算法设计与分析
第 7 章作业参考答案

作业情况

教材： 算法设计技巧与分析 [沙特]M. H. Alsuwaiyel
题目编号： 7.5, 7.9, 7.15, 7.27
题目范围： 动态规划
邮箱： wjyyy1@126.com

习题 7.5

用算法 LCS 来找两个字符串 $A=$“xzyzzyx”，$B=$“zxyyzxz”的最长公共子序列的长度。给出一个最长公共子序列。

答案：

长度为 4，最长公共子序列可以是

- xyzx
- zyyx
- xyyx
- zyzx
- xyzz
- zyzz

解析： 列表讨论，当 $a_i = b_j$ 时，可以从左上角转移，否则从上或从左转移。

		0	1	2	3	4	5	6	7
			x	z	y	z	z	y	x
0		0	0	0	0	0	0	0	0
0	z	0	0	1	1	1	1	1	1
1	x	0	1	1	1	1	1	1	2
2	y	0	1	1	2	2	2	2	2
3	y	0	1	1	2	2	2	3	3
4	z	0	1	2	2	3	3	3	3
5	x	0	1	2	2	3	3	3	4
6	z	0	1	2	2	3	4	4	4

习题 7.9

考虑用算法 MATCHAIN 把下面 5 个矩阵相乘

$M_1 : 4 \times 5, M_2 : 5 \times 3, M_3 : 3 \times 6, M_4 : 6 \times 4, M_5 : 4 \times 5$

假设为了要得到乘法 $M_1 \times M_2 \times M_3 \times M_4 \times M_5$ 的中间结果如表所示，这里的 $C[i,j]$ 是执行乘法 $M_i \times \cdots \times M_j (1 \leqslant i \leqslant j \leqslant 5)$ 所需要的数量乘法的最少次数。表中还显示了括号表达式所显示的乘法 $M_i \times \cdots \times M_j$ 执行的最优顺序。找出 $C[1,5]$ 和执行乘法 $M_1 \times \cdots \times M_5$ 的最优括号化表达式。

$C[1,1]=0$ M_1	$C[1,2]=60$ M_1M_2	$C[1,3]=132$ $(M_1M_2)M_3$	$C[1,4]=180$ $(M_1M_2)(M_3M_4)$	
	$C[2,2]=0$ M_2	$C[2,3]=90$ M_2M_3	$C[2,4]=132$ $M_2(M_3M_4)$	$C[2,5]=207$ $M_2((M_3M_4)M_5)$
		$C[3,3]=0$ M_3	$C[3,4]=72$ M_3M_4	$C[3,5]=132$ $(M_3M_4)M_5$
			$C[4,4]=0$ M_4	$C[4,5]=120$ M_4M_5
				$C[5,5]=0$ M_5

答案：

$C[1,1]=0$ M_1	$C[1,2]=60$ M_1M_2	$C[1,3]=132$ $(M_1M_2)M_3$	$C[1,4]=180$ $(M_1M_2)(M_3M_4)$	$C[1,5]=252$ $(M_1M_2)((M_3M_4)M_5)$
	$C[2,2]=0$ M_2	$C[2,3]=90$ M_2M_3	$C[2,4]=132$ $M_2(M_3M_4)$	$C[2,5]=207$ $M_2((M_3M_4)M_5)$
		$C[3,3]=0$ M_3	$C[3,4]=72$ M_3M_4	$C[3,5]=132$ $(M_3M_4)M_5$
			$C[4,4]=0$ M_4	$C[4,5]=120$ M_4M_5
				$C[5,5]=0$ M_5

解析： 考虑

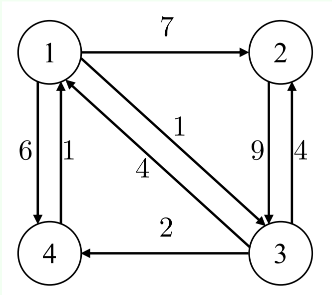
$$C[i,j] = \min_{i<k\leqslant j} \{C[i,k-1] + C[k,j] + r_i r_k r_{j+1}\}$$

枚举 $k = 2,3,4,5$，分别为

- $C[1,1] + C[2,5] + r_1 r_2 r_6 = 307$
- $C[1,2] + C[3,5] + r_1 r_3 r_6 = 252$
- $C[1,3] + C[4,5] + r_1 r_4 r_6 = 372$
- $C[1,4] + C[5,5] + r_1 r_5 r_6 = 260$

习题 7.15

在如图所示的含权有向图上执行所有点对最短路径算法。



答案:

$$\begin{bmatrix} 0 & 5 & 1 & 3 \\ 12 & 0 & 9 & 11 \\ 3 & 4 & 0 & 2 \\ 1 & 6 & 2 & 0 \end{bmatrix}$$

解析: 构造如图所示初始矩阵

$$\begin{bmatrix} 0 & 7 & 1 & 6 \\ \infty & 0 & 9 & \infty \\ 4 & 4 & 0 & 2 \\ 1 & \infty & \infty & 0 \end{bmatrix}$$

第一次迭代:

$$\begin{bmatrix} 0 & 7 & 1 & 6 \\ \infty & 0 & 9 & \infty \\ 4 & 4 & 0 & 2 \\ 1 & 8 & 2 & 0 \end{bmatrix}$$

第二次迭代:

$$\begin{bmatrix} 0 & 7 & 1 & 6 \\ \infty & 0 & 9 & \infty \\ 4 & 4 & 0 & 2 \\ 1 & 8 & 2 & 0 \end{bmatrix}$$

第三次迭代:

$$\begin{bmatrix} 0 & 5 & 1 & 3 \\ 13 & 0 & 9 & 11 \\ 4 & 4 & 0 & 2 \\ 1 & 6 & 2 & 0 \end{bmatrix}$$

第四次迭代:

$$\begin{bmatrix} 0 & 5 & 1 & 3 \\ 12 & 0 & 9 & 11 \\ 3 & 4 & 0 & 2 \\ 1 & 6 & 2 & 0 \end{bmatrix}$$

习题 7.27

另一种类型的背包问题是设集合 U 是包含物品类型的集合，目的是用每类物品的任意数量物品装满背包，使在不超过背包容量的前提下物品的总价值最大。假设每一类物品的个数都是无限的，更形式化地，设 $T=\{t_1,t_2,\cdots,t_n\}$ 是一个 n 种类型物品的集合，C 是背包的容量。对于 $1\leqslant j\leqslant n$，令 s_j 和 v_j 分别是类型 j 物品的体积和价值。找出一个非负整数 $x_1,x_2,\cdots,x_n$ 集合，使

$$\sum_{i=1}^{n} x_i v_i$$

在约束条件

$$\sum_{i=1}^{n} x_i s_i \leqslant C$$

下最大。其中 $x_1,x_2,\cdots,x_n$ 是非负整数。
注意，$x_j=0$ 是指没有第 j 种物品装到背包中去。请就这种类型的背包问题重写动态规划算法。

答案：改写递推式，但含义不变。$V[i,j]$ 表示从前 i 项 $\{t_1,t_2,\cdots,t_n\}$ 取出来的装入体积为 j 的背包的物品的最大价值，而且可以选择任意数量。

由于可以选择了任意数量，因此可以从 $V[i,j-s_i]$ 进行转移。递推式如下

$$V[i,j]=\begin{cases} 0 & i=0\text{或}j=0 \\ V[i-1,j] & j<s_i \\ \max\{V[i-1,j],V[i-1,j-s_i]+v_i,V[i,j-s_i]+v_i\} & i>0\text{且}j\geqslant s_i \end{cases}$$

Algorithm 1: KNAPSACK-2

Input: 物品集合 $T=\{t_1,t_2,\cdots,t_n\}$，体积分别为 $s_1,s_2,\cdots,s_n$，价值分别为 $v_1,v_2,\cdots,v_n$，容量为 C 的背包

Output: $\sum_{i=1}^{n} x_i v_i$ 在约束条件 $\sum_{i=1}^{n} x_i s_i \leqslant C$ 下的最大总价值

for $i \leftarrow 0$ **to** n **do**
    $V[i,0] \leftarrow 0$
end
for $j \leftarrow 0$ **to** C **do**
    $V[0,j] \leftarrow 0$
end
for $i \leftarrow 1$ **to** n **do**
    for $j \leftarrow 1$ **to** C **do**
        $V[i,j] \leftarrow V[i-1,j]$;
        if $s_i \leqslant j$ **then** $V[i,j] \leftarrow \max\{V[i,j], V[i,j-s_i]+v_i\}$;
    end
end
return $V[n,C]$

此外可以考虑将递推式写成如下形式：

$$V[j]=\begin{cases}0 & j=0\\ V[j] & j<s_i\\ \max\{V[j],V[j-s_i]+v_i\} & j\geqslant s_i\end{cases}$$

因为这种背包问题不需要保证最多一次，所以淡化了递推时第一维 i 的概念，但为了保证无后效性，i 必须递增枚举。算法具体伪代码如下，感兴趣的同学可自行研究。也可以进行二进制分解，算法从略。

Algorithm 2: KNAPSACK-3

Input: 物品集合 $T=\{t_1,t_2,\cdots,t_n\}$，体积分别为 $s_1,s_2,\cdots,s_n$，价值分别为 $v_1,v_2,\cdots,v_n$，容量为 C 的背包

Output: $\sum_{i=1}^{n} x_i v_i$ 在约束条件 $\sum_{i=1}^{n} x_i s_i \leqslant C$ 下的最大总价值

for $j \leftarrow 0$ **to** C **do**
    $V[j] \leftarrow 0$
end
for $i \leftarrow 1$ **to** n **do**
    for $j \leftarrow s_i$ **to** C **do**
        $V[j] \leftarrow \max\{V[j], V[j-s_i]+v_i\}$
    end
end
return $V[C]$

总结

题目编号：7.5, 7.9, 7.15, 7.27

日期：2023 年 4 月 13 日

批改人：王骏峣

邮箱：wjyyy1@126.com

习题 7.5：表格中任何一个数字都应该不超过它右下方（包括同一行同一列）的数。

习题 7.9：部分同学代错了 $r_i r_{j+1} r_k$，要注意在 5 个矩阵中，一共有 6 个数字，注意它们的依赖关系。

习题 7.15：注意迭代的顺序，有的矩阵值不应提前计算。

习题 7.27：有同学在做 $O(nC)$ 递推的同时枚举了每个物品的个数 x_i，这时的时间复杂度是高于 $O(nC)$ 的。此时应注意 $k \times s_i \leqslant j$ 的限制。

此外，枚举第二维时，上界是 C 而不是 n。

本次作业发现有高度重合现象，这部分同学本次作业分数减 10%。