3 - 动态规划

- 马尔可夫决策过程是强化学习中的基本问题模型之一，而解决马尔可夫决策过程的方法我们统称为强化学习算法。
- 本章开始讲强化学习中最基础的算法之一，动态规划。
- 动态规划具体指的是在某些复杂问题中，将问题转化为若干个子问题，并在求解每个子问题的过程中保存已经求解的结果。
- 常见的动态规划算法包括：值迭代（Value Iteration）、策略迭代（Policy Iteration）和 Q-learning 算法等

动态规划的编程思想

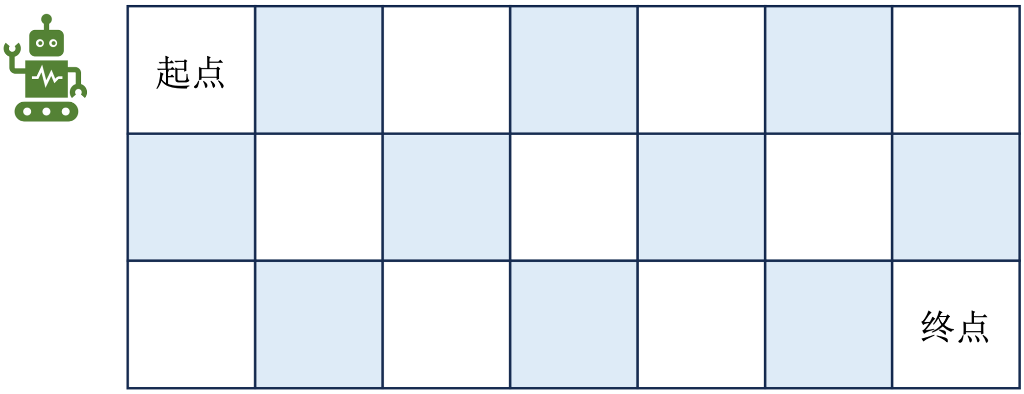


图 3.1 路径之和

如上图所示，一个机器人位于一个 $m \times n$ 网格左上角的起点。机器人每次移动一格，向下或向左。机器人达到终点意为结束，那共有多少个方法/路径可以到达终点。

使用动态规划中的由下至上，我们从终点开始剖析可能性，只可能是从上或从左来。
若终点记作 (i,j)，那从上的可能性为该路径的总和记作$f(i-1,j)$，而从左的可能性为该路径的总和记作 $f(i,j-1)$。由此我们可以得出到达终点的可能性为

$$f(i,j)=f(i-1,j)+f(i,j-1)$$

通过循环我们可以追溯会起点，将这个问题分成 n 个子问题进行计算。

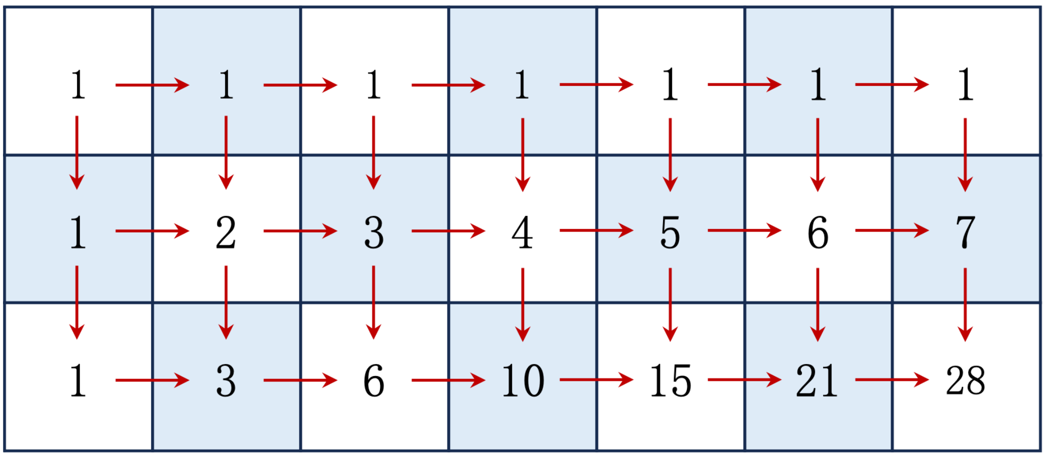


图 3.2 路径之和解析

我们可以将这个解法代码化如下：

```python
def solve(m,n):
    # 初始化边界条件
    f = [[1] * n] + [[1] + [0] * (n - 1) for _ in range(m - 1)]
    # 状态转移
    for i in range(1, m):
        for j in range(1, n):
            f[i][j] = f[i - 1][j] + f[i][j - 1]
    return f[m - 1][n - 1]
```

最后能解出来当 $m=7, n=3$ 时一共有 28 种不同的路径

状态价值函数和动作价值函数

在马尔可夫决策过程中，每个状态是有一定的价值的，可以定义为：

$$\begin{aligned} V_\pi(s) &= \mathbb{E}_\pi[R_t+\gamma R_{t+1}+\gamma^2 R_{t+2}+\ldots|S_t=s] \\ &= \mathbb{E}_\pi[G_t|S_t=s] \end{aligned}$$

这就是状态价值函数（state-value function），从特定状态出发，按照某种策略 π 进行决策所能得到的回报期望值。

另外引入动作的元素后会有一个 Q 函数，也叫做动作价值函数（action-value function），即：

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

从这儿我们能看出状态价值函数和动作价值函数之间的关系为：

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

其中 $\pi(a|s)$ 表示策略函数，一般指在状态 a 下执行动作 a 的概率分布。这个公式说明了在给定状态 s 的情况下，智能体所有动作的价值期望。

贝尔曼方程

和上一章提到的回报公式$G_t=R_{t+1}+\gamma G_{t+1}$一样，我们也可以对状态价值函数和动作价值函数做一样的推导：

$$\begin{aligned} V_\pi(s) &= \mathbb{E}_\pi[G_t|S_t=s] \\ &= \mathbb{E}_\pi[R_{t+1}+\gamma R_{t+2}+\gamma^2 R_{t+3}+\ldots|S_t=s] \\ &= \mathbb{E}[R_{t+1}|S_t=s]+\gamma\mathbb{E}[R_{t+2}+\gamma R_{t+3}+\gamma^2 R_{t+4}+\ldots|S_t=s] \\ &= R(s)+\gamma\mathbb{E}[V_\pi(s_{t+1}|S_t=s)] \\ &= R(s)+\gamma\sum_{s'\in S}P(S_{t+1}=s'|S_t=s)V_\pi(s') \\ &= R(s)+\gamma\sum_{s'\in S}P(s'|s)V_\pi(s') \end{aligned}$$

其中 $R(s)$ 表示奖励函数，$P(S_{t+1}=s'|S_t=s)$ 就是状态转移概率，一般简写成 $P(s'|s)$，这就是贝尔曼方程。

而动作价值哈数的贝尔曼方程推导如下：

$$Q_\pi(s,a)=R(s,a)+\gamma\sum_{s'\in s}p(s'|s,a)\sum_{a'\in a}Q_\pi(s',a')$$

为了得到一个最好的结果，我们可以对状态价值函数最优化以寻求最优解，那么最优策略下的状态价值函数可以表示为：

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

这就是贝尔曼最优方程，同理岁动作价值函数寻求最优策略：

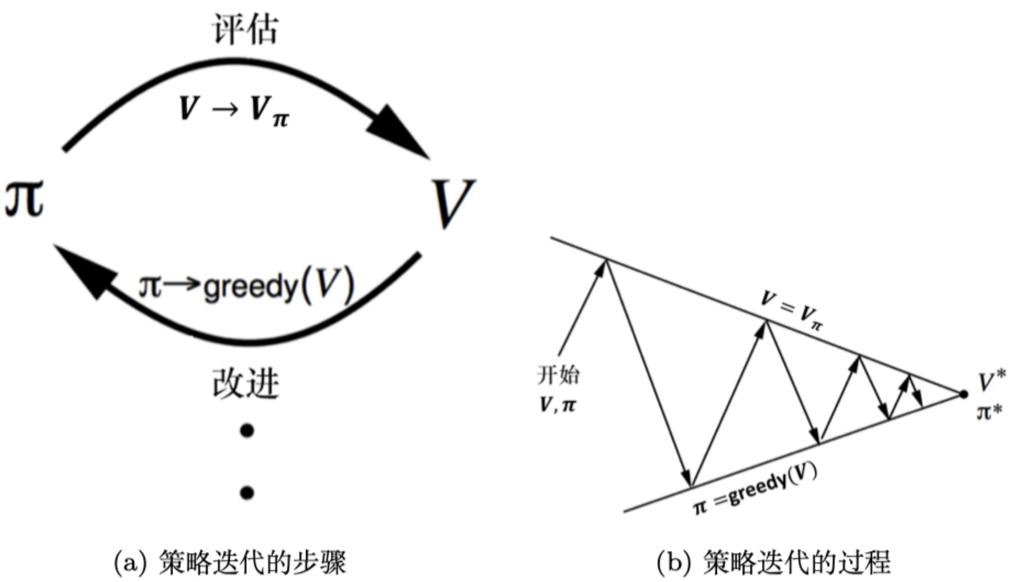
$$\begin{aligned}Q^*(s,a)&=\mathbb{E}[R_{t+1}+\gamma\max_{a'}Q^*(S_{t+1},a')|S_t=s,A_t=a]\\&=\sum_{s',r}p(s',r|s,a)[r+\gamma\max_{a'}Q^*(s',a')]\end{aligned}$$

策略迭代

策略迭代算法的思路是分成两个步骤，首先固定策略 π 不变，然后估计对应的状态价值函数 V ，这一叫做策略估计（policy evaluation）。然后根据估计好的状态价值函数 Q 结合策略推算出动作价值函数 Q ，并对 Q 函数优化然后进一步改进策略，这一步叫策略改进（policy improvement）。在策略改进的过程中一般是通过贪心策略来优化的，即定义策略函数为：

$$\pi(a|s)=\max_a Q(s,a)$$

这个策略估计的过程会重复迭代的过程直至收敛完毕。



(a) 策略迭代的步骤　　(b) 策略迭代的过程

图 3.3 策略迭代的收敛过程

策略迭代算法伪代码如下： ![[Pasted image 20231115160658.png|center|650]]

图 3.4 策略迭代算法伪代码

价值迭代

价值迭代函数：

$$V(s)=\max_{a\in A}(R(s,a)+\gamma\sum_{s^{'}\in S}p(s^{'}|s,a)V(s^{'}))$$

价值迭代的伪代码如下：

价值迭代算法

初始化一个很小的参数阈值 $\theta>0$，以及状态价值函数 $V(s)$，注意终止状态的 $V(s_T)=0$
repeat
  $\Delta \leftarrow 0$
  repeat
    $v \leftarrow V(s)$
    $V(s) \leftarrow \max_a \sum_{s',r} p\left(s', r \mid s, a\right)\left[r+\gamma V\left(s'\right)\right]$
    $\Delta \leftarrow \max(\Delta, |v-V(s)|)$
  until 遍历所有的状态 $s \in S$
until $\Delta<\theta$
输出一个确定性策略 $\pi \approx \pi_*$，
且 $\pi(s)=\arg\max_a \sum_{s',r} p\left(s', r \mid s, a\right)\left[r+\gamma V\left(s'\right)\right]$

图 3.4 价值迭代算法伪代码

练习题

动态规划问题的主要性质有哪些？

- **将复杂的问题分成若干个子问题，并独立求解**
- **保存每一个不同子问题的结果，以便减少过后的使用**

状态价值函数和动作价值函数之间的关系是什么？

动作价值函数基于状态价值函数

策略迭代和价值迭代哪个算法速度会更快？

策略迭代算法