算法分析与设计基础 第六周作业

徐浩博　软件02　2020010108

Problem 1

我们修改EXTENDED-BOTTOM-UP-CUT-ROD(p, n)

```
EXTENDED_BOTTOM_UP_CUT_ROD(p, n, c):
        let r[0...n] and s[0...n] be new arrays
        r[0] = 0
        for j = 1 to n:
                q = p[j]
                s[j] = j
                for i = 1 to j - 1:
                        if q < r[j - i] + p[i] - c:
                                q = r[j - i] + p[i] - c
                                s[j] = i
                r[j] = q
        return r and s
```

同时PRINT-CUT-ROD-SOLUTION保持不变：

```
PRINT_CUT_ROD_SOLUTION(p, n, c):
        (r, s) = EXTENDED_BOTTOM_UP_CUT_ROD(p, n, c)
        while n > 0:
                print s[n]
                n = n - s[n]
```

Problem 2

我们考虑四个矩阵相乘，$A_{6\times 5}$，$B_{5\times 10}$，$C_{10\times 8}$，$D_{8\times 20}$，那么p数组就对应为：

p[0]	p[1]	p[2]	p[3]	p[4]
6	5	10	8	20

根据贪心的算法，p[0]p[1]p[4]最小，应该划分为(1,1),(2,4)两个子问题，(2,4)中又有p[1]p[3]p[4]最小，故划分为(1,3),(4,4)两个子问题，因此乘法顺序可以确定为$(A \times ((B \times C) \times D))$，总计算规模为：

$$p[1] \times p[2] \times p[3] + p[1] \times p[3] \times p[4] + p[0] \times p[1] \times p[4] = 1800$$

而根据动态规划的方法，最优的划分为$(A \times (B \times C)) \times D$，总计算规模为：

$$p[1] \times p[2] \times p[3] + p[0] \times p[1] \times p[3] + p[0] \times p[3] \times p[4] = 1600$$

由此可见，贪心算法并不一定总能给出最优解；上面举出的反例中，贪心求解只能求出次优解.