强化学习

提 纲

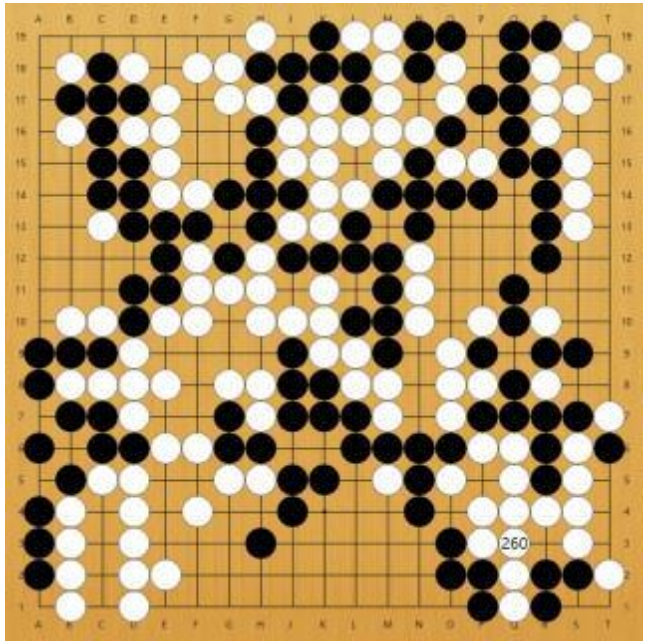
1、 强化学习定义： 马尔可夫决策过程

2、 强化学习中策略优化与策略评估

3、 强化学习求解：Q-Learning

4、 深度强化学习： 深度学习+强化学习

强化学习的应用

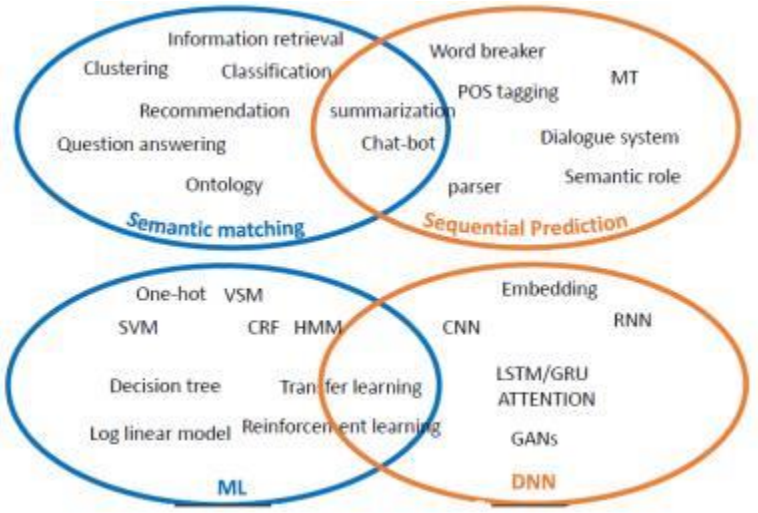


围棋游戏

注：AlphaGo的三大法宝：

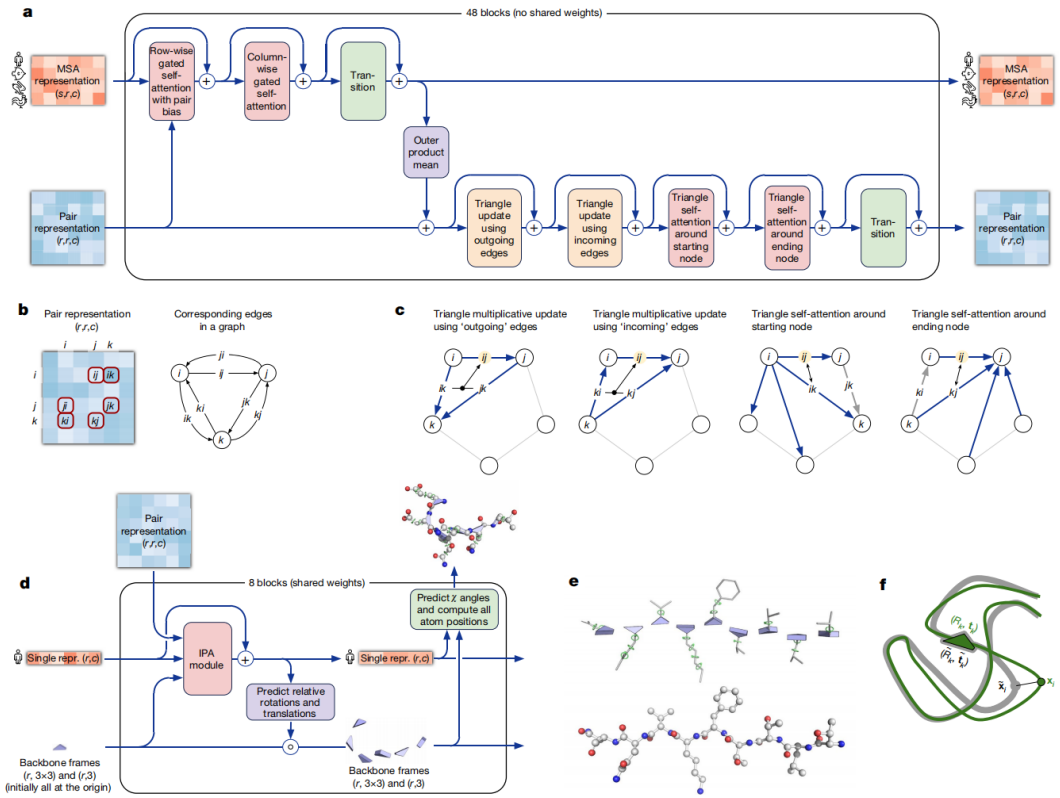
- 深度学习(感知棋面)
- 强化学习（自我博弈）
- 蒙特卡洛树搜索（采样学习）

机器人运动



自然语言理解

DeepMind：AlphaFold2



Nature：Highly accurate protein structure prediction with AlphaFold

开源：
https://github.com/deepmind/alphafold

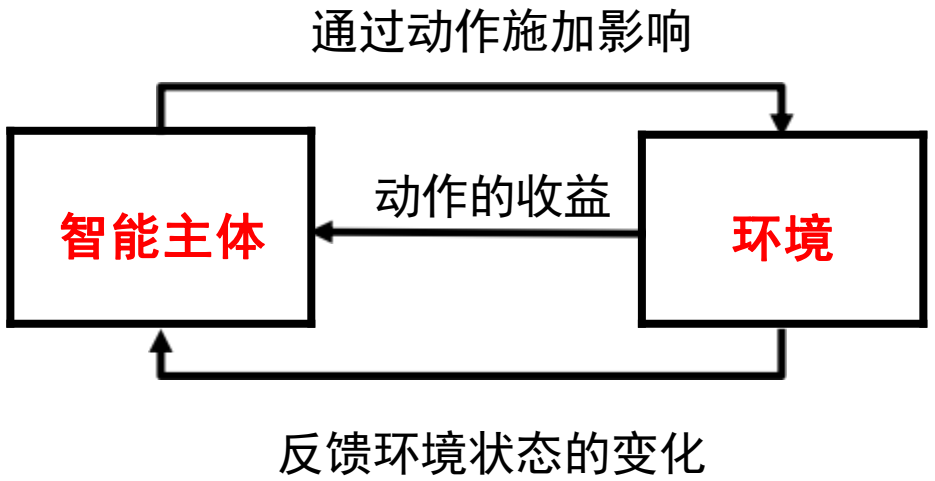
DeepMind

- 2010年，**戴密斯·哈萨比斯**等人创建了DeepMind。
- 2014年1月，**谷歌**计划斥资4亿美元收购**人工智能初创企业**DeepMind。
- 2016年3月，DeepMind开发的**AlphaGo**程序以4：1击败韩国围棋冠军李世石，成为近年来人工智能领域少有的里程碑事件。
- 2017年，DeepMind发布了**AlphaGo Zero**，在自我训练3天后以100-0击败了AlphaGo。
- 2020年11月30日，DeepMind发布消息称，其人工智能系统“**AlphaFold**”人工智能系统参加了由结构预测关键评估组织（CASP）的一项**如何计算蛋白质分子3D结构**的竞赛，并且预测准确性达到前所未有的水平。DeepMind表示，这将“为解决人类50年来的巨大挑战铺平道路”。
- 2022年2月消息，DeepMind发布了基于**Transformer模型**的AlphaCode，可以**编写与人类相媲美的计算机程序**。
- 2022年3月10日，DeepMind与威尼斯大学人文系、牛津大学古典学院以及雅典经济与商业大学信息学系联合发表了伊萨卡，第一个可以**复原受损铭文的缺失文本**、识别铭文原始（书写）位置、确定创建日期的深度神经网络。
- 2022年7月，人工智能公司DeepMind进一步**破解了几乎所有已知的蛋白质结构**，其AlphaFold算法构建的数据库中如今包含了超过2亿种已知蛋白质结构，为开发新药物或新技术来应对饥荒或污染等全球性挑战铺平了道路。
- 2022年9月，Alphabet 旗下人工智能实验室 DeepMind推出**人工智能聊天机器人 Sparrow**。
- 2023年5月，DeepMind推出了名为 **Pi 的 AI 机器人**，定位是朋友对话者，而非辅助工具。
- 2023年6月7日，AI研究实验室Google DeepMind的研究人员发布了一个新的AI系统，可以**提高计算的效率和可持续性**。

强化学习：在与环境交互之中进行学习

生活中常见的学习过程

- 人通过动作对环境产生影响（向前走一步）
- 环境向人反馈状态的变化（撞到了树上）
- 人估计动作得到的收益（疼痛）
- 更新做出动作的策略（下次避免向有树这一障碍的方向前进）



强化学习模仿了这个过程，在智能体与环境的交互中，学习能最大化收益的行动模式。

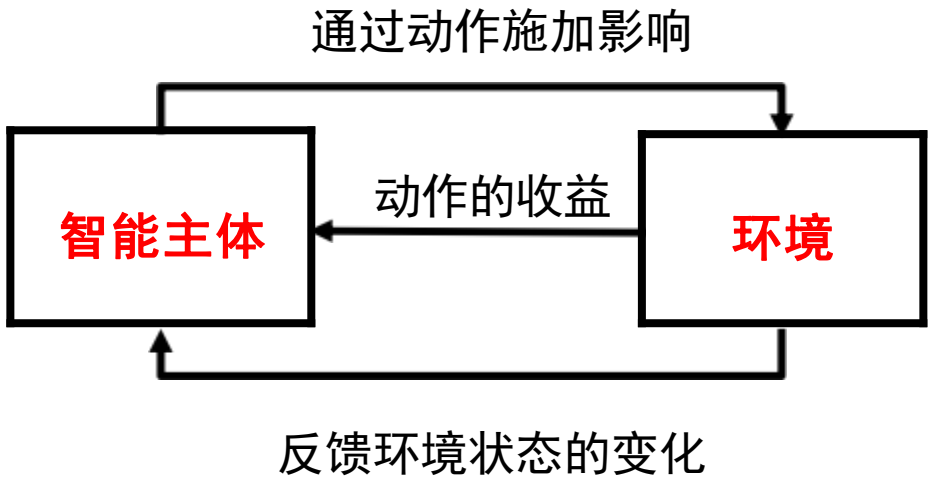
强化学习中的概念

智能主体(Agent)

- 按照某种**策略**(**Policy**)，根据当前的**状态**(**State**) 选择合适的**动作**(**Action**)。
- 状态指的是智能主体对环境的一种解释。
- 动作反映了智能主体对环境主观能动的影响， 动作带来的收益称为**奖励**(**Reward**)。
- 智能主体可能知道也可能不知道环境变化的规律。

环境(Enviroment)

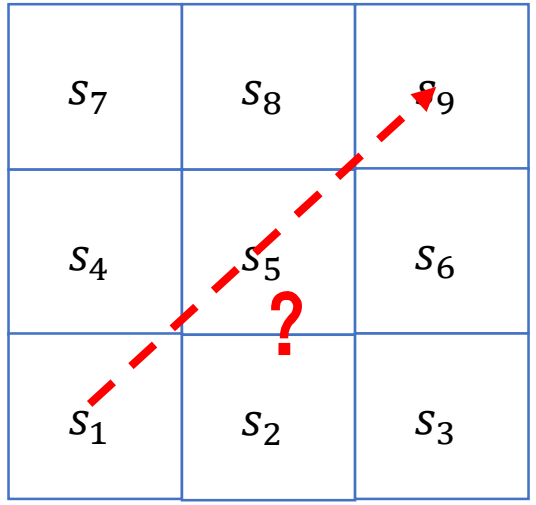
- 系统中智能主体以外的部分。
- 向智能主体反馈状态和奖励。
- 按照一定的规律发生变化。



关键词：策略(Policy)， 状态(State)， 动作(Action)，奖励(Reward)。

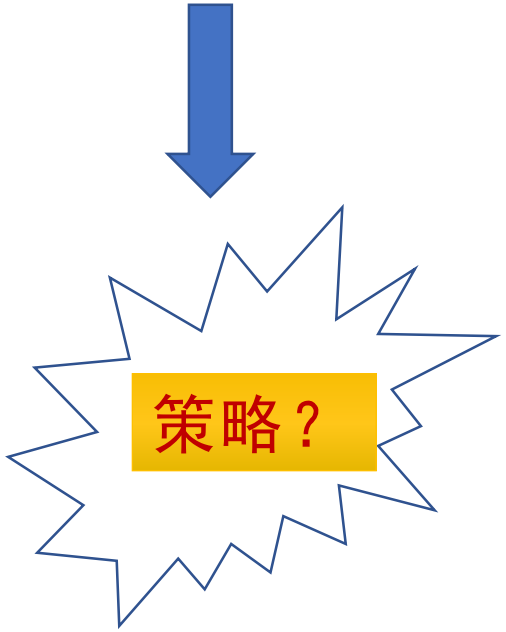
强化学习示例：机器人移动问题

- 在下图网格中，假设有一个机器人位于s_1，其每一步只能向上或向右移动一格，跃出方格会被惩罚（且游戏停止）
- 如何使用强化学习找到一种策略，使机器人从s_1到达s_9？



刻画解该问题的因素

智能主体	迷宫机器人
环境	3×3方格
状态	机器人当前时刻所处方格
动作	每次移动一个方格
奖励	到达s_9时给予奖励；越界时给予惩罚



机器学习的不同类型

	有监督学习	无监督学习	强化学习(决策)
学习依据	基于监督信息	基于对数据结构的假设	基于评价(Evaluative)
数据来源	一次性给定	一次性给定	在交互中产生(Interactive)
决策过程	单步(One-shot)	无	序贯(Sequential)
学习目标	样本到语义标签的映射	同一类数据的分布模式	选择能够获取最大收益的的状态到动作的映射

强化学习的特点

- **基于评估(评价)：** 强化学习利用环境评估当前策略，以此为依据进行优化。
- **交互性：** 强化学习的数据在与环境的交互中产生。
- **序贯决策过程：** 智能主体在与环境的交互中需要作出一系列的决策，这些决策往往是前后关联的。

注：现实中常见的强化学习问题往往还具有奖励滞后，基于采样的评估等特点。

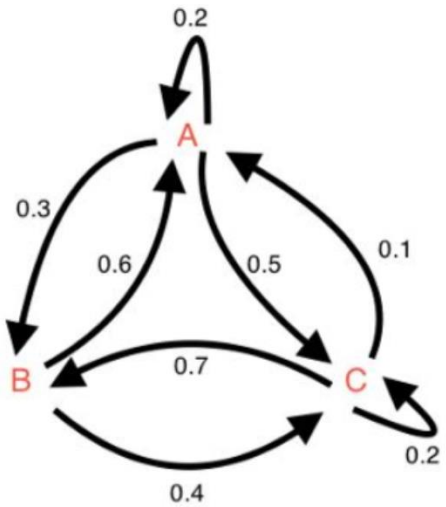
附：离散马尔可夫过程(Discrete Markov Process)

- 一个随机过程实际上是一列随时间变化的随机变量，其中当时间是离散量时，一个随机过程可以表示为$\{X_t\}_{t=0,1,2,\cdots}$，其中每个X_t都是一个随机变量，这被称为离散随机过程。
- **马尔可夫链**（Markov Chain）：满足**马尔可夫性**（Markov Property）的离散随机过程，也被称为离散马尔可夫过程。

$$Pr(X_{t+1}=x_{t+1}|X_0=x_0,X_1=x_1,\cdots,X_t=x_t)=Pr(X_{t+1}=x_{t+1}|X_t=x_t)$$

$t+1$时刻状态仅与t时刻状态相关

$$=Pr(X_{t+1}=x_{t+1}|X_t=x_t,X_{t-1}=x_{t-1})$$

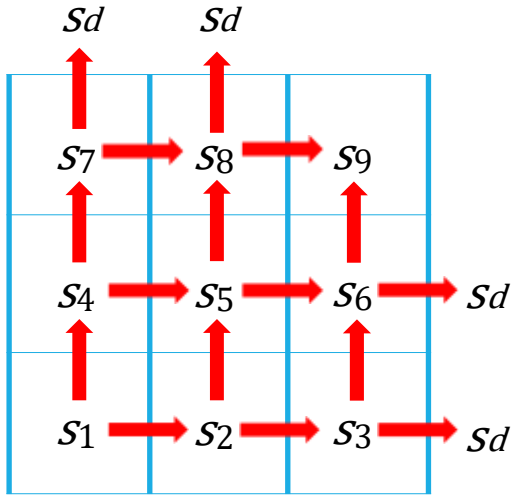


二阶马尔可夫链：$t+1$时刻状态与t和$t-1$时刻状态相关

无记忆性

离散马尔可夫过程：机器人移动问题

MP = $\{S, Pr\}$可用来刻画该问题



- 随机变量序列 $\{S_t\}_{t=0,1,2,\cdots}$， 其中S_t表示机器人第t步的位置，每个随机变量S_t的取值范围为$S=\{s_1, s_2, \cdots, s_9, s_d\}$。

- 状态（State）转移概率$Pr(S_{t+1}|S_t)$满足马尔可夫性。它的一种取值如图中箭头所示，每个箭头对应0.5的转移概率。

S集合（即状态空间）可为无限的，如用经纬度坐标表示现实中机器人的位置。

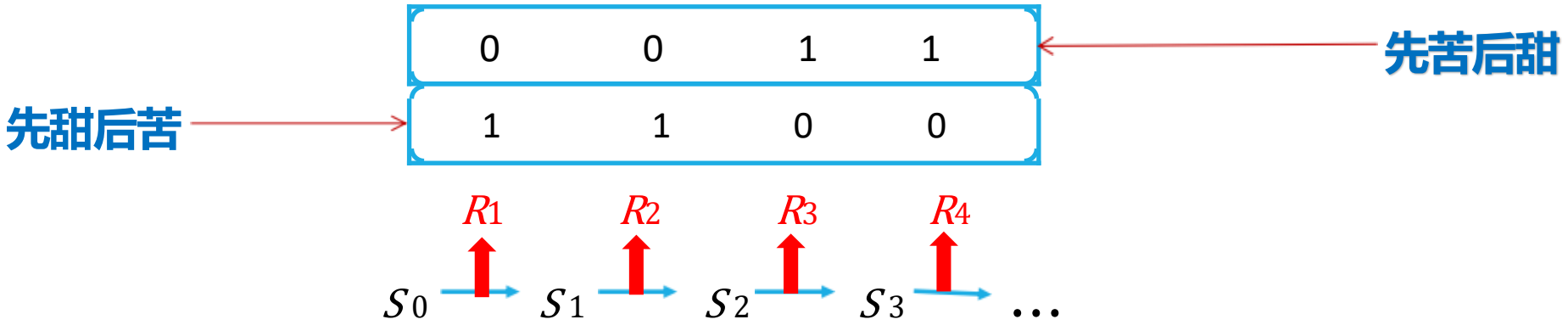
这个模型不能体现机器人能动性，缺乏与环境进行交互的手段！-无法评价好坏。

马尔可夫奖励过程(Markov Reward Process)：引入奖励

为了在序列决策中对目标进行优化， 在马尔可夫随机过程框架中加入了奖励机制：

- **奖励函数R**: $S \times S \mapsto \mathbb{R}$，其中$R(S_t, S_{t+1})$描述了从第$t$步状态转移到第$t+1$步状态所获得奖励。一般为正,就是奖励,鼓励这种行为,为负,就是惩罚,不鼓励这种行为.
- 在一个序列决策过程中，不同状态之间的转移产生了一系列的奖励$(R_1, R_2, \cdots)$，其中R_{t+1}为$R(S_t, S_{t+1})$的简便记法。**（R又叫回报,Reward）**
- 引入奖励机制， 这样可以衡量任意序列的优劣，即对序列决策进行评价。

问题： 给定两个因为状态转移而产生的奖励序列$(1, 1, 0, 0)$和$(0, 0, 1, 1)$， 哪个序列决策更好？



马尔可夫奖励过程(Markov Reward Process)

问题：给定两个因为状态转移而产生的奖励序列(1,1,0,0)和(0,0,1,1)，哪个奖励序列更好？

为了比较不同的奖励序列，定义累计回报（又叫反馈：Return），用来评估策略的好坏：

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots$$

其中折扣系数（Discount Factor）$\gamma \in [0,1]$,作用:

1)避免回报取值无穷大.2)区分远近行为的回报.

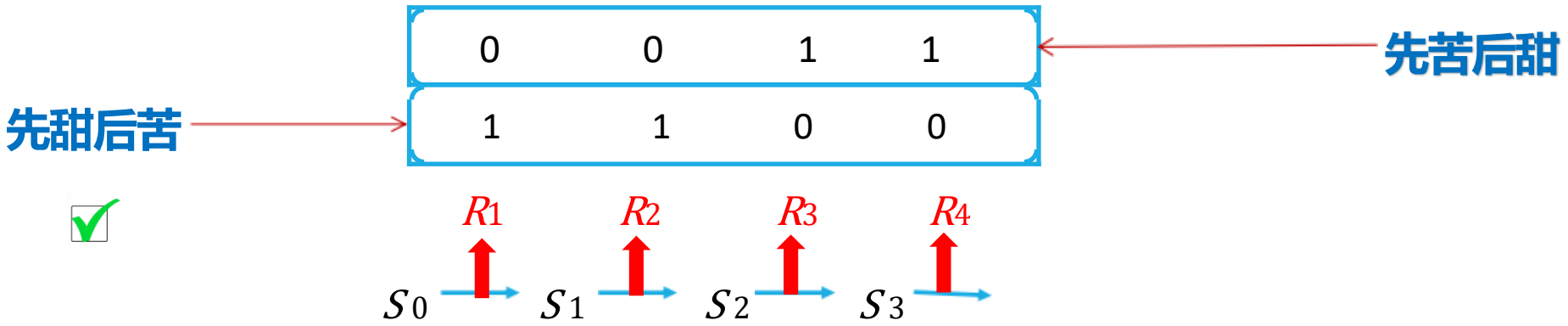
假设 $\gamma = 0.99$

(1,1,0,0)：$G_0 = 1 + 0.99 \times 1 + 0.99^2 \times 0 + 0.99^3 \times 0 = 1.99$

(0,0,1,1)：$G_0 = 0 + 0.99 \times 0 + 0.99^2 \times 1 + 0.99^3 \times 1 = 1.9504$

反馈值反映了某个时刻后所得到累加奖励，当衰退系数小于1时，越是遥远的未来对累加反馈的贡献越少。

-----越早拿到奖励越好.



马尔可夫奖励过程(MRP:Markov Reward Process)

使用离散马尔可夫过程描述机器人移动问题

- 随机变量序列$\{S_t\}_{t=0,1,2,\cdots}$：S_t表示机器人第t步的位置，每个随机变量S_t 的取值范围为$S = \{s_1, s_2, \cdots, s_9, s_d\}$。
- 状态转移概率：$Pr(S_{t+1}|S_t)$满足马尔可夫性。

⇦ 马尔可夫过程

- 定义奖励函数$R(S_t, S_{t+1})$：从St到St+1所获得奖励， 其取值如图中所示：
- 定义衰退系数：$\gamma \in [0, 1]$(限制Gt值是有限的)。

奖励

-1

0	0	1
0	0	0
0	0	0

-1

综合以上信息， 可用$MRP = \{S, Pr, R, \gamma\}$来刻画马尔可夫奖励过程。

注意:这个模型不能体现机器人能动性，仍然缺乏与环境进行进行交互的手段。–动作?

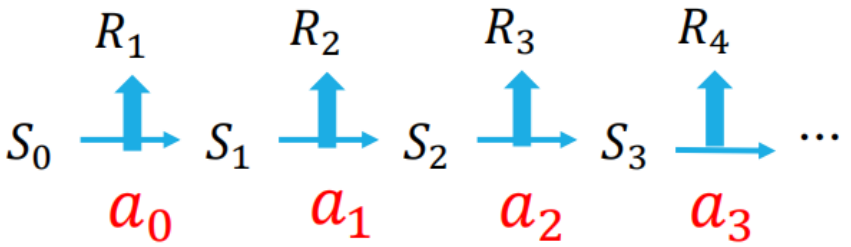
马尔可夫决策过程 (MDP:Markov Decision Process)：引入动作

在强化学习问题中，智能主体与环境交互过程中可自主决定所采取的动作，不同**动作**会对环境产生不同影响，为此：

- 定义智能主体能够采取的动作集合为A。
- 由于不同的动作对环境造成的影响不同， 因此状态转移概率定义为$Pr(S_{t+1}|S_t, a_t)$，其中$a_t \in A$为第t步采取的**动作**。
- 奖励可能受动作的影响，因此修改奖励函数为$R(S_t, a_t, S_{t+1})$。

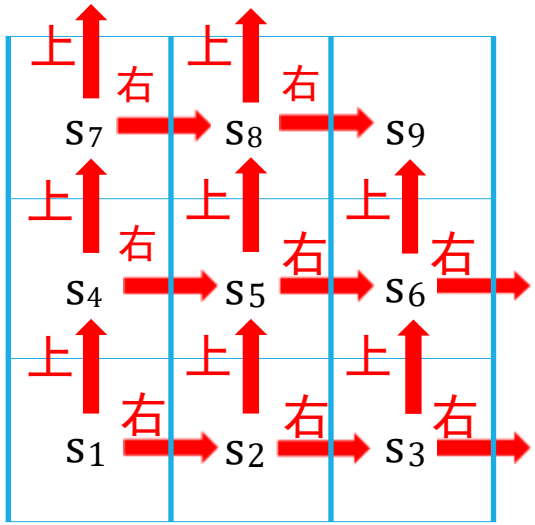
立即奖励，和期望奖励相对。

- 动作集合A可以是有限的，也可以是无限的。
- 状态转移可是确定（Deterministic）的，也可以是随机概率性（Stochastic）的。
- 确定状态转移相当于发生从S_t到S_{t+1}的转移概率为1。



马尔可夫决策过程 (Markov Decision Process)

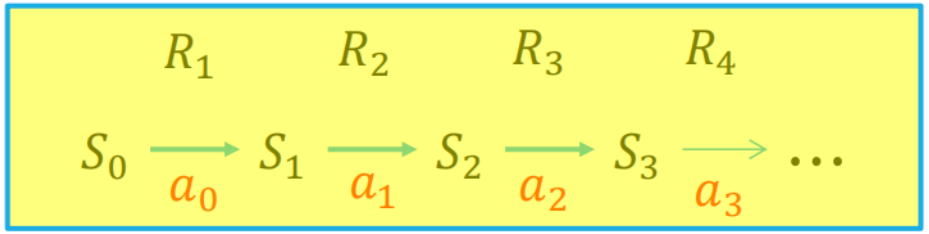
- 使用离散马尔可夫过程描述机器人移动问题：
- 随机变量序列$\{S_t\}_{t=0,1,2,\cdots}$：S_t表示机器人第t步所在位置（即**状态**），每个随机变量S_t的取值范围为$S=\{s_1,s_2,\cdots,s_9,s_d\}$。
- 动作集合：$A=\{$上，右$\}$。
- 状态转移概率$Pr(S_{t+1}|S_t,a_t)$：满足马尔可夫性，其中$a_t \in A$。状态转移如图所示。
- 奖励函数：$R(S_t,a_t,S_{t+1})$。
- 衰退系数：$\gamma \in [0,1]$。



综合以上信息，可通过$MDP=\{S,A,Pr,R,\gamma\}$来刻画**马尔可夫决策过程**。

马尔可夫决策过程 (Markov Decision Process)

- 马尔可夫决策过程的五元组$MDP = \{S, A, Pr, R, \gamma\}$是刻画强化学习中环境的标准形式。
- 马尔可夫决策过程可用如下序列来表示：



马尔可夫过程中产生的状态序列称为轨迹(Trajectory)，可如下表示：

$$(S_0, a_0, R_1, S_1, a_1, R_2, \cdots, S_T)$$

轨迹长度可以是无限的，也可以有终止状态S_T。有终止状态的问题叫做分段的(即存在回合的)(Episodic)，否则叫做持续的(Continuing)。

分段问题中，一个从初始状态到终止状态的完整轨迹称为一个片段或回合(Episode)。如围棋对弈中一个胜败对局为一个回合。

马尔可夫决策过程 (Markov Decision Process)

在机器人移动问题中：状态、行为、衰退系数、起始/终止状态、反馈、状态转移概率矩阵的定义如下：

$S = \{s_1, s_2, \cdots, s_9, s_d\}$

$A = \{\text{上}, \text{右}\}$

$\gamma = 0.99$

起始状态：$S_0 = s_1$

终止状态:$S_T \in \{s_9, s_d\}$

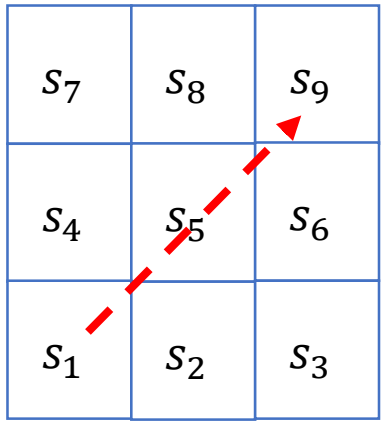
$$R(S_t, a_t, S_{t+1}) = \begin{cases} 1, \text{如果} S_{t+1} = s_9 \\ -1, \text{如果} S_{t+1} = s_d \\ 0, \text{其他情况} \end{cases}$$

$Pr(S_{t+1}|S_t, a_t = \text{右})$

S_t \ S_{t+1}	s_1	s_2	s_3	...	s_9	s_d
s_1	0	1	0	...	0	0
s_2	0	0	1	...	0	0
s_3	0	0	0	...	0	1
	...			...		...
s_8	0	0	0	...	1	0
s_9	0	0	0	...	0	1

$Pr(S_{t+1}|S_t, a_t = \text{上})$

S_t \ S_{t+1}	s_1	s_4	s_7	...	s_9	s_d
s_1	0	1	0	...	0	0
s_4	0	0	1	...	0	0
s_7	0	0	0	...	0	1
	...			...		...
s_6	0	0	0	...	1	0
s_9	0	0	0	...	0	1



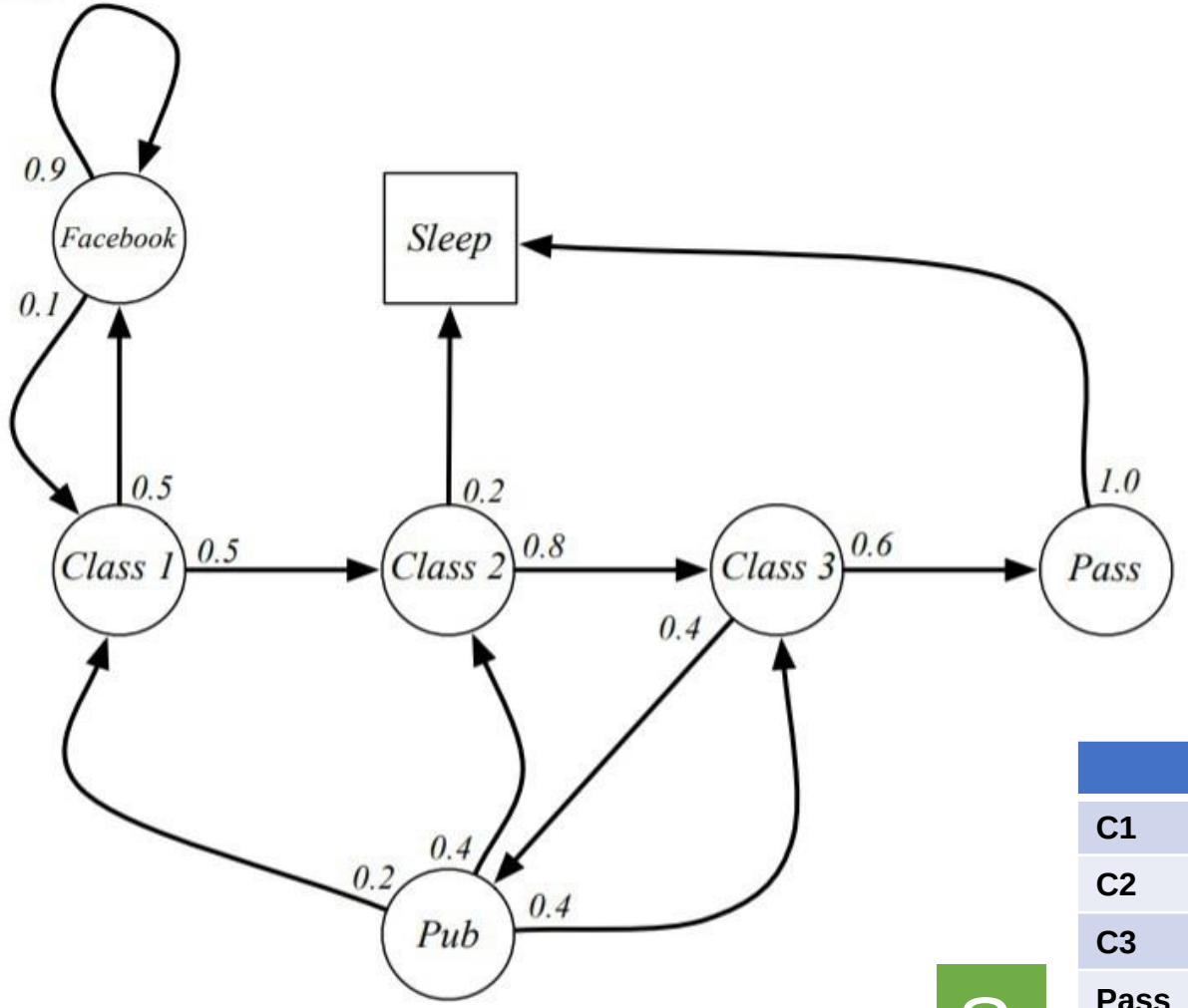
如何从起始状态到终止状态?

机器人寻路问题的状态转移函数

一个动作一个矩阵:一共A个9*10的矩阵

练习：画出学生马尔可夫链的状态转移函数

学生马尔可夫链，状态-状态（不包括动作a）

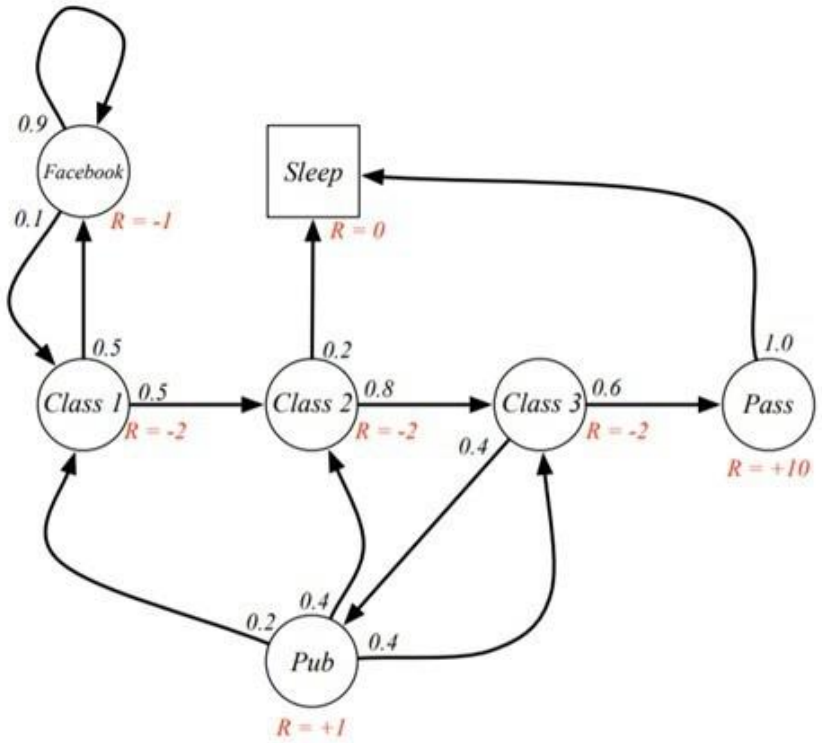


S_{t+1}

S_t

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

练习：求出学生马尔可夫链在s1=Class1状态时的价值函数V(s1)



Sample returns for Student MRP:
Starting from $S_1 = \text{C1}$ with $\gamma = \frac{1}{2}$

$$G_1 = R_2 + \gamma R_3 + \ldots + \gamma^{T-2} R_T$$

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

求不同轨迹下的V

累计回报G_t，智能体和环境一次交互过程收到的累计奖励

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

状态值函数$V(s)$

$$V(s) = E[G_t \mid S_t = s]$$

马尔可夫决策过程 (Markov Decision Process)中的策略学习

马尔可夫决策过程$MDP = \{S, A, Pr, R, \gamma\}$对环境进行了描述，智能主体如何与环境交互而完成

任务：进行策略学习

对环境中各种因素的说明	
已知的：S, A, R, γ	不一定已知的：Pr
观察到的：$(S_0, a_0, R_1, S_1, a_1, R_2, \cdots, S_T)$	

策略函数：

- 策略函数$\pi: S \times A \mapsto [0, 1]$，其中$\pi(s, a)$的值表示在状态$s$下采取动作$a$的概率。
- 策略函数的输出可以是确定的，即给定s情况下，只有一个动作a使得概率$\pi(s, a)$取值为1。

  对于确定性策略（又叫纯策略，与之对应的是随机策略、混合策略。），记为$a = \pi(s)$。

马尔可夫决策过程 (Markov Decision Process)中的策略学习

如何进行策略学习：一个好的策略是在当前状态下采取了一个行动后，该行动能够在未来收到最大化的反馈：$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots$。为了对策略函数$\pi$进行评估，定义：

马尔可夫性

- **价值函数（Value Function）**$V: S \mapsto \mathbb{R}$，其中$V_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s]$，即在第$t$步状态为$s$时，按照策略$\pi$行动后在未来所获得反馈值的期望。价值函数衡量了**某个状态的好坏程度**，反映了智能体从当前状态转移到该状态时能够为目标完成带来多大“好处”。

动作-价值函数又叫Q函数！

- **动作-价值函数(Action-Value Function)** $q: S \times A \mapsto \mathbb{R}$，其中$q_\pi(s,a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$表示在第$t$步状态为$s$时，按照策略$\pi$采取动作$a$后，在未来所获得反馈值的期望。

这样，策略学习转换为如下优化问题：

寻找一个最优策略π^*，对任意$s \in S$使得$V_{\pi^*}(s)$值最大。

一个好的策略函数应该能够使得智能体在采取了一系列行动后可获得最佳奖励。

价值函数与动作-价值函数的关系：对策略进行评估

定义：给定一个马尔可夫决策过程$MDP = (S, A, P, R, \gamma)$，学习一个最优策略$\pi^*$，对任意$s \in S$使得$V_{\pi^*}(s)$值最大。一个好的策略函数应该能够使得智能体在采取了一系列行动后可获得最佳奖励。

$$V_\pi(s) = \sum_{a \in A} \pi(s,a) q_\pi(s,a)$$

$$q_\pi(s,a) = \sum_{s' \in S} P(s'|s,a)\left[R(s,a,s') + \gamma V_\pi(s')\right]$$

$$\begin{aligned} V_\pi(s) &= \sum_{a \in A} \pi(s,a) \sum_{s' \in S} P(s'|s,a)\left[R(s,a,s') + \gamma V_\pi(s')\right] \\ &= \mathbb{E}_{a \sim \pi(s,\cdot)}\, \mathbb{E}_{s' \sim P(\cdot|s,a)}\left[R(s,a,s') + \gamma V_\pi(s')\right] \end{aligned}$$

- 价值函数和动作-价值函数反映了智能体在某一策略下所对应状态序列获得回报的期望，它比回报本身更加准确地刻画了智能体的目标。
- 价值函数和动作-价值函数的定义之所以能够成立，离不开决策过程所具有的马尔可夫性，即当位于当前状态s时，无论当前时刻t的取值是多少，一个策略回报值的期望是一定的（当前状态只与前一状态有关，与时间无关）。

如何求取价值函数和动作-价值函数？

Bellman方程的推导

马尔可夫决策过程

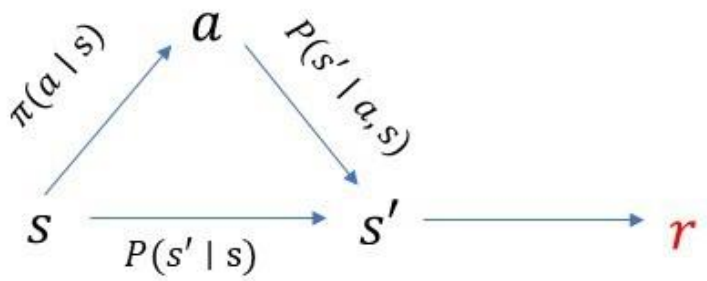
贝尔曼方程（Bellman Equation）：也被称作动态规划方程（Dynamic Programming Equation），由理查德·贝尔曼（Richard Bellman）提出。

$$s_0, a_0, s_1, r_1, a_1, s_2, r_2 \dots s_{t-1}, r_{t-1}, a_{t-1}, s_t, r_t \dots,$$

三个概率

- $\pi(a \mid s) = P(a \mid s)$：表示从状态$s$采取动作$a$的概率，也称策略
- $P(s' \mid a, s)$：表示在状态s下采取动作a后转移到状态s'的概率
- $P(s' \mid s)$：表示从状态s下转移到状态s'的概率

$$p(s'|s) = \sum_{a \in A} \pi(a|s) p(s'|a, s)$$ 全概率公式 $$\Longleftrightarrow P(A) = \sum_{i=1}^{n} P(B_i) P(A|B_i)$$



一组序列为(s, a, s', r)，从状态s转移到状态s'后才能计算此时的回报。
也可以写成$r_t = r(s_{t-1}, a_{t-1}, s_t)$

Bellman方程的推导

一个递归的过程：

$$
\begin{aligned}
v(s) &= \mathbb{E}[G_t \mid S_t = s] \\
&= \mathbb{E}\left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots \mid S_t = s\right] \\
&= \mathbb{E}\left[R_{t+1} + \gamma\left(R_{t+2} + \gamma R_{t+3} + \ldots\right) \mid S_t = s\right] \\
&= E[R_{t+1} + \gamma G_{t+1} | S_t = s] \\
&= E[R_{t+1} | S_t = s] + \gamma E[G_{t+1} | S_t = s]
\end{aligned}
$$

Bellman Equation

$$V(s) = E[R_{t+1} + \gamma V(S_{t+1}) \mid S_t = s]$$

也就是说，前一个状态的$V(s)$**的值通过下一个状态**$V(s^{'})$（也可写成$V(S_{t+1})$）的值和该过程得到的奖励函数R_{t+1}来更新。

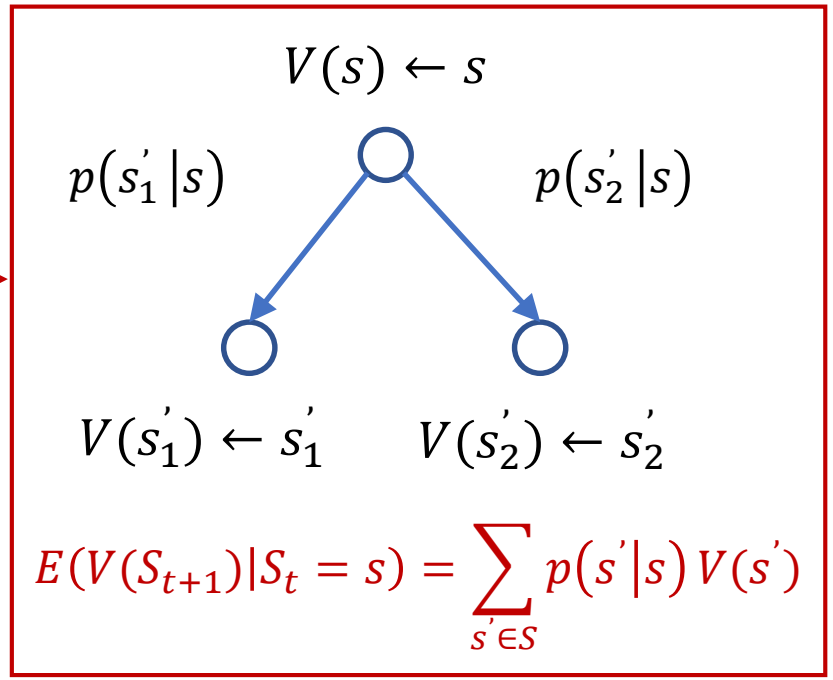
$$V(s^{'}) = V(S_{t+1})$$

$$V(s) = R_s + \gamma \sum_{s' \in S} P(s' \mid s)\, V(s')$$

全概率公式

$$R_s = E[R_{t+1} \mid S_t = s]$$

$$V(s) = R_s + \gamma \sum_{s^{'} \in S} p(s^{'}|s) V(s^{'}) = \sum_{s^{'} \in S} p(s^{'}|s)\left(R(s^{'}|s) + \gamma V(s^{'})\right)$$



Bellman方程的推导

求解V值相当于解方程

The Bellman equation can be expressed concisely using matrices,

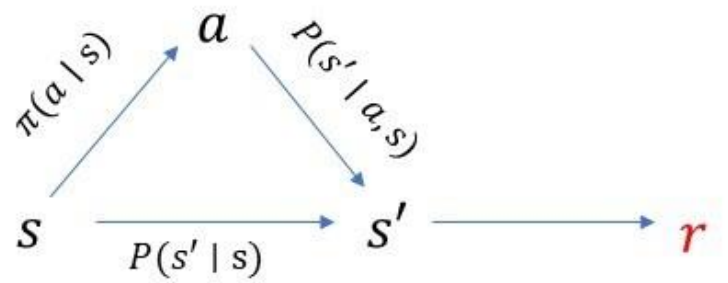
$$v = \mathcal{R} + \gamma \mathcal{P} v$$

where v is a column vector with one entry per state

$$\begin{bmatrix} v(1) \\ \vdots \\ v(n) \end{bmatrix} = \begin{bmatrix} \mathcal{R}_1 \\ \vdots \\ \mathcal{R}_n \end{bmatrix} + \gamma \begin{bmatrix} \mathcal{P}_{11} & \dots & \mathcal{P}_{1n} \\ \vdots & & \\ \mathcal{P}_{11} & \dots & \mathcal{P}_{nn} \end{bmatrix} \begin{bmatrix} v(1) \\ \vdots \\ v(n) \end{bmatrix}$$

- 用解线性方程组的方法，只适合小规模的问题；
- 大规模问题用迭代方法，包括：动态规划、时序差分学习、蒙特卡洛估计等。

Bellman方程的推导



马尔可夫奖励过程中引入行为a行为后构成马尔可夫决策过程，如果我们把行为a当成s和$s^{'}$的中间状态（分步进行），依然可以将马尔可夫决策过程理解成马尔可夫奖励过程，模型依然可以看成按照马尔可夫链的方式驱动。

$$p(s^{'}|s)=\sum_{a\in A}\pi(a|s)p(s^{'}|a,s)$$

遍历所有动作

类似于全概率公式

- $\pi(a\mid s)=P(a\mid s)$：表示从状态s采取动作a的概率，也称策略
- $P(s'\mid a,s)$：表示在状态s下采取动作a后转移到状态s'的概率
- $P(s'\mid s)$：表示从状态s下转移到状态s'的概率

在增加了策略后，状态转移概率的计算公式为(计算每一个动作的策略概率再求和):

$$P_{\pi}(s'\mid s)=\sum_{a\in A}\pi(a\mid s)P(s'\mid a,s)=\pi(a_1\mid s)P(s'\mid a_1,s)+\pi(a_2\mid s)P(s'\mid a_2,s)+\cdots$$

在增加了策略后，奖励的计算公式为(计算每一个策略下的奖励，即期望):

$$R_s^{\pi}=\sum_{a\in A}\pi(a|s)R_s^a\,,\qquad R_s^a=E(R_{t+1}|S_t=s\,,A_t=a)=\sum_{s^{'}\in S}p(s^{'}|a,s)R(s^{'}|a,s)$$

遍历所有状态

价值函数与动作-价值函数的关系

$$V_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s]$$

$$= \mathbb{E}_{a\sim\pi(s,\cdot)}\left[\mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s, A_t = a]\right]$$

$$= \sum_{a\in A} \underbrace{\pi(s,a)}_{\text{采取动作}a\text{的概率}}$$

$$V_\pi(s) = \sum_{a\in A} \pi(s,a) q_\pi(s,a)$$

书上式7.3

可以用动作-价值函数来表达价值函数，即从状态s出发、采用策略π完成任务所得回报期望可如下计算：在状态s可采取的每个动作的概率值与采取这一动作而获得价值的乘积之和（即期望）。这里$\pi(s,a)$表示在状态s下采取动作a的概率、$q_\pi(s,a)$为在状态s采取动作a后的回报期望。也就是说，状态s的价值可用该状态下可采取所有动作而取得的期望价值来表述。

价值函数与动作-价值函数的关系

$$q_\pi(s,a) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s, A_t = a]$$
$$= \mathbb{E}_{s' \sim P(\cdot|s,a)}\left[R(s,a,s') + \gamma \mathbb{E}_\pi[R_{t+2} + \gamma R_{t+3} + \cdots | S_{t+1} = s']\right]$$
$$= \sum_{s' \in S} \underbrace{P(s'|s,a)}_{\text{在状态}s\text{采取行动}a\text{进入状态}s'\text{的概率}} \times [\underbrace{R(s,a,s')}_{\text{在}s\text{采取}a\text{进入}s'\text{得到的回报}} + \gamma \times \underbrace{V_\pi(s')}_{\text{在}s'\text{获得的回报期望}}]$$

$$q_\pi(s,a) = \sum_{s' \in S} P(s'|s,a)\,[R(s,a,s') + \gamma V_\pi(s')] \quad \text{书上式7.4}$$

可知：可用价值函数来表示动作-价值函数，即在状态s采取动作a所取得价值如下计算：采取某个具体动作a进入状态s'的概率，乘以进入后续状态s'所得回报$R(s,a,s')$与后续状态s'价值函数的折扣值之和，然后对在状态s采取动作a后所进入全部状态的如上取值进行累加。这里$P(s'|s,a)$表示在状态s下采取动作a转移到状态s'的概率。也就是说，在某个状态下执行某一个动作所取得价值可以通过执行该动作之后进入的所有状态获得的瞬时奖励和后续状态可取得价值的期望来表示。

$$V_\pi(s) = \sum_{a\in A}\pi(s,a)q_\pi(s,a)$$

$$q_\pi(s,a) = \sum_{s'\in \boldsymbol{S}} P(s'|s,a)\left[R(s,a,s') + \gamma V_\pi(s')\right]$$

价值函数的贝尔曼方程

$$\begin{aligned} V_\pi(s) &= \sum_{a\in A}\pi(s,a)\sum_{s'\in \boldsymbol{S}} P(s'|s,a)\left[R(s,a,s') + \gamma V_\pi(s')\right] \\ &= \mathbb{E}_{a\sim\pi(s,\cdot)}\,\mathbb{E}_{s'\sim P(\cdot|s,a)}\left[R(s,a,s') + \gamma V_\pi(s')\right] \end{aligned}$$

从这个公式可知，价值函数取值与时间没有关系，只与策略π、在策略π下从某个状态转移到其后续状态所取得的回报以及在后续所得回报有关。

进一步分析价值函数的贝尔曼方程，可见状态s可获得“好处”$V_\pi(s)$由两个部分构成：一个是在状态s执行当前动作所得到的瞬时奖励，另外一个是在后续状态所能得回报期望（价值）的折扣值。该式中出现了期望，是因为在状态s可以一定概率进入到多个后续状态，且从状态s进入某个后续状态均会有不同的价值。

动作-价值函数中的回报与价值函数中的回报值不一样，动作-价值函数中的回报是在某个状态执行完某个具体动作之后取得回报的期望值，而价值函数中的回报值是在某个状态下选择所有动作执行后所得回报的期望值。

$$V_\pi(s) = \sum_{a\in A} \pi(s,a) q_\pi(s,a)$$

$$q_\pi(s,a) = \sum_{s'\in \boldsymbol{S}} P(s'|s,a)\left[R(s,a,s') + \gamma V_\pi(s')\right]$$

$$\begin{aligned} V_\pi(s) &= \sum_{a\in A} \pi(s,a) \sum_{s'\in \boldsymbol{S}} P(s'|s,a)\left[R(s,a,s') + \gamma V_\pi(s')\right] \\ &= \mathbb{E}_{a\sim\pi(s,\cdot)}\, \mathbb{E}_{s'\sim P(\cdot|s,a)}\left[R(s,a,s') + \gamma V_\pi(s')\right] \end{aligned}$$

- 由于在每个状态都有多种动作可供智能体选择，在每个状态施以某个具体动作后可按照一定概率转移到一个新的后续状态，因此相比于关心每个状态的价值，显然关心在当前状态下施以某个动作带来的价值更加直观实用。
- 如果智能体已经知道在某个状态下所有可供选择动作带来的不同价值，那么智能体在当前状态就应该去选择能带来最大价值的对应动作去执行，这就是设计动作-价值函数的初衷。

$$V_\pi(s) = \sum_{a\in A}\pi(s,a)q_\pi(s,a)$$

$$q_\pi(s,a) = \sum_{s'\in \boldsymbol{S}} P(s'|s,a)\left[R(s,a,s') + \gamma V_\pi(s')\right]$$

动作-价值函数的贝尔曼方程

$$\begin{aligned} & q_\pi(s,a) \\ &= \sum_{s'\in \boldsymbol{S}} P(s'|s,a)\left[R(s,a,s') + \gamma \sum_{a'\in A}\pi(s',a')q_\pi(s',a')\right] \\ &= \mathbb{E}_{s'\sim P(\cdot|s,a)}[R(s,a,s') + \gamma \mathbb{E}_{a'\sim\pi(s',\cdot)}[q_\pi(s',a')]] \end{aligned}$$

这个公式说明动作-价值函数取值同样与时间没有关系，而是与瞬时奖励和下一步的状态和动作有关。动作-价值函数表示在状态s采取了动作a后获得的“好处”，这也可以分为两部分：一是从状态s采取动作a后带来的瞬时奖励，二是进入后续状态后根据当前策略选择动作所得期望回报的折扣值。

贝尔曼方程(Bellman Equation)

价值函数的贝尔曼方程 $$V_\pi(s) = \mathbb{E}_{a\sim\pi(s,\cdot)}\,\mathbb{E}_{s'\sim P(\cdot|s,a)}[R(s,a,s') + \gamma V_\pi(s')]$$

动作-价值函数的贝尔曼方程 $$q_\pi(s,a) = \mathbb{E}_{s'\sim P(\cdot|s,a)}[R(s,a,s') + \gamma \mathbb{E}_{a'\sim\pi(s',\cdot)}[q_\pi(s',a')]]$$

价值函数取值与时间没有关系，只与策略π、在策略π下从某个状态转移到其后续状态所取得的立即回报以及在后续所得价值有关。

动作-价值函数取值同样与时间没有关系，而是与立即回报和下一步的状态和动作、后续所得动作价值有关。

- 贝尔曼方程描述了价值函数或动作-价值函数的递归关系，是研究强化学习问题的重要手段。其中**价值函数的贝尔曼方程描述了当前状态价值函数和其后续状态价值函数之间的关系**，即当前状态价值函数等于瞬时奖励（立即回报）的期望加上后续状态的（折扣）价值函数的期望。而**动作-价值函数（Q函数）的贝尔曼方程描述了当前动作-价值函数和其后续动作-价值函数之间的关系**，即当前状态下的动作-价值函数等于瞬时奖励的期望加上后续状态的（折扣）动作-价值函数的期望。

贝尔曼方程(Bellman Equation)

价值函数的贝尔曼方程 $$V_\pi(s) = \mathbb{E}_{a\sim\pi(s,\cdot)}\,\mathbb{E}_{s'\sim P(\cdot|s,a)}[R(s,a,s') + \gamma V_\pi(s')]$$

动作-价值函数的贝尔曼方程 $$q_\pi(s,a) = \mathbb{E}_{s'\sim P(\cdot|s,a)}[R(s,a,s') + \gamma \mathbb{E}_{a'\sim\pi(s',\cdot)}[q_\pi(s',a')]]$$

- 在实际中，需要计算得到最优策略以指导智能体在当前状态如何选择一个可获得最大回报的动作。求解最优策略的一种方法就是去**求解最优的价值函数或最优的动作-价值函数（即基于价值方法，Value-based Approach）**。一旦找到了最优的价值函数或动作-价值函数，自然而然也就是找到最优策略。当然，**在强化学习中还有基于策略（Policy-Based）和基于模型（Model-Based）等不同方法。**

利用贝尔曼方程求取价值函数和动作-价值函数，实现策略评估，进而进行策略优化。

价值函数与动作-价值函数的关系：对策略进行评估

$$V_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s]$$

$$= \mathbb{E}_{a\sim\pi(s,\cdot)}[\mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s, A_t = a]]$$

全期望公式

状态s下，采取动作a的概率。

$$= \sum_{a\in A} \pi(s,a) q_\pi(s,a)$$

状态s下，采取动作a后带来的价值期望。

$$q_\pi(s,a) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s, A_t = a]$$

$$= \mathbb{E}_{s'\sim Pr(\cdot|s,a)}\left[R(s,a,s') + \gamma \mathbb{E}_\pi[R_{t+2} + \gamma R_{t+3} + \cdots | S_{t+1} = s']\right]$$

状态s下，采取动作a后进入状态s'的概率。

状态s下，采取动作a后进入状态s'时获得的立即回报。

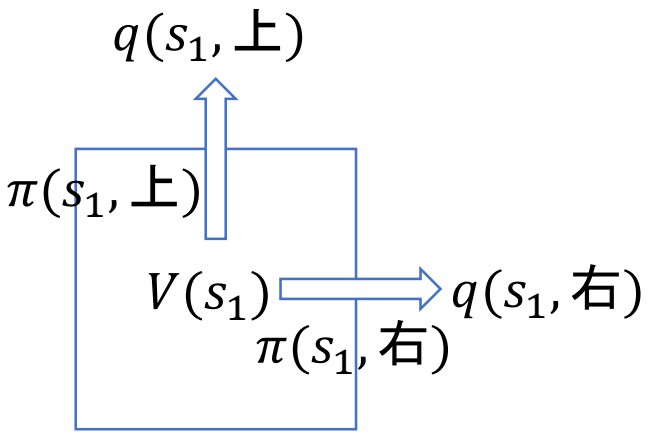
$$= \sum_{s'\in S} Pr(s'|s,a)\left[R(s,a,s') + \gamma V_\pi(s')\right]$$

状态s'下的价值期望。

价值函数与动作-价值函数的关系：以状态s_1 的计算为例

$$V_\pi(s) = \sum_{a \in A} \pi(s,a) q_\pi(s,a)$$

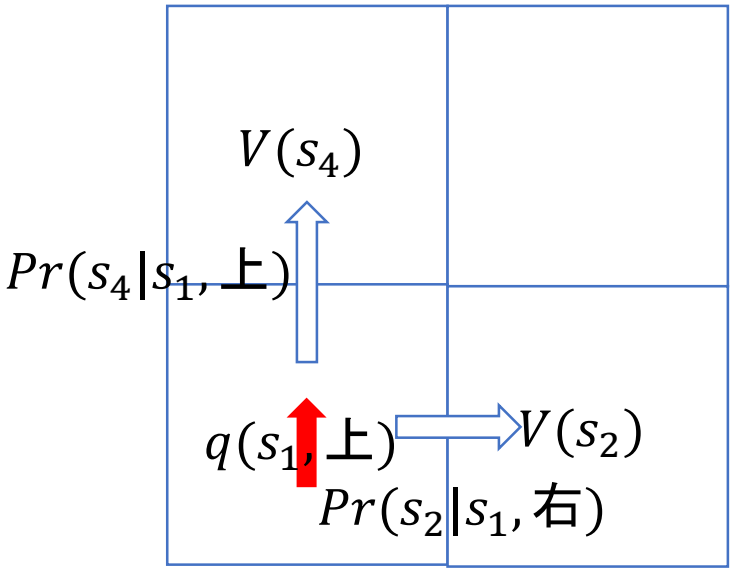
$$V_\pi(s_1) = \pi(s_1, 上) q_\pi(s_1, 上) + \pi(s_1, 右) q_\pi(s_1, 右)$$



不同动作下的反馈累加

$$q_\pi(s,a) = \sum_{s' \in S} Pr(s'|s,a) [R(s,a,s') + \gamma V_\pi(s')]$$

$$q_\pi(s_1, 上) = Pr(s_4|s_1, 上)[R(s_1, 上, s_4) + \gamma V_\pi(s_4)]$$



动作确定时状态转移后的反馈结果

最优价值函数与最优动作-价值函数

$V^*(s), Q^*(s,a)$

$V^*(s) = max\, V_\pi(s)$，对于所有的策略，找到一个策略使得V值函数最大

$Q^*(s,a) = max\, Q_\pi(s,a)$，对于所有的策略，找到一个策略使得Q值函数最大

一个很显然的方法是通过迭代的方式寻找最优策略，如果使得值函数最大，那么就选这个策略，把概率值设为1

$$\pi'(a|s) = \begin{cases} 1 & \text{if } a = \arg\max_a Q^\pi(s,\hat{a}) \\ 0 & \text{otherwise} \end{cases}$$

比如这个策略好，将其值设置为1
说明在分支中只选择这个动作后转移到其他状态。

在增加了策略后，状态转移概率的计算公式为(计算每一个动作的策略概率再求和):

$$R_s^\pi = \sum_{a\in A} \pi(a|s) R_s^a, \quad R_s^a = E(R_{t+1}|S_t = s, A_t = a) = \sum_{s'\in S} p(s'|a,s) R(s'|a,s)$$

立即回报

提 纲

1、 强化学习定义： 马尔可夫决策过程

2、 强化学习中的策略优化与策略评估

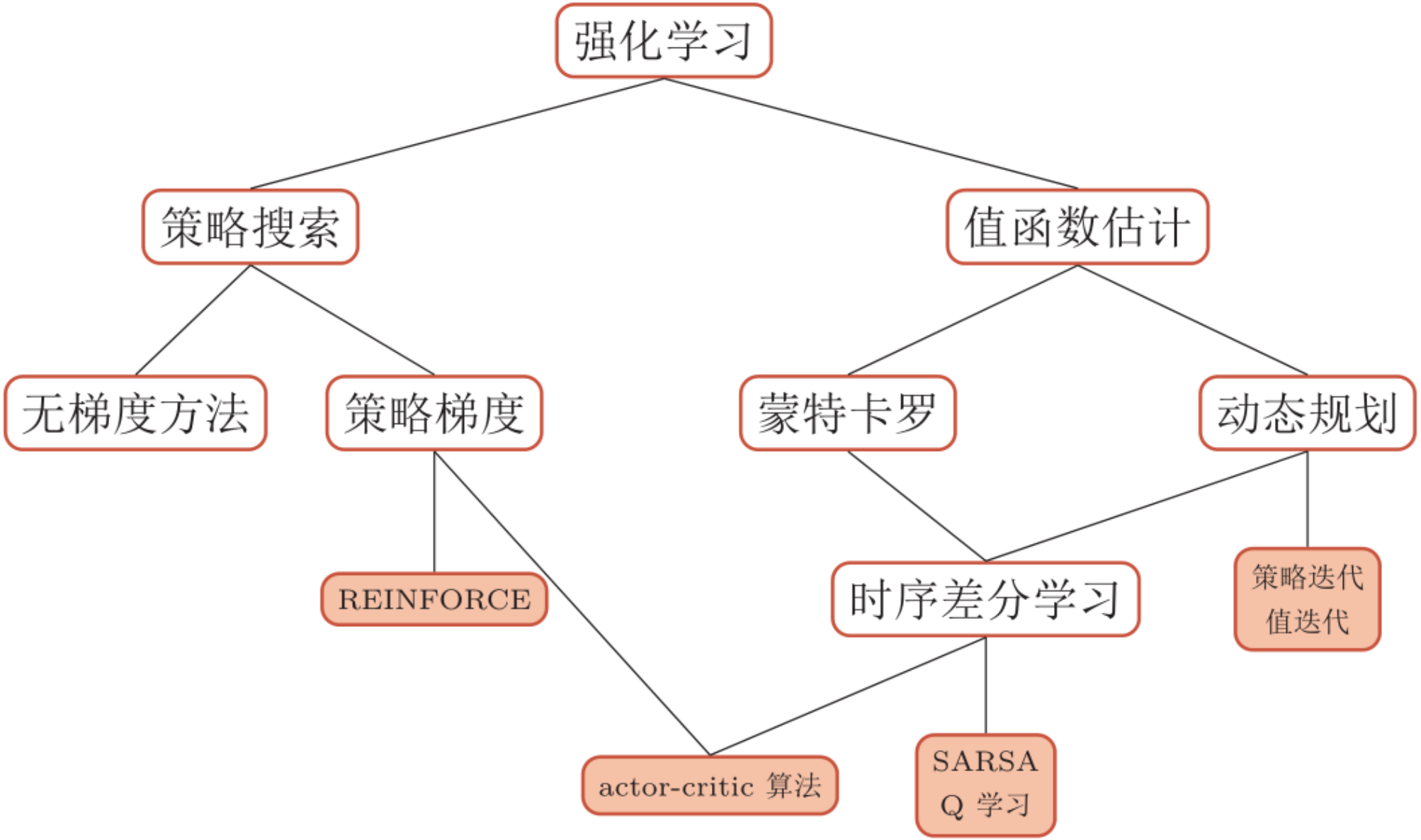
3、 强化学习求解：Q-Learning

4、 深度强化学习： 深度学习+强化学习

强化学习的求解方法

- **基于价值(Value-based)的方法**
  - 对价值函数进行建模和估计， 以此为依据制订策略。
- **基于策略(Policy-based)的方法**
  - 对策略函数直接进行建模和估计，优化策略函数使反馈最大化。
- **基于模型(Model-based)的方法**
  - 对环境的运作机制建模，然后进行规划 (Planning)等。

强化学习的求解方法

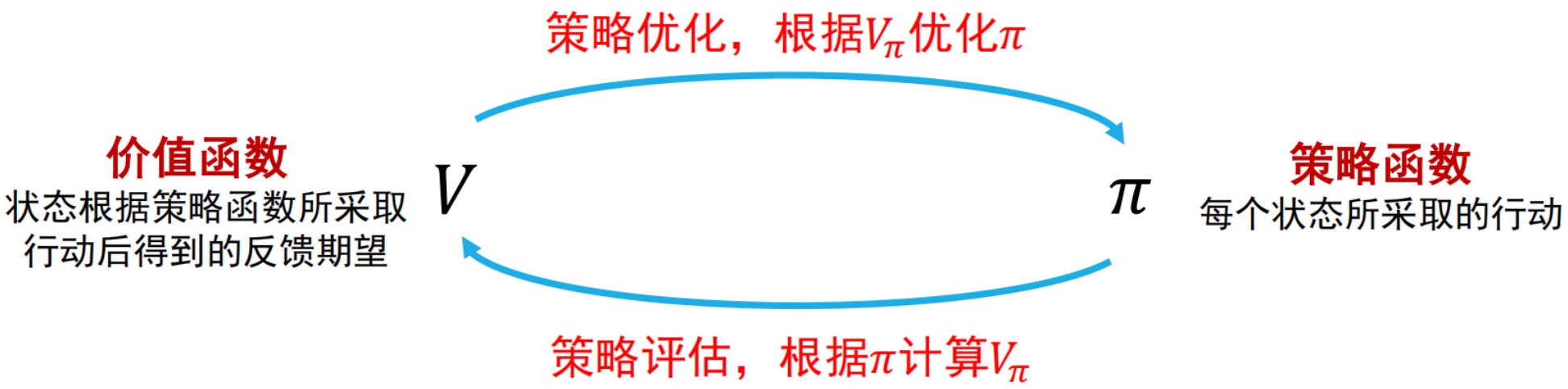

强化学习
策略搜索
值函数估计
无梯度方法
策略梯度
蒙特卡罗
动态规划
REINFORCE
时序差分学习
策略迭代
值迭代
actor-critic 算法
SARSA
Q 学习

强化学习的问题与求解

强化学习的问题定义：给定马尔可夫决策过程$MDP = \{S, A, Pr, R, \gamma\}$

寻找一个最优策略π^*，对任意$s \in S$使得$V_{\pi^*}(s)$值最大

强化学习求解：在策略优化和策略评估的交替迭代中优化参数



强化学习中的策略优化

策略优化定理：

对于确定的策略π和π'，如果对于任意状态$s \in S$

$$q_\pi\big(s,\pi'(s)\big) \geq q_\pi(s,\pi(s))$$

那么对于任意状态$s \in S$，有

$$V_{\pi'}(s) \geq V_\pi(s)$$

即策略π'不比π差

注意，不等式左侧的含义是只在当前这一步将动作修改为$\pi'(s)$，未来的动作仍然按照π的指导进行

在讨论如何优化策略之前，首先需要明确什么是“更好”的策略。分别给出π和π'两个策略，如果对于任意状态$s \in S$，有$V_\pi(s) \leq V_{\pi'}(s)$，那么可以认为策略$\pi'$不比策略$\pi$差，可见“更优”策略是一个偏序关系。

强化学习中的策略优化

给定当前策略π、价值函数V_π和行动-价值函数q_π时，可如下构造新的策略π'，

只要π'满足如下条件：

$$\pi'(s) = \text{argmax}_a q_\pi(s,a) \quad （对于任意s \in S）$$

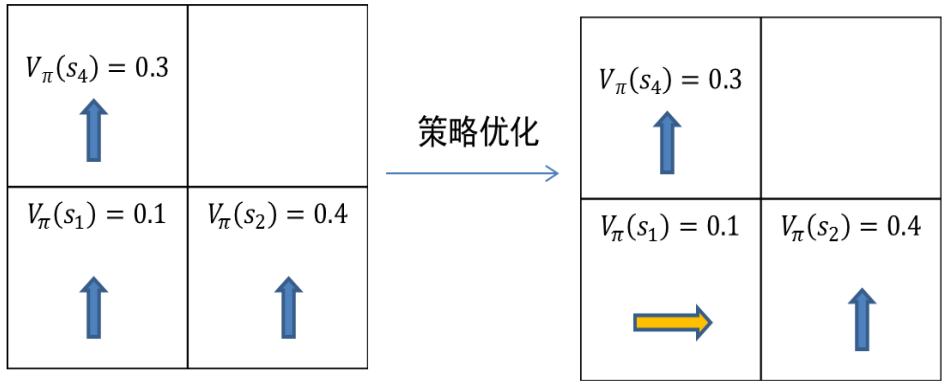
π'便是对π的一个改进。于是对于任意$s \in S$，有

$$q_\pi\big(s,\pi'(s)\big) = q_\pi\big(s,\text{argmax}_a\ q_\pi(s,a)\big)$$

$$= \max_a q_\pi(s,a)$$

$$\geq q_\pi(s,\pi(s))$$

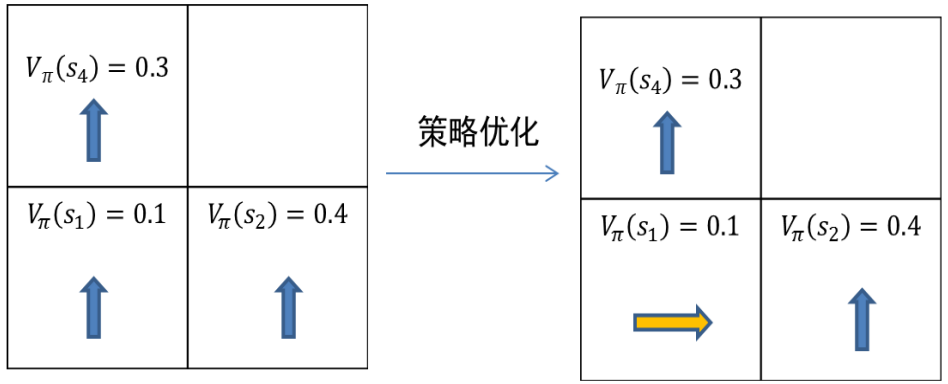
强化学习中的策略优化：机器人寻路问题为例子



原有策略　　优化后策略

左图给出了机器人寻路问题的原有策略及其对应价值函数。在左图中，根据原有策略，智能体位于状态s_1的价值函数取值为0.1。当智能体位于s_1状态时，智能体在原有策略指引下将选择向上移动一个方格的行动，从状态s_1进入状态s_4，状态s_4的价值函数取值为0.3。原有策略所给出的其他信息，可从左图周知。

强化学习中的策略优化：机器人寻路问题为例子

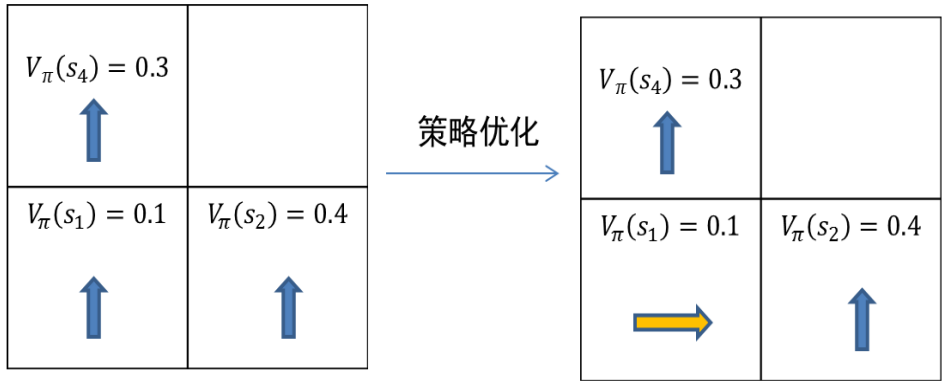


原有策略　　优化后策略

$$q_\pi(s_1, 上) = \sum_{s' \in S} \mathrm{P}(s'|s_1, 上)\,[R(s_1, 上, s') + \gamma V_\pi(s')]$$
$$= 1 \times (0 + 0.99 \times 0.3) + 0 \times \cdots = 0.297$$

下面来看智能体如何通过策略优化来改变在状态s_1所采取的行动。由于智能体在状态s_1能够采取“向上移动一个方格”或“向右移动一个方格”两个行动中的一个。首先计算状态s_1选择“向上移动一个方格”后所得动作-价值函数取值。

强化学习中的策略优化：机器人寻路问题为例子

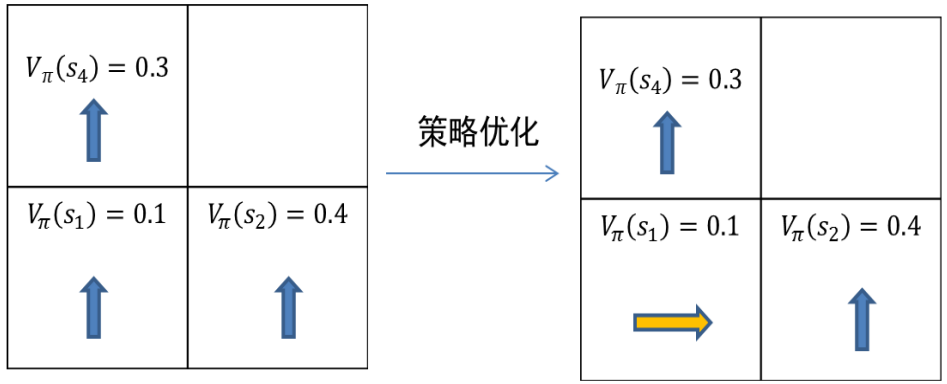


原有策略　　优化后策略

$$q_\pi(s_1, 右) = \sum_{s' \in S} \mathrm{P}(s'|s_1, 右)\,[R(s_1, 右, s') + \gamma V_\pi(s')]$$
$$= 1 \times (0 + 0.99 \times 0.4) + 0 \times \cdots = 0.396$$

接着计算状态s_1选择“向右移动一个方格”后所得动作-价值函数取值。

强化学习中的策略优化：机器人寻路问题为例子



原有策略　　优化后策略

$$q_\pi(s_1, 右) == 0.396 > q_\pi(s_1, 上) = 0.297$$

可见，智能体在状态s_1选择“向上移动一个方格”行动所得回报$q_\pi(s_1, 上)$值为0.297、选择“向右移动一个方格”行动所得回报$q_\pi(s_1, 右)$值为0.396。显然，智能体在状态s_1应该选择“向右移动一个方格”行动，这样能够获得更大的回报。于是，经过策略优化后，状态s_1处的新策略为$\pi'(s_1) = \text{argmax}_a q_\pi(s, a) = 右$，则将$s_1$处的策略从“上”更新为 “右”。其他状态的情况可用类似方法计算得到。

策略评估

通过迭代进行策略评估

- **动态规划**(基于贝尔曼方程)
- 蒙特卡洛采样
- 时序差分(Temporal Difference)

动态规划策略评估-基于贝尔曼方程

- 迭代方法：循环次数趋近于无穷，算法收敛。所有状态的价值函数初值可任意设置。

算法7.1

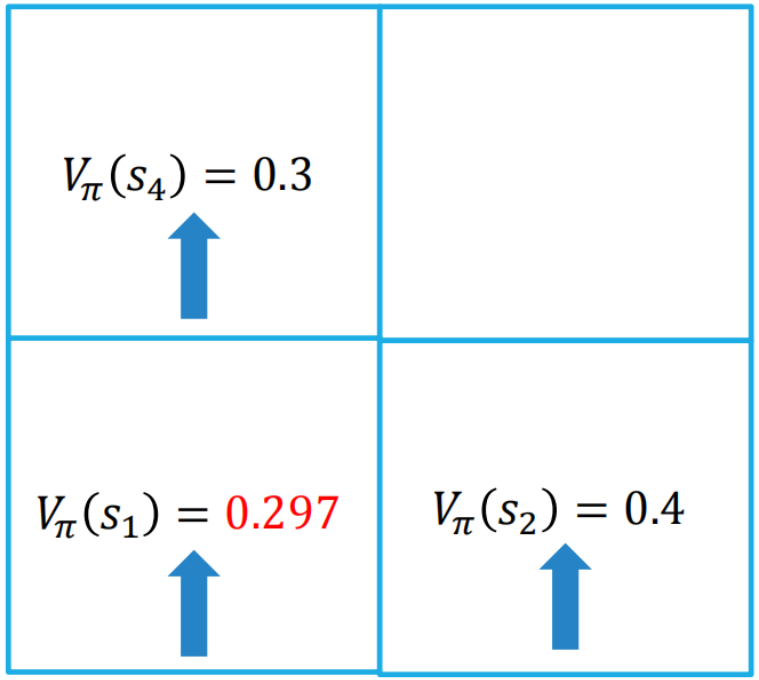
初始化V_π函数
循环
　　枚举$s \in S$
　　$V_\pi(s) \leftarrow \sum_{a \in A} \pi(s,a) \underbrace{\sum_{s' \in S} Pr(s'|s,a)\,[R(s,a,s') + \gamma V_\pi(s')]}_{q_\pi(s,a)}$

直到V_π收敛

更新$V_\pi(s_1)$的值：

$$q_\pi(s_1, 上) = 1 \times (0 + 0.99 \times 0.3) + 0 \times (0 + 0.99 \times 0.4) + \ldots = 0.297$$

$$V_\pi(s_1) = 1 \times q_\pi(s_1, 上) + 0 \times q_\pi(s_1, 右) = 0.297$$



动态规划法的缺点：1) 智能主体需要事先知道状态转移概率;2) 无法处理状态集合大小无限的情况

策略评估-基于蒙特卡罗采样

- 大数定理：当采样足够大时，样本的均值向期望收敛。

算法7.2 **基于蒙特卡洛采样的价值函数更新**

选择不同的起始状态，按照当前策略π采样若干轨迹，记它们的集合为D
枚举$s \in S$
　　计算D中s每次出现时对应的反馈$G_1, G_2, \cdots, G_k$
　　$V_\pi(s) \leftarrow \frac{1}{k}\sum_{i=1}^{k} G_i$

假设按照当前策略可样得到以下两条轨迹

$$(s_1, s_4, s_7, s_8, s_9)$$
$$(s_1, s_2, s_3, s_d)$$

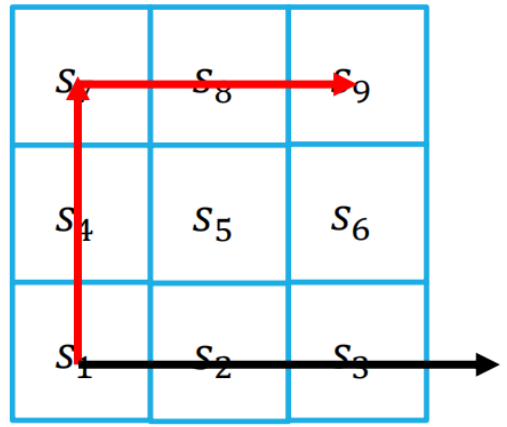
s_1对应的反馈值分别为

$$0 + \gamma \times 0 + \cdots + \gamma^3 \times 1 = 0.970$$
$$0 + \gamma \times 0 + \gamma^2 \times (-1) = -0.980$$

因此估计

$$V(s_1) = \frac{1}{2}(0.970 - 0.980) = -0.005$$

如果是确定的策略，每个起点只会产生一种轨迹



策略评估-基于蒙特卡罗采样

基于蒙特卡洛采样的优缺点

选择不同的起始状态，按照当前策略π采样若干轨迹，记它们的集合为D
枚举$s \in S$
　　计算D中s每次出现时对应的反馈$G_1, G_2, \cdots, G_k$
　　$V_\pi(s) \leftarrow \frac{1}{k}\sum_{i=1}^{k} G_i$

蒙特卡洛采样法的优点：

- 智能主体不必知道状态转移概率。
- 容易扩展到无限状态集合的问题中。

蒙特卡洛采样法的缺点：

- 状态集合比较大时，一个状态的轨迹可能非常稀疏，不利于估计期望。
- 在实际问题中，最终反馈需要在终止状态才能知晓，导致反馈周期较长。

策略评估-基于时序差分

- 结合蒙特卡罗采样和动态规划方法。

算法7.3 **基于时序差分（Temporal Difference）的价值函数更新**

初始化V_π函数
循环（**不断采样片段轨迹**）
　　初始化s为初始状态
　　循环（**不断采样后续状态**）
　　$a \sim \pi(s,\cdot)$
　　执行动作a，观察奖励R和下一个状态s'
　　更新$V_\pi(s) \leftarrow V_\pi(s) + \alpha[R(s,a,s') + \gamma V_\pi(s') - V_\pi(s)]$
　　$s \leftarrow s'$
　　直到s是终止状态
直到V_π收敛

贝尔曼方程

$$V_\pi(s) = \mathbb{E}_{a\sim\pi(s,\cdot)}\,\mathbb{E}_{s'\sim P(\cdot|s,a)}[R(s,a,s') + \gamma V_\pi(s')]$$

根据贝尔曼方程$V_\pi(s) = \mathbb{E}_{a\sim\pi(s,\cdot),s'\sim Pr(\cdot|s,a)}[R(s,a,s') + \gamma V_\pi(s')]$

利用蒙特卡洛采样的思想，通过采样a和s'来估计期望

$R(s,a,s') + \gamma V_\pi(s')$是对$V_\pi(s)$的一个估计值（一个样本值）

部分更新$V_\pi(s)$的值：$V_\pi(s) \leftarrow (1-\alpha)V_\pi(s) + \alpha[R(s,a,s') + \gamma V_\pi(s')]$

对新的估计值添加权重，使其更接近真实值

过去的价值函数值　　学习得到的价值函数值

策略评估-基于时序差分

基于时序差分（Temporal Difference）的价值函数更新

初始化V_π函数
循环
　　初始化s为初始状态
　　循环
　　　　$a \sim \pi(s,\cdot)$
　　　　执行动作a，观察奖励R和下一个状态s'
　　　　更新$V_\pi(s) \leftarrow V_\pi(s) + \alpha[R(s,a,s') + \gamma V_\pi(s') - V_\pi(s)]$
　　　　$s \leftarrow s'$
　　直到s是终止状态
直到V_π收敛

假设$\alpha = 0.5$，更新$V_\pi(s_1)$的值：
从$\pi(s_1,\cdot)$ 中采样得到动作$a =$ 上
从$Pr(\cdot\,|s_1, 上)$中采样得到下一步状态$s' = s_4$

$$V_\pi(s_1) \leftarrow V_\pi(s_1) + \alpha[R(s_1, 上, s_4) + \gamma V_\pi(s_4) - V_\pi(s_1)]$$
$$= 0.1 + 0.5 \times [0 + 0.99 \times 0.3 - 0.1] = 0.199$$