算法分析和复杂性理论

第 6 次作业

张瀚文 2201212865

1 背包问题

题目描述：

一个旅行者随身携带一个背包，可以放入背包的物品有n种，每种物品的重量和价值分别是w_i, v_i, $i=1,\cdots,n$. 如果背包的最大容量限制是b, 怎样选择放入背包的物品以使得背包的价值最大?

解题思路：对于第 i 个物品，有两种状态：

1. 拿，背包容量减少 wi，价值增加 vi。
2. 不拿，背包容量不变，价值不变，继续考虑第 i-1 个物品。

V[i, j]表示在面对第 i 件物品，且背包容量为 j 时所能获得的最大价值。

那么状态转移方程为：

$$f[i][j]=max(f[i-1][j],f[i-1][j-w[i]]+c[i])$$

测试代码：

```
def bag(n, c, w, v):
    """
    测试数据:
    n = 6  物品的数量,
    c = 10 书包能承受的重量,
    w = [2, 2, 3, 1, 5, 2] 每个物品的重量,
    v = [2, 3, 1, 5, 4, 3] 每个物品的价值
    """
    # 置零，表示初始状态
    value = [[0 for j in range(c + 1)] for i in range(n + 1)]
    for i in range(1, n + 1):
        for j in range(1, c + 1):
            value[i][j] = value[i - 1][j]
            # 背包总容量够放当前物体，遍历前一个状态考虑是否置换
            if j >= w[i - 1] and value[i][j] < value[i - 1][j - w[i - 1]] +
v[i - 1]:
                value[i][j] = value[i - 1][j - w[i - 1]] + v[i - 1]
    for x in value:
        print(x)
```

```
    return value
```

2 投资问题

题目描述：

设有m元钱，n项投资，函数 $f_i(x)$ 表示将x元钱投入到第 i 项项目所产生的效益，i=1,···, n. 问：如何分配这m元钱，使得投资的总效益最高?

解题思路：假设 $F_k(x)$为 x 万元钱投资给前 k 个项目获得的最大收益，用动态规划思路求解，问题就转成求 $F_k(m)$。当 k=1 时，也就是说只投一个项目时最大收益为：$F_k(x)=f_1(x)$；当 1<k<=n 时，即至少投资两个项目以上时，设 p 为分配给第 k 个项目的资金，此时还剩下 x-p 的资金可分配给前 k-1 个项目，则获得的最大收益为：$f_k(p)+F_{k-1}(x-p)$。根据上面的分析，可以得到转移方程：

$$F_k(x)=\begin{cases} max\{f_k(p)+F_{k-1}(x-p)\}, 0\leqslant p\leqslant x 0<k\leqslant n \\ f_1(x), k=1 \end{cases}$$

测试代码：

```
import numpy as np
'''
m = 5  # 投资总额
n = 6  # n 项投资
k = 4  # 项目数

'''
dp = np.zeros((m, n))  # dp[i][j]  从 1-i 号项目中选择，投资 j 万元，所取得的最大收益
mark = np.zeros((m, n))  # 从 1-i 号项目中选择，投资 j 万元，获得最大收益时，在第 i 号项目中投资了多少钱
f = np.array([[0, 0, 0, 0, 0, 0],
              [0, 11, 12, 13, 14, 15],
              [0, 0, 5, 10, 15, 20],
              [0, 2, 10, 30, 32, 40],
              [0, 20, 21, 22, 23, 24]])

# 初始化第一行
for j in range(m + 1):
    dp[1][j] = f[1][j]
    mark[1][j] = j
for i in range(1, k + 1):
    for j in range(1, m + 1):
        for k in range(j):
            if dp[i][j] < f[i][k] + dp[i - 1][j - k]:
```

```python
                dp[i][j] = f[i][k] + dp[i - 1][j - k]  # 更新当前最优解
                mark[i][j] = k  # 更新标记函数

print("最大收益", dp[k][m])

for i in range(1, k + 1):
    for j in range(m + 1):
        print("(%d, %d)" % (dp[i][j], mark[i][j]), end="\t")
    print("\n")

for i in range(k, 0, -1):
    print(f"第{i}个项目投资{mark[i][m]}元")
    m = m - int(mark[i][m])
```