第3章

数学建模

马尔可夫决策过程(Markov Decision Process,MDP)是强化学习的数学理论基础。马尔可夫决策过程以概率形式对强化学习任务进行建模,并对强化学习过程中出现的状态、动作、状态转移概率和奖赏等概念进行抽象表达,从而实现对强化学习任务的求解,即得到最优策略,获得最大累计奖赏。

在线视频 03

3.1 马尔可夫决策过程

在介绍马尔可夫决策过程之前,先来简单介绍马尔可夫性质以及马尔可夫过程等概念。

马尔可夫性质:在某一任务中,如果 Agent 从环境中得到的下一状态仅依赖于当前状态,而不考虑历史状态,即 $P[S_{t+1}|S_t]=P[S_{t+1}|S_1,S_2,\cdots,S_t]$,那么该任务即满足马尔可夫性质。虽然从定义上来看只有当前状态和与其相邻的下一个状态具有关联性,但实际上当前状态蕴含了前面所有历史状态的信息,只不过在已知当前状态的情况下,可以丢弃其他所有的历史信息。然而,现实中的任务很多无法严格满足马尔可夫性质,为简化强化学习问题的求解过程,通常假设强化学习所需要解决的任务均满足马尔可夫性质。

马尔可夫过程(Markov Process,MP):由二元组$(\mathcal{S},\mathcal{S})$中的$(S_t,S_{t+1})$组成的马尔可夫链,该链中的所有状态都满足马尔可夫性质。

马尔可夫奖赏过程(Markov Reward Process,MRP):由三元组$(\mathcal{S},P,\mathcal{R})$组成的马尔可夫过程。根据概率,状态自发地进行转移,其状态转移概率 P 与动作无关,记为 $P[S_{t+1}=s'|S_t=s]$。

马尔可夫决策过程:由四元组$(\mathcal{S},\mathcal{A},P,\mathcal{R})$组成的马尔可夫过程,状态依靠动作进行转移。根据状态空间或动作空间是否有穷,马尔可夫决策过程分为**有穷马尔可夫决策过**

程(finite MDPs)和**无穷马尔可夫决策过程**(infinite MDPs)。其中四元组$(\mathcal{S},\mathcal{A},P,\mathcal{R})$定义如下。

1. 状态(state)或观测值(observation)

状态空间通常分为两种：用$\mathcal{S}$表示不包含终止状态的状态空间；用$\mathcal{S}^+$表示包含终止状态的状态空间。状态空间可以用集合来表示。用s表示状态空间中的某一状态，可分为离散状态和连续状态两种类型。

有的资料也将状态分为**环境状态**(environment state)、**Agent 状态**和**信息状态**(information state)，本书不对状态进行以上区分。另外将状态和观测值视为同一概念，Agent 从环境中获得观测值，而对 Agent 来说，观测值就是它的状态。

2. 动作(action)

$\mathcal{A}$表示动作空间，$\mathcal{A}(s)$表示状态s的动作空间。动作空间可以用集合来表示。用a表示动作空间中的某一个动作，可分为离散动作和连续动作两种类型。

3. 状态转移(state transition)函数

P表示状态转移概率，即在状态s下，执行动作a转移到s'的概率。可以表示为如下两种形式：

$$p(s',r|s,a)=P[S_{t+1}=s',R_{t+1}=r|S_t=s,A_t=a],\quad \sum_{s'}\sum_{r}p(s',r|s,a)=1 \tag{3.1}$$

$$p(s'|s,a)=P[S_{t+1}=s'|S_t=s,A_t=a],\quad \sum_{s'}p(s'|s,a)=1 \tag{3.2}$$

式(3.1)和式(3.2)都可用于表示 MDP 中的状态转移概率。式(3.1)既考虑到进入下一状态的随机性，又考虑了到达下一状态获得奖赏的随机性；而式(3.2)只体现出 Agent 执行动作后进入下一状态的随机性。

在强化学习中，根据状态转移的状态转移概率，可以将环境分为**确定环境**(deterministic environment)和**随机环境**(stochastic environment)。通常将$p(s',r|s,a)$恒为 1 的环境称为确定环境。也就是说，Agent 在状态s下执行动作a后，所到达的下一状态是唯一确定的。因此在确定环境下，执行动作a后，状态s可以映射为下一个确定的状态s'；除此之外其他类型的环境都称为随机环境。即 Agent 在状态s下执行动作a后，不一定能到达预期的下一状态s'，或者说到达下一状态存在多种可能性。随机环境通常使用式(3.1)或式(3.2)的形式来表达。可以看出，确定环境是随机环境的一个特例，即 Agent 在状态s下执行动作a后，以 1 的概率到达下一状态s'，以 0 的概率到达其他状态。

4. 奖赏(reward)函数

$\mathcal{R}$表示**奖赏空间**。$r(s'|s,a)$表示 Agent 在状态s下采取动作a到达状态s'时，所获得的**期望奖赏**(expected reward)。可分为离散奖赏和连续奖赏两种类型。其公式可表

示为

$$r(s,a,s')=\mathbb{E}[R_{t+1}|S_t=s,A_t=a,S_{t+1}=s']=\sum_r r\frac{p(s',r|s,a)}{p(s'|s,a)} \tag{3.3}$$

$$r(s,a)=\mathbb{E}[R_{t+1}|S_t=s,A_t=a]=\sum_r r\sum_{s'}p(s',r|s,a) \tag{3.4}$$

式(3.3)和式(3.4)都可以表示 MDP 中的奖赏。由于状态转移概率 $p(s',r|s,a)$的随机性，Agent 在状态 s 下多次执行动作 a 得到的奖赏的期望，才是真实的奖赏。但是如果遵循先获得奖赏，再到达下一状态这一思想，在实际计算过程中，r 通常都是确定的，此时 $r(s,a)$即可表示相应的奖赏。

对于奖赏，可以从两个方面进行理解。

(1) 先获得奖赏再进入下一状态：奖赏 R_{t+1} 与当前状态 S_t 和动作 A_t 相关。

(2) 先进入下一状态再获得奖赏：奖赏 R_{t+1} 与当前状态 S_t、动作 A_t 和下一状态 S_{t+1} 相关，这也是奖赏用 R_{t+1} 表示的一个重要原因，即与 $t+1$ 时刻到达的状态 S_{t+1} 有关。

例 3.1　确定环境下扫地机器人任务的 MDP 数学建模。

在线视频 04

考虑图 3.1 所描述的确定环境 MDP 问题：一个扫地机器人，在躲避障碍物的同时，需要到指定的位置收集垃圾，或者到指定位置给电池充电。

图 3.1　扫地机器人环境

扫地机器人任务的 MDP 数学建模如下。

1. 状态

在该问题中，状态 s 用来描述机器人所在的位置，可以用 2 维向量表示。为了简化问题，将状态空间离散化为 24 个不同的状态(图 3.1 中，共 25 个小方格，除去[3,3]障碍物位置外，将每个小方格作为一个状态)，按照图 3.1 中序号出现的先后顺序，用集合表示为

$$\mathcal{S}^+=\{S_0:[1,1],S_1:[2,1],S_2:[3,1],\cdots,S_{11}:[2,3],$$
$$S_{13}:[4,3],\cdots,S_{19}:[5,4],\cdots,S_{23}:[4,5],S_{24}:[5,5]\}$$

其中，充电桩所在的位置为状态 $S_0=[1,1]$，垃圾所在的位置为状态 $S_{19}=[5,4]$。坐标[3,3]表示障碍物的位置，机器人不会到达这里，因此不作为状态。通常状态 S_0 和 S_{19} 为终止状态或**吸收状态**(absorbing state)，即一旦机器人到达这两个状态之一，就不会再离开，**情节**(episode)即结束。

2. 动作

动作 a 用来描述机器人运动的方向，可以用 2 维向量表示。为了简化问题，将动作空间离散化为上、下、左、右 4 个不同的动作，用集合表示为

$$\mathcal{A}=\{上:[0,1],下:[0,-1],左:[-1,0],右:[1,0]\}$$

对于不同的状态 s，动作空间可能不同。例如，对于状态 $S_0=[1,1]$，它的动作空间为

$\mathcal{A}(S_0)=\{[0,1],[1,0]\}$，即机器人处于该状态时，只能采取向上、向右动作，但因为 $S_0=[1,1]$ 为吸收状态，无论采取哪个动作，都会保持原地不动；对于状态 $S_9=[5,2]$，它的动作空间为 $\mathcal{A}(S_9)=\{[0,1],[0,-1],[-1,0]\}$，即机器人处于该状态时，不能采取向右的动作；对于坐标[3,3]周围的 4 个状态，动作空间中的 4 个动作都可以采用，但采取指向障碍物的动作时，会保持原地不动。

3. 状态转移函数

在确定环境下，状态转移可以用两种方式来表示：既可以映射为下一个状态，又可以映射为到达下一个状态的概率。

映射为下一个状态：

$$f(s,a)=\begin{cases}s+a, & s\neq[1,1]\text{ 且 }s\neq[5,4]\text{ 且 }s+a\neq[3,3]\\ s, & \text{其他}\end{cases}\tag{3.5}$$

映射为到达下一个状态的概率：

$$p(s,a,s')=\begin{cases}1, & (s+a=s'\text{ 且 }s+a\neq[3,3])\text{ 或}((s=[1,1]\text{ 或 }s=[5,4]\text{ 或 }s+a=[3,3])\text{ 且 }s=s')\\ 0, & \text{其他}\end{cases}\tag{3.6}$$

4. 奖赏函数

到达状态 $S_{19}=[5,4]$，机器人可以捡到垃圾，并得到 +3 的奖赏；到达状态 $S_0=[1,1]$，机器人充电，并得到 +1 的奖赏；机器人采取动作向坐标[3,3]处移动时，会撞到障碍物，保持原地不动，并得到 −10 的奖赏；其他情况，奖赏均为 0。特别是当机器人到达吸收状态后，无论采取什么动作，只能得到 0 的奖赏。对应的奖赏函数为

$$r(s,a)=\begin{cases}+3, & s\neq[5,4]\text{ 且 }s+a=[5,4]\\ +1, & s\neq[1,1]\text{ 且 }s+a=[1,1]\\ -10, & s+a=[3,3]\\ 0, & \text{其他}\end{cases}\tag{3.7}$$


在线视频 05

例 3.2 随机环境下扫地机器人任务的 MDP 数学建模。

重新考虑例 3.1 的扫地机器人问题。假设由于地面的原因，采取某一动作后，状态转换不再确定。当采取某一动作试图向某一方向移动时，机器人成功移动的概率为 0.80，保持原地不动的概率为 0.15，移动到相反方向的概率为 0.05，具体如图 3.2 所示，其中 s' 表示下一个状态，s'' 表示相反方向的下一个状态。

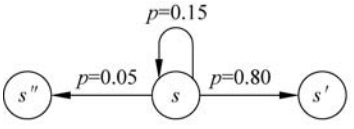

图 3.2 随机环境扫地机器人状态转移示意图

该问题在随机环境下，状态空间、动作空间与确定环境是完全相同的，其随机性主要体现在状态转移函数和奖赏函数中。根据任务的随机性，状态转移只能用概率来表示。具体的状态转移函数可以定义为

$$p(s,a,s')=\begin{cases}0.80, & s+a=s' \text{ 且 } s\neq s'\\ 0.15, & s=s' \text{ 且 } s\neq[1,1] \text{ 且 } s\neq[5,4]\\ 0.05, & s-a=s' \text{ 且 } s\neq s'\end{cases} \tag{3.8}$$

在随机环境下，奖赏的获取不单纯受(s,a)的影响，还与下一状态s'相关，具体的奖赏函数定义为

$$r(s,a,s')=\begin{cases}+3, & s\neq[5,4] \text{ 且 } s'=[5,4]\\ +1, & s\neq[1,1] \text{ 且 } s'=[1,1]\\ -10, & s+a=[3,3] \text{ 且 } s=s'\\ 0, & \text{其他}\end{cases} \tag{3.9}$$

3.2　基于模型与无模型

从 MDP 四元组中可以看出，在强化学习任务求解中，状态转移概率 p 是非常重要的。但获取如例 3.2 的状态转移概率难度非常大，需要在保证环境完全相同的情况下，重复大量的实验。从状态转移概率是否已知的角度，强化学习可以分为**基于模型**(model-based)强化学习和**无模型**(model-free)强化学习两种。

(1) 基于模型：状态转移概率 p 已知，能够通过建立完备的环境模型来模拟真实反馈。相关算法如动态规划法。

(2) 无模型：状态转移概率 p 未知，Agent 所处的环境模型是未知的。相关算法如蒙特卡洛法、时序差分法、值函数近似以及策略梯度法。

基于模型的优缺点如下。

优点：①能够基于模拟经验数据直接模拟真实环境；②具备推理能力，能够直接评估策略的优劣性；③能够与监督学习算法相结合来求解环境模型。

缺点：存在二次误差。两次近似误差具体体现在以下几方面。

(1) 第一次近似误差：基于真实经验对模型进行学习，得到的模型仅仅是 Agent 对环境的近似描述。

(2) 第二次近似误差：基于模拟模型对值函数或策略进行学习时，存在学习误差。

3.3　求解强化学习任务

对强化学习任务的求解是以 MDP 条件为基础的，因此构建基于 MDP 的强化学习基本框架，如图 3.3 所示。

在 t 时刻，Agent 从环境中得到当前状态 S_t，根据策略 π 执行动作 A_t，并返回奖赏 R_{t+1} 和下一状态 S_{t+1}。Agent 通过不断地与环境交互进行学习，并在学习过程中不断更新策略，经过多次学习后，得到解决问题的最优策略。其中，R_{t+1} 被理解为存在一定延迟的奖赏。

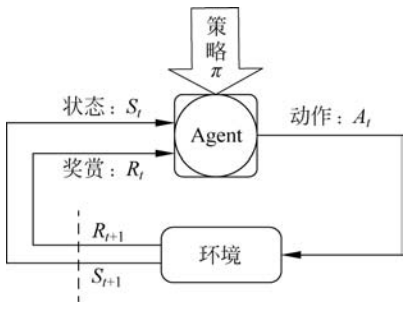

图 3.3　基于 MDP 的强化学习基本框架

以吃豆人游戏为例，Agent 的目标是在网络中，躲避幽灵、吃掉尽量多的食物，其 MDP 模型的基本元素构成如下。

(1) 吃豆人是 Agent，网格世界是环境，吃豆人在网格世界中的位置表示 Agent 所处状态。

(2) 吃豆人的动作空间为上、下、左、右。根据策略，即玩家想法，选择相应动作。

(3) 当吃豆人吃掉食物或被幽灵杀死时，将获得奖赏(奖赏也包括惩罚)。

(4) 达到回报阈值时，Agent 赢得比赛。

在线视频 06

3.3.1 策略

在强化学习基本框架中，除了已知的 MDP 四元组外，还有一个核心要素，即策略(policy)π。强化学习的目的：在 MDP 中搜索到最优策略。策略表示状态到动作的映射，即在某一状态下采取动作的概率分布。与状态转移概率不同，策略概率通常是人为设定的。根据概率分布形式，策略可以分为**确定策略**(deterministic policy)和**随机策略**(stochastic policy)两种。

1. 确定策略

在确定策略下，Agent 在某一状态下只执行一个固定的动作。如确定策略扫地机器人任务，其动作概率分布形如(1,0,0,0)，即在当前状态下，只能执行向上走的动作。确定策略可以将状态映射为一个具体的动作。确定策略函数可以表示为

$$a=\pi(s) \tag{3.10}$$

式(3.10)表示 Agent 处于状态 s 时，根据策略 π，将执行动作 a。

2. 随机策略

在随机策略下，Agent 在一个状态下可能执行多种动作，其动作概率分布形如(0.2,0.2,0.3,0.3)，随机策略将状态映射为执行动作的概率。随机策略函数可以表示为

$$\pi(a|s)=P(a|s)=P(A_t=a|S_t=s) \tag{3.11}$$

式(3.11)表示 Agent 处于状态 s 时，根据策略 π，执行动作 a 的概率。

动态规划法、蒙特卡洛法和时序差分法都只能获得确定策略，而策略梯度法可以获得随机策略。

MDP 应用一个策略产生序列的方法如下。

(1) 从初始状态分布中产生一个初始状态 $S_i=S_0$。

(2) 根据策略 $\pi(a|S_i)$，给出采取的动作 A_i，并执行该动作。

(3) 根据奖赏函数和状态转移函数得到奖赏 R_{i+1} 和下一个状态 S_{i+1}。

(4) $S_i=S_{i+1}$。

(5) 不断重复第(2)步到第(4)步的过程，产生一个序列：

$$S_0,A_0,R_1,S_1,A_1,R_2,S_2,A_2,R_3,S_3,\cdots$$

(6) 如果任务是情节式的，序列将终止于状态 S_{goal}；如果任务是连续式的，序列将无穷延续。

在实际问题中，策略通常都是随机的。强化学习任务一般都具有两种随机性，即策略$\pi(a|s)$的随机性和状态转移概率$p(s',r|s,a)$的随机性。策略的随机性可以是人为设定的，而状态转移的随机性是任务本身所固有的特性，如地面湿滑等。

3.3.2　奖赏与回报


在线视频 07

归根结底，策略是根据值函数进行更新的。当给定一个策略π时，Agent会依据该策略得到一个状态-动作序列，其形式为

$$S_0,A_0,R_1,S_1,A_1,R_2,S_2,A_2,R_3,S_3,\cdots$$

马尔可夫决策过程将**回报**(return)G_t定义为：从当前状态S_t到终止状态S_T所获得的奖赏值之和。其表达式如式(3.12)所示。这里R的下标从$t+1$开始：

$$G_t=R_{t+1}+R_{t+2}+\cdots+R_T \tag{3.12}$$

在实际情况中，对于未来的奖赏，总是存在较大的不确定性。为了避免序列过长或连续任务中回报趋于无穷大的情况，引入**折扣系数**(discounting rate)γ，用于对未来奖赏赋予折扣，带折扣回报定义如下所示：

$$G_t=R_{t+1}+\gamma R_{t+2}+\gamma^2R_{t+3}+\cdots,\quad \gamma\in[0,1] \tag{3.13}$$

从式(3.13)可知，距离当前状态越远的未来状态，对当前状态的回报贡献越小。可以用下一状态S_{t+1}的回报来表示当前状态S_t的回报时，其递归关系式的推导如下所示：

$$\begin{aligned}G_t&=R_{t+1}+\gamma R_{t+2}+\gamma^2R_{t+3}+\cdots\\&=R_{t+1}+\gamma(R_{t+2}+\gamma R_{t+3}+\cdots)\\&=R_{t+1}+\gamma G_{t+1}\end{aligned} \tag{3.14}$$

折扣系数γ存在两种极端情况：当$\gamma=0$时，未来状态无法对当前回报提供任何价值，也就是说，此时只关注当前状态的奖赏，即Agent“看得不够远”；当$\gamma=1$时，若环境已知，则能够确定所有的未来状态，也就是说，Agent能够“清晰准确地向后看”，得到准确的回报值。

例 3.3　设折扣系数$\gamma=0.2$，$T=4$，奖赏序列为$R_1=2$，$R_2=1$，$R_3=5$，$R_4=4$，计算各时刻的回报：$G_0,G_1,\cdots,G_4$。

这里假设终止状态的回报为：$G_4=0$。根据式(3.14)，可通过下一个状态的回报来计算当前状态的回报。这样从最后一个状态开始，反向计算更加简便。计算结果如下：

$$\begin{aligned}G_4&=0\\G_3&=R_4+\gamma\times G_4=4+0.2\times0=4\\G_2&=R_3+\gamma\times G_3=5+0.2\times4=5.8\\G_1&=R_2+\gamma\times G_2=1+0.2\times5.8=2.16\\G_0&=R_1+\gamma\times G_1=2+0.2\times2.16=2.432\end{aligned}$$

例 3.4　考虑例3.1的确定环境下扫地机器人任务。机器人每走一步都伴随着奖赏，奖赏由奖赏函数给出。当机器人到达状态$S_{19}=[5,4]$时，得到+3的奖赏；当机器人到达状态$S_0=[1,1]$时，得到+1的奖赏；当机器人碰到障碍物[3,3]时，得到−10的奖赏，即惩罚；其他情况都得到0的中性奖赏。图3.4为机器人的一段移动轨迹，令折扣系数$\gamma=0.8$，计算轨迹中每个状态的折扣回报。

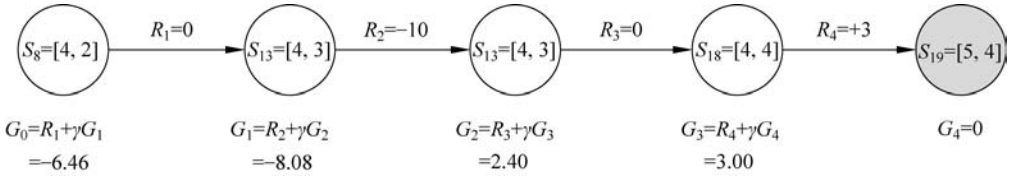

图 3.4 扫地机器人的折扣回报示意图

在线视频 08

3.3.3 值函数与贝尔曼方程

在判定一个给定策略 π 的优劣性时，若 $p(s',r|s,a)$恒等于 1，则每次执行一个动作后进入的下一状态是确定的，此时可以直接使用回报作为评判指标。但在实际情况中，由于状态转移概率的存在，环境通常具有随机性，Agent 从当前状态到达终止状态可能存在多种不同的状态序列，也就存在多个不同的回报 G_t，此时无法直接使用 G_t 来判定策略 π 的优劣性，需要以回报的期望作为策略优劣性的评判指标，于是引入**值函数**(value function)来改进策略。

1. 状态值函数(state-value function)

状态值函数 $v_\pi(s)$表示遵循策略 π，状态 s 的价值，即在 t 时刻，从状态 s 开始，Agent 采取策略 π 得到的期望回报。其计算公式如下所示：

$$v_\pi(s)=\mathbb{E}_\pi(G_t|S_t=s) \tag{3.15}$$

这里将函数 v_π 称为策略 π 的状态值函数。对状态而言，每个状态值函数是由该状态所具有的价值决定的。

2. 动作值函数(action-value function)

动作值函数 $q_\pi(s,a)$表示遵循策略 π，状态 s 采取动作 a 的价值，即在 t 时刻，从状态 s 开始，执行动作 a 后，Agent 采取策略 π 得到的回报期望。其计算公式如下所示：

$$q_\pi(s,a)=\mathbb{E}_\pi(G_t|S_t=s,A_t=a) \tag{3.16}$$

这里将函数 q_π 称为策略 π 的动作值函数。对状态-动作对而言，每个动作值函数是由每个状态采取动作所具有的价值决定的。

3. 状态值函数的贝尔曼方程

由式(3.15)和式(3.16)可知，动作值函数是在状态值函数的基础上考虑了执行动作 a 所产生的影响。根据这一联系，可以构建值函数之间的递归关系，结合回报递归关系式(3.14)，对任意策略 π，当前状态 s 的价值与其下一个状态 s'的价值，满足如下关系式：

$$\begin{aligned}v_\pi(s)&=\mathbb{E}_\pi(G_t|S_t=s)\\&=\mathbb{E}_\pi(R_{t+1}+\gamma G_{t+1}|S_t=s)\\&=\sum_a\pi(a|s)\sum_{s',r}p(s',r|s,a)[r+\gamma\,\mathbb{E}_\pi(G_{t+1}|S_{t+1}=s')]\\&=\sum_a\pi(a|s)\sum_{s',r}p(s',r|s,a)[r+\gamma v_\pi(s')]\end{aligned} \tag{3.17}$$

该推导过程使用了期望公式的展开形式。在式(3.17)的最后一行，对每个$(a、s'、r)$三元组，首先计算$\pi(a|s)p(s',r|s,a)$的概率值，并用该概率对相应的奖赏估计值进行加权，然后对所有可能的取值求和，得到最终期望。式(3.17)称为状态值函数$v_\pi(s)$的贝尔曼方程，用于估计在策略π下，状态s的期望回报。

这里，$\sum_a \pi(a|s)\sum_{s',r} p(s',r|s,a)$也可以写成$\sum_{s',r} p[s',r|s,\pi(s)]$的形式。

根据状态值函数贝尔曼方程，可以构建**状态值函数更新图**(backup diagram)，如图3.5所示，这里空心圆表示状态，实心圆表示动作。

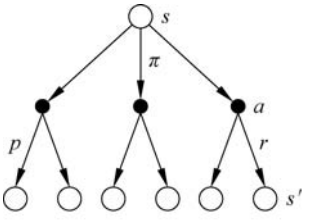

图3.5　状态值函数更新图

由图3.5可知，状态值函数与动作值函数满足关系式

$$v_\pi(s)=\sum_a \pi(a|s)q_\pi(s,a) \tag{3.18}$$

式(3.18)表示Agent遵循策略π，值函数$v_\pi(s)$等于状态s下所有动作值函数$q_\pi(s,a)$的加权和。其中，策略$\pi(a|s)$起到权重的作用。

4. 动作值函数的贝尔曼方程

与状态值函数的贝尔曼方程推导方式类似，同理可以得到动作值函数的贝尔曼方程。如下所示：

$$\begin{aligned}
q_\pi(s,a)&=\mathbb{E}_\pi(G_t|S_t=s,A_t=a)\\
&=\mathbb{E}_\pi(R_{t+1}+\gamma G_{t+1}|S_t=s,A_t=a)\\
&=\mathbb{E}_\pi[R_{t+1}+\gamma v_\pi(S_{t+1})|S_t=s,A_t=a]\\
&=\sum_{s',r} p(s',r|s,a)[r+\gamma v_\pi(s')]\\
&=\sum_{s',r} p(s',r|s,a)\Big[r+\gamma\sum_{a'}\pi(a'|s')q_\pi(s',a')\Big]
\end{aligned} \tag{3.19}$$

根据动作值函数的贝尔曼方程，可以构建**动作值函数更新图**，如图3.6所示。

由图3.6可知，动作值函数与状态值函数满足如下关系式：

$$q_\pi(s,a)=r+\gamma\sum_{s'} p(s'|s,a)v_\pi(s') \tag{3.20}$$

例3.5　如图3.7所示，已知$s_1'、s_2'、s_3'、s_4'$的状态值，利用状态值函数的贝尔曼方程表示s的状态值。

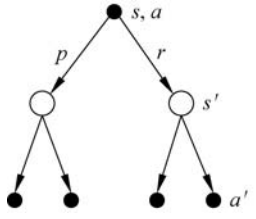

图3.6　动作值函数更新图

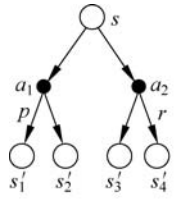

图3.7　例3.5中状态值函数更新图

$$\begin{aligned}v_\pi(s)&=\mathbb{E}_\pi(G_t|S_t=s)\\&=\pi(a_1|s)p(s'_1,r_1|s,a_1)[r_1+\gamma v_\pi(s'_1)]+\pi(a_1|s)p(s'_2,r_2|s,a_1)[r_2+\gamma v_\pi(s'_2)]+\\&\quad\pi(a_2|s)p(s'_3,r_3|s,a_2)[r_3+\gamma v_\pi(s'_3)]+\pi(a_2|s)p(s'_4,r_4|s,a_2)[r_4+\gamma v_\pi(s'_4)]\end{aligned}$$

实际上贝尔曼方程给出了每个状态的状态值或每个状态-动作对的动作值之间的关系。因此可以通过解方程组，得到实际的状态值或动作值。

阅读代码 03

例 3.6 在例 3.1 确定情况扫地机器人任务中，采用的随机策略为：$\pi(a|S_i)=\frac{1}{|\mathcal{A}(S_i)|}$，$a\in\mathcal{A}(S_i)$，这里，$S_i$ 表示序号为 i 的状态，$|\mathcal{A}(S_i)|$表示状态 S_i 可以采取的动作数。在折扣系数 $\gamma=0.8$ 的情况下，求扫地机器人任务中每个状态的状态值。

根据状态值函数贝尔曼方程，可以通过联立线性方程组的方法求解得到在对应策略 π 下，每个状态的状态值。针对处于状态 S_i 的扫地机器人，可以列出如式(3.21)的贝尔曼方程为

$$v_\pi(S_i)=\sum_{a\in\mathcal{A}(S_i)}\pi(a|S_i)p(S_i,a,s')[r(S_i,a)+\gamma v_\pi(s')] \tag{3.21}$$

在式(3.21)中，确定环境下 $p(S_i,a,s')=1$。状态值 $v_\pi(s)$是对状态 s 的累积回报求期望。因为 S_0 和 S_{19} 是终止状态，在这两个状态下无论采取任何动作，都会保持原地不动，且累积奖赏为 0，即这两个状态的状态值 $v_\pi(S_0)$、$v_\pi(S_{19})$均为 0。将每个状态的值函数 $v_\pi(s)$作为未知数，它们之间的关系满足贝尔曼方程，于是可以得到如下方程组：

$$\begin{cases}v_\pi(S_1)=\frac{1}{3}\times[0+0.8\times v_\pi(S_6)]+\frac{1}{3}\times[1+0.8\times v_\pi(S_0)]+\frac{1}{3}\times[0+0.8\times v_\pi(S_2)]\\v_\pi(S_2)=\frac{1}{3}\times[0+0.8\times v_\pi(S_7)]+\frac{1}{3}\times[0+0.8\times v_\pi(S_1)]+\\\qquad\frac{1}{3}\times[0+0.8\times v_\pi(S_3)]\\\qquad\vdots\\v_\pi(S_{11})=\frac{1}{4}\times[0+0.8\times v_\pi(S_{16})]+\frac{1}{4}\times[0+0.8\times v_\pi(S_6)]+\\\qquad\frac{1}{4}\times[0+0.8\times v_\pi(S_{10})]+\frac{1}{4}\times[-10+0.8\times v_\pi(S_{11})]\\v_\pi(S_{13})=\frac{1}{4}\times[0+0.8\times v_\pi(S_{18})]+\frac{1}{4}\times[0+0.8\times v_\pi(S_8)]+\\\qquad\frac{1}{4}\times[-10+0.8\times v_\pi(S_{13})]+\frac{1}{4}\times[0+0.8\times v_\pi(S_{14})]\\\qquad\vdots\\v_\pi(S_{23})=\frac{1}{3}\times[0+0.8\times v_\pi(S_{18})]+\frac{1}{3}\times[0+0.8\times v_\pi(S_{22})]+\\\qquad\frac{1}{3}\times[0+0.8\times v_\pi(S_{24})]\\v_\pi(S_{24})=\frac{1}{2}\times(3+0.8\times v_\pi(19))+\frac{1}{2}\times[0+0.8\times v_\pi(S_{23})]\end{cases}$$

求解方程组，得到状态值，如图 3.8 所示。

−1.111	−1.359	−1.615	−0.329	1.368
−1.417	−2.372	−4.369	−0.987	0.000
−1.831	−4.716		−3.987	−0.300
−0.731	−2.162	−4.649	−2.160	−0.887
0.000	−0.716	−1.772	−1.280	−0.867

图 3.8　基于等概率策略的确定环境扫地机器人任务的状态值

3.3.4　最优策略与最优值函数

利用强化学习方法解决任务的关键在于搜索出 MDP 中的最优策略。**最优策略**就是使得值函数最大的策略。在有穷 MDP 中，由于状态空间和动作空间都是有穷的，所以策略也是有穷的。通过计算值函数可以精确地得到至少一个最优策略 π_*，优于或等效于其他所有策略。对于一个更优策略 π'，执行该策略时，所有状态的期望回报都大于或等于执行 π 策略的期望回报。也就是说，对于所有 $s\in\mathcal{S}$，π' 优于 π，都存在 $v_{\pi'}(s)$ 优于 $v_\pi(s)$。

最优策略可能不止一个，它们共享相同的状态值函数，称为**最优状态值函数**(optimal state-value function)，记为 v_*，定义为

$$v_*(s)=v_{\pi_*}(s)=\max_\pi v_\pi(s),\quad s\in\mathcal{S} \tag{3.22}$$

式(3.22)表明，在状态 s 处，执行最优策略 π_* 的期望回报，也就是在状态 s 处能够获得的最大价值。同理，最优策略共享相同的**最优动作值函数**(optimal action-value function)，记为 q_*，定义为

$$q_*(s,a)=q_{\pi_*}(s,a)=\max_\pi q_\pi(s,a),\quad s\in\mathcal{S},\quad a\in\mathcal{A} \tag{3.23}$$

式(3.23)表明，在状态 s 处，执行动作 a，并在随后的过程中采取最优策略 π_* 得到的期望回报，也就是在状态-动作对(s,a)处能够获得的最大价值。

1. 贝尔曼最优方程

对于最优策略值函数 v_* 而言，一方面因为 v_* 是策略的值函数，必定满足贝尔曼方程(3.17)。另一方面又因为它是最优的值函数，因此引入一种不与任何特定策略有关的表达形式来表示它，即**贝尔曼最优方程**(Bellman optimality equation)。贝尔曼最优方程表示：Agent 在采用最优策略 π_* 时，各个状态 s 的价值一定等于在该状态下执行最优动作 a 的期望回报。v_* 的贝尔曼最优方程定义为

$$\begin{aligned}
v_*(s)&=\max_{a\in\mathcal{A}(s)}q_{\pi_*}(s,a)\\
&=\max_a\mathbb{E}_{\pi_*}(R_{t+1}+\gamma G_{t+1}\mid S_t=s,A_t=a)\\
&=\max_a\mathbb{E}(R_{t+1}+\gamma v_*(S_{t+1})\mid S_t=s,A_t=a)\\
&=\max_a\sum_{s',r}p(s',r\mid s,a)[r+\gamma v_*(s')]
\end{aligned} \tag{3.24}$$

式(3.24)中最后两行的表达式为关于 v_* 的贝尔曼最优方程的两种表达形式。同理，q_* 的贝尔曼最优方程定义为

$$\begin{aligned} q_*(s,a) &= \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) | S_t = s, A_t = a] \\ &= \mathbb{E}[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') | S_t = s, A_t = a] \\ &= \sum_{s',r} p(s', r | s, a)[r + \gamma \max_{a'} q_*(s', a')] \end{aligned} \tag{3.25}$$

关于 v_* 和 q_* 的贝尔曼最优方程，其更新图分别如图 3.9 和图 3.10 所示，弧线表示取最大值，其他部分都与图 3.5 和图 3.6 的 v_π、q_π 的更新图相同。

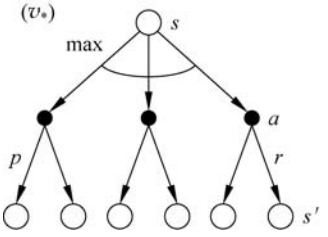

图 3.9 关于 v_* 的贝尔曼最优方程更新图

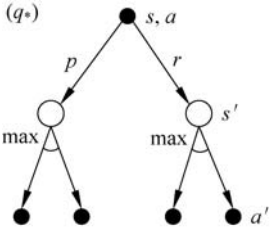

图 3.10 关于 q_* 的贝尔曼最优方程更新图

2. 最优策略

Agent 通过与环境不断地交互获得的信息来更新策略，以最终获得最优值函数。一旦获得最优状态值函数 v_* 或最优动作值函数 q_*，Agent 便能得到最优策略。

若已知 v_*，单步搜索后，最好的动作就是全局最优动作。即对于 v_* 来说，任意贪心策略都是最优策略。贪心用于描述仅基于当前情况或局部条件进行的最优选择，不考虑未来的变化与影响，是一种基于短期视角的方案。由于 v_* 本身包含了未来所有可能的动作产生的回报影响，因此可以将对最优的长期(全局)回报期望的求解，转化为计算每个状态对应的局部变量，以单步搜索方式产生长期的最优动作序列。简单来看，基于贪心策略，Agent 总是朝着具有更大状态值函数的状态进行移动。

若已知 q_*，由于最优动作值函数以最大的长期回报期望来表达每组状态-动作对的局部变量，最优动作的选择无须知道后续状态及其对应的价值，Agent 可以直接选择最大动作值函数所对应的动作，这一方法也称为贪心动作选择方法，其表达式为

$$A_t = \arg\max_a q_t(s,a) \tag{3.26}$$

由于受环境的完备性、计算能力和计算资源的限制，在寻找最优策略时，直接对包含 max 的非线性最优贝尔曼方程组求解是比较困难的。通常采用迭代方式，如动态规划法，经过一轮一轮的迭代，最终收敛为最优解。第 4 章将会详细介绍动态规划法。

阅读代码 04

例 3.7 求解例 3.1 中确定环境下扫地机器人任务的最优状态值函数，并给出最优策略。

设折扣系数 $\gamma=0.8$。利用式(3.24)，可以显式地给出在该扫地机器人任务中，状态 S_i 的最优贝尔曼方程为

$$v_*(S_i) = \max_a p(S_i, a, s')[r(S_i, a) + \gamma v_*(s')] \tag{3.27}$$

在式(3.27)中，确定环境下 $p(S_i,a,s')=1$。将每个状态的值函数 $v_\pi(s)$作为未知数，它们之间的关系满足最优贝尔曼方程，于是利用式(3.27)可以得到扫地机器人任务的最优贝尔曼方程组。利用第4章的值迭代算法，可以对贝尔曼最优方程组进行求解，并找到该扫地机器人任务的最优策略，计算结果如图3.11所示。

1.229	1.536	1.920	2.400	3.000
1.536	1.920	2.400	3.000	0.000
1.229	1.536		2.400	3.000
1.000	1.229	1.536	1.920	2.400
0.000	1.000	1.229	1.536	1.920

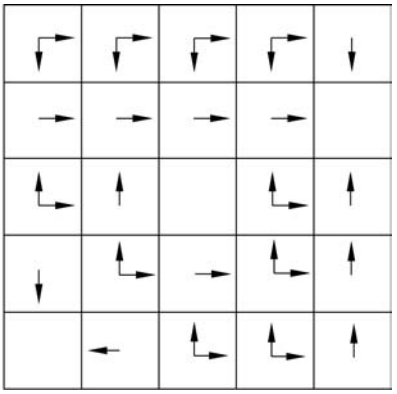

图3.11　确定环境下扫地机器人任务的最优值函数和最优策略

通过图3.11可以看出，当机器人处于状态 $S_6=[2,2]$时，可以采取4个不同的动作。通过一步搜索，计算出上、下、左和右4个动作值，分别为1.229、0.8、0.8、1.229，于是得出状态 $S_6=[2,2]$的最优动作为向上和向右，共2个动作。当机器人处于状态 $S_{17}=[3,4]$时，仍可以采取4个不同的动作。通过一步搜索，计算出上、下、左和右4个动作值，分别为1.536、−8.080、1.536、2.400，于是得出状态 $S_{17}=[3,4]$的最优动作为向右，1个动作。

综上，利用最优值函数计算最优策略时，需要通过一步搜索，计算出其所有动作值，通过选取具有最大动作值的动作，得到最优策略。而利用动作值寻找最优策略时，由于动作值中包含了可采取动作的信息，因此只需要选取具有最大动作值的动作，即可得到最优策略。

3.4　探索与利用

由于强化学习的目的是得到一个最优策略，使得回报最大化。这就导致了强化学习的一大矛盾：探索与利用的平衡。一方面，Agent秉持**利用机制**(exploitation)，为了得到最大回报，需要始终采用最优动作，即根据当前的值函数选择最优动作，最大限度地提升回报；另一方面，Agent需要**探索机制**(exploration)，摒弃基于值函数的贪心策略，找到更多可能的动作来获得更好的策略，探索更多的可能性。

通常我们采用具有探索性的策略来解决以上矛盾。常用的两种方法为**同策略**和**异策略**。使用这两种方法一方面保证足够的探索性，另一方面充分地利用最优策略。在介绍它们之前，先来了解行为策略和目标策略。

(1) **行为策略**(behavior policy)：用于产生采样数据的策略，具备探索性，能够覆盖

所有情况，通常采用**ε-柔性策略**(ε-soft policy)。

(2) **目标策略**(target policy)：强化学习任务中待求解的策略，也就是待评估和改进的策略，一般不具备探索性，通常采用确定贪心策略。

下面对同策略和异策略的特点分别阐述。

(1) **同策略**(on-policy)：行为策略和目标策略相同。通过ε-贪心策略平衡探索和利用，在保证初始状态-动作对(S_0,A_0)不变的前提下，确保每一组(s,a)都有可能被遍历到。常用算法为 Sarsa 和 Sarsa(λ)算法。

(2) **异策略**(off-policy)：行为策略和目标策略不同。将探索与利用分开，在行为策略中贯彻探索原则：采样数据，得到状态-动作序列；在目标策略中贯彻利用原则：更新值函数并改进目标策略，以得到最优目标策略。常用算法为 Q-Learning 算法和 DQN 算法。

同策略方法较为简单，通常会被优先考虑，但异策略方法更为通用，效果也更好，可以用来学习由传统的非学习型控制器或人类专家生成的数据。异策略方法的优缺点分别阐述如下。

异策略优点：

- 效果好，更具有一般性，可以从示例样本或其他 Agent 给出的数据样本中进行学习。
- 可以重复利用旧策略。
- 可以在使用一个探索策略的基础上，学习一个确定策略。
- 可以用一个行为策略进行采样，多个目标策略同时进行学习。

异策略缺点：

- 由于数据来源于一个不同的策略，存在方差较大、收敛速度慢等缺陷。

下面介绍 2 个概念。

偏差(bias)：在机器学习中，偏差和欠拟合相关，预测值与样本之间的偏差越大，说明精度越低。在强化学习中，偏差指通过采样数据获得的估计平均值与实际平均值的偏离程度。

方差(variance)：在机器学习中，方差和过拟合相关，样本值之间的方差越大，说明样本的置信度越差，模型适用性越差。在强化学习中，方差指单次采样结果相对于均值的波动大小。

3.5 小结

本章主要介绍了强化学习的基础数学理论，以马尔可夫决策过程描述了 Agent 与环境的交互过程。状态是 Agent 选择动作的基础，通过动作的选择完成状态的转移，并以奖赏评判 Agent 动作选择的优劣。

有限的状态、动作和收益共同构成了有限马尔可夫决策过程，回报刻画了 Agent 能获得的全部未来奖赏，对于不同的任务，未来状态的奖赏会有不同的折扣，而 Agent 的任务就是最大化回报。动作的选择依赖于 Agent 所采取的策略，而强化学习的目的就是获

得最优策略。

引入状态值函数和动作状态值函数来描述回报，通过贝尔曼最优方程将马尔可夫决策过程表达抽象化，从而可以相对容易地求解得到最优价值函数。在强化学习问题中，定义环境模型和明确最优值函数是计算最优策略的基础，在后续章节中，将进一步讨论如何求解最优策略。

3.6　习题

1. 举例说明基于模型与无模型强化学习的异同点。

2. 分别给出如图 3.12 所示的确定环境和随机环境下扫地机器人任务的 MDP 数学模型。与例 3.1 和例 3.2 相比，主要有两方面的变化：

(1) 图 3.12 中障碍物、充电桩及垃圾位置不同；

(2) 在任何状态下都有上、下、左、右 4 个不同的动作，当采取冲出边界的动作时，机器人保持原地不同。其他参数设置与例 3.1 和例 3.2 相同。

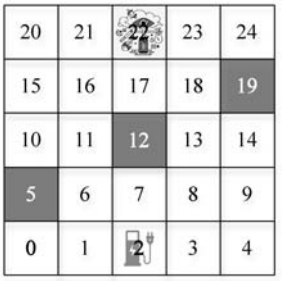

图 3.12　扫地机器人任务(变化)

3. 考虑一个折扣系数为 γ 的连续式任务，其奖赏序列为 $R_1, R_2 = R_3 = \cdots =$，计算 G_0, G_1 的值。

4. (编程)通过解方程组计算：在确定环境、等概率策略下，图 3.12 扫地机器人在折扣系数 $\gamma = 0.8$ 的情况下，每个状态的状态值。

5. 简述同策略与异策略强化学习的异同点。