HW7

page222: 15.3-3 15.3-5

page226:15.4-3

page230:15.5-2

15.3

page222: 15.3-3 15.3-5

15.3-3 考虑矩阵链乘法问题的一个变形：目标改为最大化矩阵序列括号化方案的标量乘法运算次数，而非最小化。此问题具有最优子结构性质吗？

具有，m[i,j]的min改为max即为最大化问题的最优子结构。

$$m[i,j]=\begin{cases}0 & \text{如果 } i=j\\ \min\limits_{i\le k<j}\{m[i,k]+m[k+1,j]+p_{i-1}p_kp_j\} & \text{如果 } i<j\end{cases} \qquad (15.7)$$

15.3-5 对 15.1 节的钢条切割问题加入限制条件：假定对于每种钢条长度 $i(i=1,2,\cdots,n-1)$，最多允许切割出 l_i 段长度为 i 的钢条。证明：15.1 节所描述的最优子结构性质不再成立。

length i	1	2	3	4
limit li	2	1	1	1
price pi	15	20	33	36

从图中看出如果切割1个长度为4的钢条，最佳切割方式是切成4条长度为1的钢条可以获得60最大收益，但是我们给每种长度对应的价格做了限制，比如长度为1的钢条最多只能切割2段，那么4>2超出预期。所以我们在不违反切割限制前提下只能选择切成长度分别为1,1和2的钢条作为最佳切割方式，收益为50。由于加了切割限制，所以最优子结构性质不再成立。

15.4

page226:15.4-3

15.4-3 设计 LCS-LENGTH 的带备忘的版本，运行时间为 $O(mn)$。

```
LCS-LENGTH( X, Y, m ,n )
Let b[0..m][0..n] be an new array
for i = 1 to m
    for j = 1 to n
        b[i][j] = -1
return LCS-LENGTH-AUX(X, Y, m ,n ,b ,s)
```

```
LCS-LENGTH-AUX(X, Y, i ,j ,b ,s)
if b[i][j] != -1
    return b[i][j] && s
if i = 0 || j = 0
    b[i][j] = 0
else if X[i] == X[j]
    b[i][j] = LCS-LENGTH-AUX(X, Y, i-1 ,j-1 ,b ,s) + 1
else
    b[i][j] = LCS-LENGTH-AUX(X, Y, i-1 ,j ,b ,s) > LCS-LENGTH-AUX(X, Y, i ,j-1 ,b
,s) ? LCS-LENGTH-AUX(X, Y, i ,j-1 ,b ,s):LCS-LENGTH-AUX(X, Y, i-1 ,j ,b ,s)
return b[i][j]
```

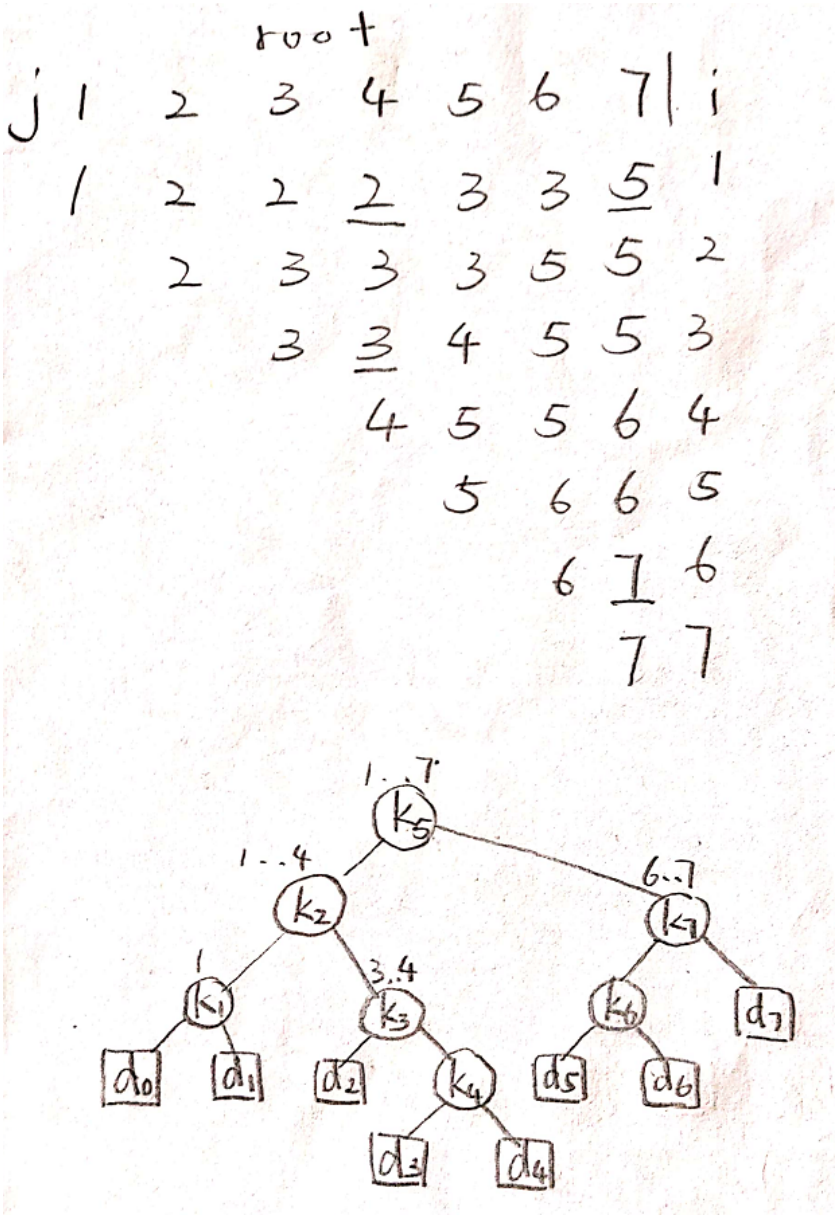
15.5

page230:15.5-2

15.5-2 若 7 个关键字的概率如下所示，求其最优二叉搜索树的结构和代价。

i	0	1	2	3	4	5	6	7
p_i		0.04	0.06	0.08	0.02	0.10	0.12	0.14
q_i	0.06	0.06	0.06	0.06	0.05	0.05	0.05	0.05

结构:



代价 = 3.12

```
def opt_BST(p,q,n):
    # 输入: p[1..n]非叶节点搜索概率, q[0..n]叶节点搜索概率, n个非叶节点

    # e[1..n+1, 0..n]: e[i,j] = 非叶节点k[i..j], 叶节点d[i-1..j]平均搜索代价
    e = np.array(np.zeros((n+2,n+1),float))
    # w[1..n+1, 0..n]: w[i,j] = k[i..j], d[i-1,j]的搜索概率之和
    w = np.array(np.zeros((n+2,n+1),float))
    # root[1..n, 1..n]: root[i,j]记录子树k[i..j],  d[i-1..j]以哪个k为根
    root = np.array(np.zeros((n+1,n+1),int))

    for i in range(1,n+2):
        e[i,i-1] = q[i-1]
        w[i,i-1] = q[i-1]
    for l in range(1,n+1):
        for i in range(1,n-l+2):
            j = i+l-1
            # e[i,j] = infinity
            e[i,j] = 1000
```

```
            w[i,j] = w[i,j-1]+p[j]+q[j]
            for r in range(i,j+1):
                t = e[i,r-1]+e[r+1,j]+w[i,j]
                if t<e[i,j]:
                    e[i,j] = t
                    root[i,j] = r
    print(f'e\n{e}\nw\n{w}\nroot\n{root}')
    return e, root

def test_opt_BST():
    # p[1..n], 注意p[0]用0填充
    # p = [0, 0.15, 0.1, 0.05,0.1,0.2]
    p = [0, 0.04, 0.06, 0.08, 0.02, 0.1,0.12,0.14]
    # q[0..n]
    # q = [0.05,0.1,0.05,0.05,0.05,0.1]
    q = [0.06,0.06,0.06,0.06,0.05,0.05,0.05,0.05]
    n = len(p)-1
    opt_BST(p,q, n)

test_opt_BST()
```

输出:

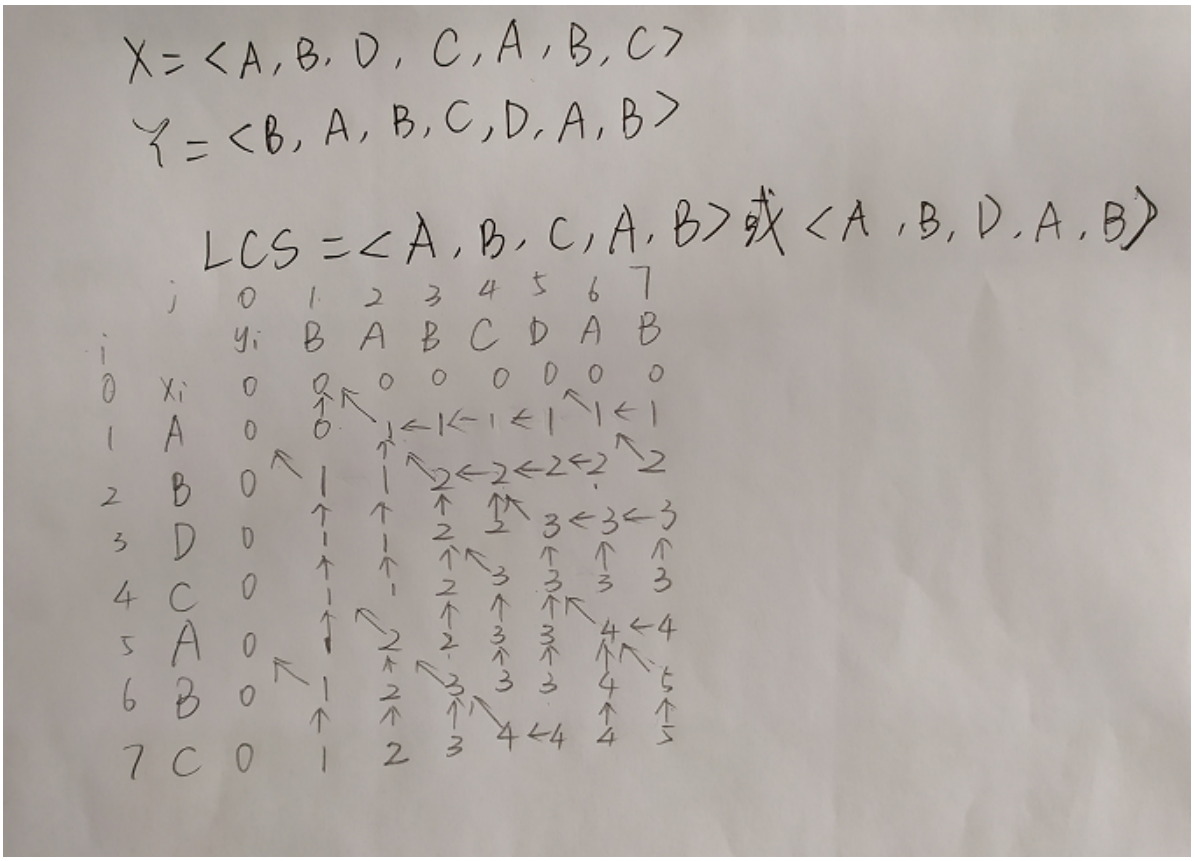
```
e
[[0.   0.   0.   0.   0.   0.   0.   0.  ]
 [0.06 0.28 0.62 1.02 1.34 1.83 2.44 3.12]
 [0.   0.06 0.3  0.68 0.93 1.41 1.96 2.61]
 [0.   0.   0.06 0.32 0.57 1.04 1.48 2.13]
 [0.   0.   0.   0.06 0.24 0.57 1.01 1.55]
 [0.   0.   0.   0.   0.05 0.3  0.72 1.2 ]
 [0.   0.   0.   0.   0.   0.05 0.32 0.78]
 [0.   0.   0.   0.   0.   0.   0.05 0.34]
 [0.   0.   0.   0.   0.   0.   0.   0.05]]
w
[[0.   0.   0.   0.   0.   0.   0.   0.  ]
 [0.06 0.16 0.28 0.42 0.49 0.64 0.81 1.  ]
 [0.   0.06 0.18 0.32 0.39 0.54 0.71 0.9 ]
 [0.   0.   0.06 0.2  0.27 0.42 0.59 0.78]
 [0.   0.   0.   0.06 0.13 0.28 0.45 0.64]
 [0.   0.   0.   0.   0.05 0.2  0.37 0.56]
 [0.   0.   0.   0.   0.   0.05 0.22 0.41]
 [0.   0.   0.   0.   0.   0.   0.05 0.24]
 [0.   0.   0.   0.   0.   0.   0.   0.05]]
root
[[0 0 0 0 0 0 0 0]
 [0 1 2 2 2 3 3 5]
 [0 0 2 3 3 3 5 5]
 [0 0 0 3 3 4 5 5]
 [0 0 0 0 4 5 5 6]
 [0 0 0 0 0 5 6 6]
```

```
 [0 0 0 0 0 0 6 7]
 [0 0 0 0 0 0 0 7]]
```

随测2

题目：已知：X=<A,B,D,C,A,B,C>,Y=<B,A,B,C,D,A,B>，求出X和Y的一个LCS

答案：



(参考自155号同学)