{t+1} = s']\right]$$

$$= \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$$

- 线性方程组。它可以基于$p(s',r|s,a)$与$\pi(a|s)$求出$v_\pi(s)$
- 构成了很多策略学习算法的理论基础。

回溯图（backup diagram）



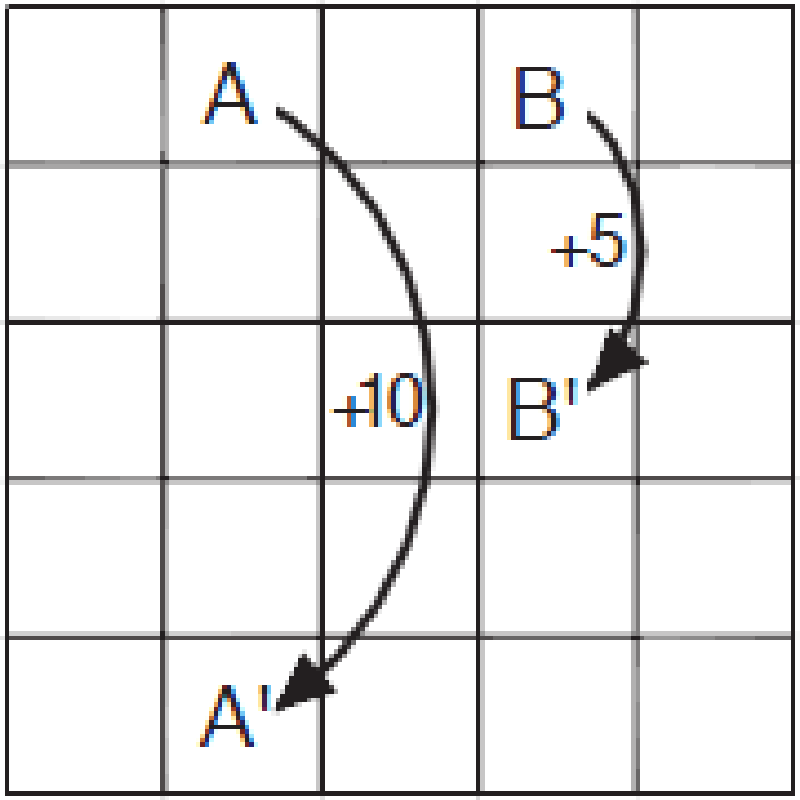
$$v_\pi(s) = \sum_a q_\pi(s,a)\pi(a|s)$$

$$q_\pi(s,a) = \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$$

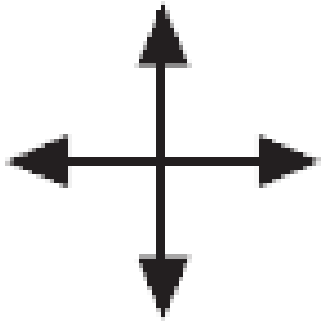
$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$$

例子：网格世界（Gridword）

- 行动：上下左右。
- 后果：被移动一步（回报0）、不动（撞墙，回报-1）、A所示、B所示。



$\gamma = 0.9$



Actions

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

随机移动策略下的状态价值函数。

最优策略（optimal policy）

- 在所有策略中，总存在一个（或一组）最优策略π^*，其状态价值函数记为$v_*(s)$，其行动价值函数记为$q_*(s,a)$。
  - 最优策略下的一个状态的价值一定等于这个状态下最优动作的价值：

$$
\begin{aligned}
v_*(s) &= \max_a q_{\pi*}(s,a) \\
&= \max_a \mathbb{E}_{\pi*}[G_t|S_t=s,A_t=a] \\
&= \max_a \mathbb{E}_{\pi*}[R_{t+1}+\gamma G_{t+1}|S_t=s,A_t=a] \\
&= \max_a \mathbb{E}_{\pi*}[R_{t+1}+\gamma v_*(S_{t+1})|S_t=s,A_t=a] \\
&= \max_a \sum_{s',r} p(s',r|s,a)[r+\gamma v_*(s')]
\end{aligned}
$$

贝尔曼最优方程

- 类似地，可以得到关于行动价值函数的贝尔曼最优方程：

$$q_*(s,a) = \sum_{s',r} p(s',r|s,a)\left[r + \gamma \max_{a'} q_*(s',a')\right]$$

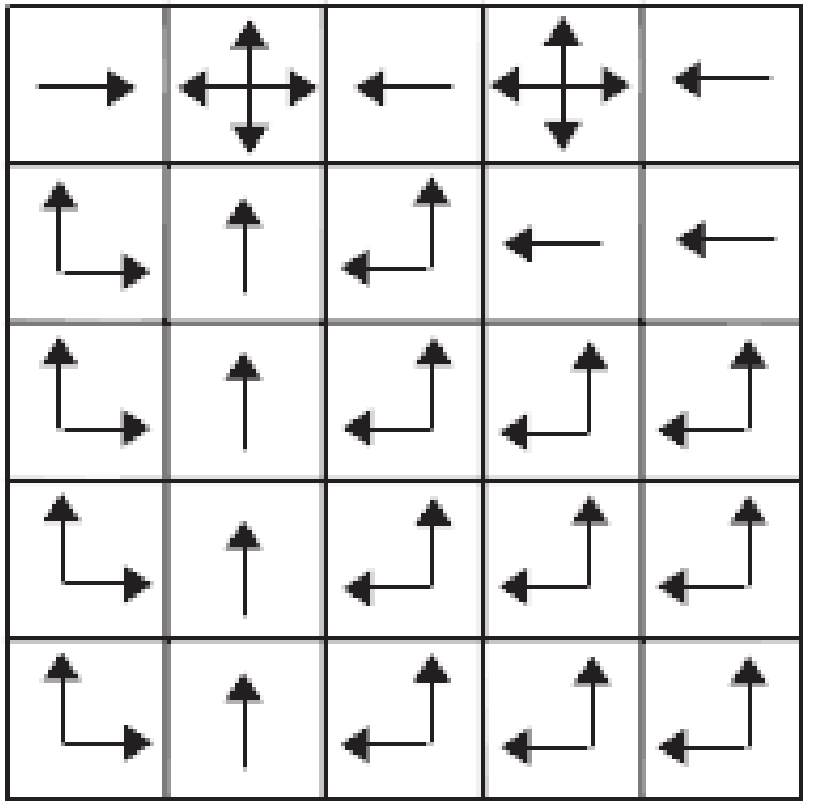
- 原则上可以基于贝尔曼最优方程求解$v_*(s)$与$q_*(s,a)$。
- $v_*(s)$与$q_*(s,a)$包含了长期回报的信息。
- 如果$v_*(s)$与$q_*(s,a)$已经求得，则（最优）决策变得非常容易，只需要利用$v_*(s)$往前搜索一步（$\max_a \mathbb{E}_{\pi*}[R_{t+1} + \gamma v_*(S_{t+1})|S_t = s, A_t = a]$）或直接最大化$q_*(s,a)$就可以了。
- 很多决策方法都可以看做是贝尔曼最优方程的近似求解。

例子：网格世界的最优策略及其价值函数

$\gamma = 0.9$

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_*



π_*

小结

- 有限马尔科夫决策过程
  - 体系状态s与行动a的可能取值是有限（分立）的；
  - 下一状态与回报只决定于当前状态及行动：$p(s', r|s, a)$。
- 目标函数（goal, return）：$G_t \doteq \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$
- 策略：当状态为s时采取行动a的概率，$\pi(a|s)$
- 价值函数：状态s（及行动a）在策略π下的最后达到目标的期望值
- 贝尔曼方程：$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_\pi(s')]$
- 最优策略与最优方程：$v_*(s) = \max_a \sum_{s',r} p(s', r|s, a)[r + \gamma v_*(s')]$

- Reference:
  - Sutton 3；
- 扩展阅读：
  - Silver, Lecture 2: Markov Decision Processes. 2-MDP.pdf
  - https://www.jiqizhixin.com/graph/technologies/9a91aa59-8dc1-4eff-833b-6180ba53cb68
    贝尔曼方程 机器之心.mht
  - https://36kr.com/p/1263509580895877
    DeepMind大神Silver联手Sutton论证无限猴子原理：用强化学习就能搞定通用人工智能.mht

谢谢大家！