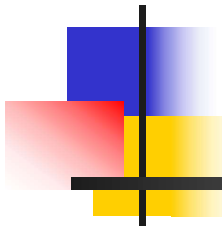

最优二叉搜索树

二叉搜索树

- 二叉排序树或者是一棵空树；或者是具有如下特性的二叉树：
- 若它的左子树不空，则左子树上所有结点的值均小于根结点的值；
- 若它的右子树不空，则右子树上所有结点的值均大于根结点的值；
- **它的左、右子树也都分别是二叉排序树**

问题

- 每个数据元素查找的概率不一样。
- 对键值$K_1,K_2,\cdots,K_n$,查找概率$p_1,p_2,\cdots,p_n$,构造一棵二叉搜索树，使得平均查找时间最小。
- 不失一般性令：$K_1<K_2<\cdots<K_n$
- 若选K_k为根节点，则 $K_1<K_2<\cdots<K_{k-1}$为左子树；
  $K_{k+1}<K_{k+2}<\cdots<K_n$为右子树
- 如何确定根结点？穷举法？

n个结点的二叉树数量h(n)

- n=0时，只有1棵空树
- n=1时，只有1个根结点，只有1棵空树
- n=2时，1个根结点，还有2-1个结点。这一个结点可以分成（1,0），（0,1）两组。即左子树放1个结点，右子树放0个结点；或者左子树放0个结点，右子树放1个结点。即：h(2)=h(0)*h(1)+h(1)*h(0)=2，则能组成2种形态的二叉树。
- n=3时，1个根结点，还有2个结点。这2个结点可以分成(2,0)，(1,1)，(0,2) 3组。即h(3)=h(0)*h(2)+h(1)*h(1)+h(2)*h(0)=5，则能组成5种形态的二叉树。
- ……

n个结点的二叉树数h(n)

- 以此类推，当n≥2时，可组成的二叉树数量为:
- h(n)=h(0)*h(n-1)+h(1)*h(n-2)+...+h(n-1)*h(0)种，

$$h(n)=\sum_{i=1}^{n} h(i-1)h(n-i)$$

类似的问题

- 矩阵链乘： $P=A_1\times A_2\times A_3\times\cdots\cdots\times A_n$，依据乘法结合律，不改变其顺序，只用括号表示成对的乘积，试问有几种括号化的方案？（有多少种不同的计算次序？）
- 一个栈(无穷大)的进栈序列为1，2，3，…，n，有多少个不同的出栈序列?
- 有2n个人排成一行进入剧场。入场费5元。其中只有n个人有一张5元钞票，另外n人只有10元钞票，剧院无其它钞票，问有多少中方法使得只要有10元的人买票，售票处就有5元的钞票找零？(将持5元者到达视作将5元入栈，持10元者到达视作使栈中某5元出栈)

1，2，3，…，i-1, i,… i+1,, n，

一个栈(无穷大)的进栈序列为1，2，3，…，n，有多少个不同的出栈序列?

- 设h(n)为序列个数为n的出栈序列种数。
- 假定，出栈序列中最后一个出栈的元素为i:

1. i取不同值时的情况是相互独立的，即求出每种i最后出栈的情况数后—相加，可得到所要的出栈总数。
2. i最后出栈，那么在i入栈之前，比i小的前i-1个数据均已经入栈并且出栈，这i-1个数据的出栈序列有h(i-1)种；而之后比i大的数据入栈，且都在i之前出栈，因此有h(n-i)种出栈序列。
3. 由于比i小和比i大的值入栈出栈情况是相互独立的，此处可用乘法原则，当最后一个出栈的元素为i时，有h(i-1)*h(n-i)种出栈序列 $(i=1,2,\cdots,n)$

$$h(n)=\sum_{i=1}^{n}h(i-1)h(n-i)$$

一个栈(无穷大)的进栈序列为1，2，3，…，n，有多少个不同的出栈序列?

- 对于每一个数来说，必须进栈一次、出栈一次。
- 进栈设为状态‘1’，出栈设为状态‘0’。n个数据的所有状态对应n个1和n个0组成的2n位二进制数。
- 由于等待入栈的数据按照1‥n的顺序排列、入栈的数据的数量n1大于等于出栈的数据的数量n2(n2≤n1)
- **出栈序列对应的2n位二进制数：任一位置处1的累计数不小于0的累计数**
- 2n位二进制数中填入n个1的方案数为c(2n,n),不填1的其余n位自动填0。从中减去不符合要求（由左而右扫描，0的累计数大于1的累计数）的方案数即为所求。

一个栈(无穷大)的进栈序列为1，2，3，…，n，有多少个不同的出栈序列?

- 不符合要求的数（即：不可能的出栈序列）的特征是由左而右扫描时，必然在某一奇数位2m+1位上首先出现m+1个0的累计数和m个1的累计数，此后的2(n-m)-1位上有n-m个 1和n-m-1个0。
- 如若把后面这2(n-m)-1位上的0和1互换，使之成为n-m个0和n-m-1个1，结果得1个由n+1个0和n-1个1组成的2n位数
- 即一个不合要求的数（**出栈序列**）对应于一个由n+1个0和n-1个1组成的排列。

一个栈(无穷大)的进栈序列为1，2，3，…，n，有多少个不同的出栈序列?

- 任何一个由n+1个0和n-1个1组成的2n位二进制数，由于0的个数多2个，2n为偶数，故必在某一个奇数位上出现0的累计数超过1的累计数。同样在后面部分0和1互换，使之成为由n个0和n个1组成的2n位数，即n+1个0和n-1个1组成的2n位数必对应一个不符合要求的数。
- 因而不合要求的2n位数与n＋1个0，n－1个1组成的排列一一对应。
- 显然，不符合要求的方案数为C(2n,n+1)。由此得出 输出序列的总数目=C(2n,n)-C(2n,n+1)= C(2n,n)-C(2n,n+1)。

$$h(n)=\sum_{i=1}^{n}h(i-1)h(n-i)=C(2n,n)-C(2n,n-1)$$

$$h(n)=\frac{(2n)!}{n!n!}-\frac{(2n)!}{(n-1)!(n+1)!}=\frac{(2n)!}{n!n!}-\frac{n(2n)!}{n!n!(n+1)}$$

$$h(n)=C(2n,n)(1-\frac{n}{n+1})=\frac{C(2n,n)}{n+1}$$

$$n!=\sqrt{2\pi n}\,(\frac{n}{e})^{n}$$

$$h(n)=\frac{1}{n+1}\frac{(2n)!}{n!n!}=\frac{1}{n+1}\frac{\sqrt{4\pi n}(\frac{2n}{e})^{2n}}{\sqrt{2\pi n}(\frac{n}{e})^{n}\sqrt{2\pi n}(\frac{n}{e})^{n}}=\frac{4^{n}}{(n+1)\sqrt{\pi n}}=O(4^{n})$$

方法

- 具有最优子结构性质
- 分治法：重叠子问题----动态规划
- 对键值$K_1,K_2,\cdots,K_n$,查找概率$p_1,p_2,\cdots,p_n$,构造一棵二叉搜索树，使得平均查找时间最小。
- 若$K_1<K_2<\cdots<K_n$,选K_k为根节点，$K_1<K_2<\cdots<K_{k-1}$为左子树(最优二叉搜索树)；$K_{k+1}<K_{k+2}<\cdots<K_n$为右子树(最优二叉搜索树)；

方法

- A(low,high):对键值$K_{low},K_{low+1},\cdots,K_{high}$,查找概率$p_{low}, p_{low+1}, \cdots, p_{high}$,构造一棵二叉搜索树，使得平均查找时间最小。
- 选K_k为根结点，则$K_{low},K_{low+1},\cdots,K_{k-1}$为其的左子树， $K_{k+1},K_{k+2},\cdots,K_{high}$为其的右子树,根据最优子结构性质，左右子树均为最优二叉搜索树
- A(low, k-1)为$K_{low},K_{low+1},\cdots,K_{k-1}$对应的最优二叉搜索树的平均检索时间

$$A(low, k-1) = \sum_{i=low}^{k-1} p_i C_i^L$$

- A(k+1, high)为$K_{k+1},K_{k+2},\cdots,K_{high}$对应的最优二叉搜索树的平均检索时间

$$A(k+1, high) = \sum_{i=k+1}^{high} p_i C_i^R$$

- A(low, high, k)为$K_{low},K_{low+1},\cdots,K_{high}$选$K_k$为根结点对应的最优二叉搜索树的平均检索时间

$$A(low, high, k) = \sum_{i=low}^{high} p_i C_i = p(low, high) + A(low, k-1) + A(k+1, high)$$

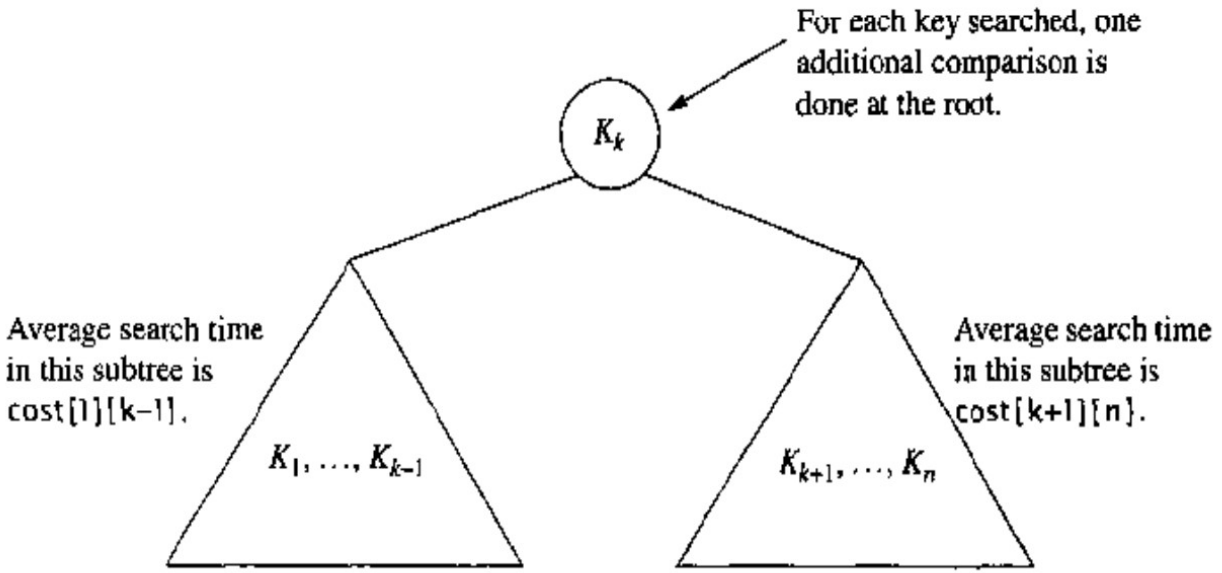

Figure 10.4 Choosing K_k as the root

$$A(low, high, k) = \sum_{i=low}^{high} p_i C_i = \sum_{i=low}^{k-1} p_i C_i + p_k + \sum_{i=k}^{high} p_i C_i$$

$$A(low, high, k) = \sum_{i=low}^{k-1} p_i (C_i^L + 1) + p_k + \sum_{i=k+1}^{high} p_i (C_i^R + 1)$$

$$A(low, high, k) = \sum_{i=low}^{k-1} p_i C_i^L + \sum_{i=low}^{k-1} p_i + p_k + \sum_{i=k+1}^{high} p_i C_i^R + \sum_{i=k+1}^{high} p_i$$

$$= p(low, high) + A(low, k-1) + A(k+1, high)$$

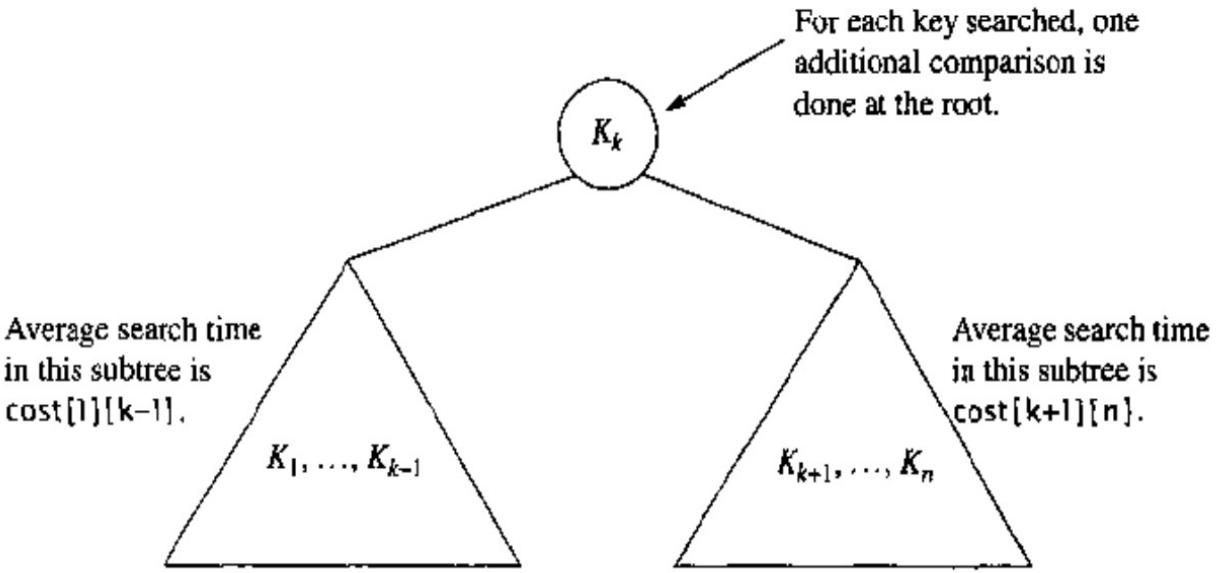

Figure 10.4 Choosing K_k as the root

$$A(low, high, k) = \sum_{i=low}^{high} p_i C_i = \sum_{i=low}^{k-1} p_i C_i + p_k + \sum_{i=k}^{high} p_i C_i$$

$$A(low, high, k) = \sum_{i=low}^{k-1} p_i C_i^L + \sum_{i=low}^{k-1} p_i + p_k + \sum_{i=k+1}^{high} p_i C_i^R + \sum_{i=k+1}^{high} p_i$$

$$= p(low, high) + A(low, k-1) + A(k+1, high)$$

A(low, high)=min{A(low, high, k)|low≤k≤high}

cost[i][j]=A(i,j),K_i,K_{i+1},…,K_j的最优二叉搜索树，root[i][j]为根节点

假定各元素的查找概率分别为A（0.20），B（0.24），C（0.16），D（0.28），E（0.04）和F（0.08），要得到最优二叉搜索树

	COST数组							ROOT数组						
	0	1	2	3	4	5	6	0	1	2	3	4	5	6
1														
2														
3														
4														
5														
6														
7														

从空树开始，依次构造1个节点的最优二叉搜索树，2 个节点的最优二叉搜索树，……

cost[i][j]=A(i,j),K_i,K_{i+1},…,K_j的最优二叉搜索树，root[i][j]为根节点

假定各元素的查找概率分别为A（0.20），B（0.24），C（0.16），D（0.28），E（0.04）和F（0.08），要得到最优二叉搜索树

p_1=0.20，p_2=0.24，p_3=0.16，p_4=0.28，p_5=0.04，p_6=0.08

	COST数组							ROOT数组						
	0	1	2	3	4	5	6	0	1	2	3	4	5	6
1														
2														
3														
4														
5														
6														
7														

从空树开始，依次构造1个节点的最优二叉搜索树，2 个节点的最优二叉搜索树，……

cost[i][j]=A(i,j),K_i,K_{i+1},…,K_j的最优二叉搜索树，root[i][j]为根节点

假定各元素的查找概率分别为A（0.20），B（0.24），C（0.16），D（0.28），E（0.04）和F（0.08），要得到最优二叉搜索树

p_1=0.20，p_2=0.24，p_3=0.16，p_4=0.28，p_5=0.04，p_6=0.08

	COST数组							ROOT数组						
	0	1	2	3	4	5	6	0	1	2	3	4	5	6
1		**0.2**							**1**					
2			**0.24**							**2**				
3				**0.16**							**3**			
4					**0.28**							**4**		
5						**0.04**							**5**	
6							**0.08**							**6**
7														

从空树开始，依次构造1个节点的最优二叉搜索树，2 个节点的最优二叉搜索树，……

cost[i][j]=A(i,j),K_i,K_{i+1},…,K_j的最优二叉搜索树，root[i][j]为根节点

假定各元素的查找概率分别为A（0.20），B（0.24），C（0.16），D（0.28），E（0.04）和F（0.08），要得到最优二叉搜索树

p_1=0.20，p_2=0.24，p_3=0.16，p_4=0.28，p_5=0.04，p_6=0.08

	COST数组							ROOT数组						
	0	1	2	3	4	5	6	0	1	2	3	4	5	6
1		**0.2**							**1**					
2			**0.24**							**2**				
3				**0.16**							**3**			
4					**0.28**							**4**		
5						**0.04**							**5**	
6							**0.08**							**6**
7														

A(low，high)=min{A(low，high, r)|low≤r≤high}　A(1，2)=min{A(1,2, r)|1≤r≤2}

A(1，2, r)=p_r+p(1,r-1) +A(1,r-1) +p(r+1,2)+A(r+1,2)

= p(1,2) + A(1,r-1)+A(r+1,2)

A(i，i-1)代表空树

A(1，2, r)=p(1,2) + A(1,r-1)+A(r+1,2)=0.44+ A(1,r-1)+A(r+1,2)
r=1: 0.44+A(1,0)+A(2,2)=0.68; r=2: 0.44+A(1,1)+A(3,2)=0.64

假定各元素的查找概率分别为A（0.20），B（0.24），C（0.16），D（0.28），E（0.04）和F（0.08），要得到最优二叉搜索树

p_1=0.20，p_2=0.24，p_3=0.16，p_4=0.28，p_5=0.04，p_6=0.08

	COST数组							ROOT数组						
	0	1	2	3	4	5	6	0	1	2	3	4	5	6
1	0	**0.2**						-1	1					
2		**0**	**0.24**						-1	2				
3			**0**	**0.16**						-1	3			
4				0	**0.28**						-1	4		
5					**0**	**0.04**						-1	5	
6						0	**0.08**						-1	6
7							0							-1

A(low，high)=min{A(low，high, r)|low≤r≤high}　A(1，2)=min{A(1,2, r)|1≤r≤2}

A(1，2, r)=p_r+p(1,r-1) +A(1,r-1) +p(r+1,2)+A(r+1,2)
= p(1,2) + A(1,r-1)+A(r+1,2)

A(i，i-1)代表空树

A(1，2, r)=p(1,2) + A(1,r-1)+A(r+1,2)=0.44+ A(1,r-1)+A(r+1,2)

r=1: 0.44+A(1,0)+A(2,2)=0.68; r=2: 0.44+A(1,1)+A(3,2)=0.64

假定各元素的查找概率分别为A（0.20），B（0.24），C（0.16），D（0.28），E（0.04）和F（0.08），要得到最优二叉搜索树

p_1=0.20，p_2=0.24，p_3=0.16，p_4=0.28，p_5=0.04，p_6=0.08

	COST数组							ROOT数组						
	0	1	2	3	4	5	6	0	1	2	3	4	5	6
1	0	**0.2**	**0.64**					-1	1	2				
2		**0**	**0.24**						-1	2				
3			**0**	**0.16**						-1	3			
4				0	**0.28**						-1	4		
5					**0**	**0.04**						-1	5	
6						0	**0.08**						-1	6
7							0							-1

A(low，high)=min{A(low，high, r)|low≤r≤high}　A(1，2)=min{A(1,2, r)|1≤r≤2}

A(1，2, r)=p_r+p(1,r-1) +A(1,r-1) +p(r+1,2)+A(r+1,2)

= p(1,2) + A(1,r-1)+A(r+1,2)

A(i，i-1)代表空树

假定各元素的查找概率分别为A（0.20），B（0.24），C（0.16），D（0.28），E（0.04）和F（0.08），要得到最优二叉搜索树

p_1=0.20，p_2=0.24，p_3=0.16，p_4=0.28，p_5=0.04，p_6=0.08

	COST数组							ROOT数组						
	0	1	2	3	4	5	6	0	1	2	3	4	5	6
1	0	**0.2**	**0.64**					-1	1	2				
2		**0**	**0.24**	**0.56**					-1	2	2			
3			**0**	**0.16**						-1	3			
4				0	**0.28**						-1	4		
5					**0**	**0.04**						-1	5	
6						0	**0.08**						-1	6
7							0							-1

A(low，high)=min{A(low，high, r)|low≤r≤high}

A(2，3, r)=p(2,3) + A(2,r-1)+A(r+1,3)=0.4+ A(2,r-1)+A(r+1,3)

r=2: 0.4+A(2,1)+A(3,3)=0.56；　r=3: 0.4+A(2,2)+A(4,3)=0.64

A(i，i-1)代表空树

假定各元素的查找概率分别为A（0.20），B（0.24），C（0.16），D（0.28），E（0.04）和F（0.08），要得到最优二叉搜索树

p_1=0.20，p_2=0.24，p_3=0.16，p_4=0.28，p_5=0.04，p_6=0.08

	COST数组							ROOT数组						
	0	1	2	3	4	5	6	0	1	2	3	4	5	6
1	0	**0.2**	**0.64**					-1	1	2				
2		**0**	**0.24**	**0.56**					-1	2	2			
3			**0**	**0.16**	**0.6**					-1	3	4		
4				0	**0.28**						-1	4		
5					**0**	**0.04**						-1	5	
6						0	**0.08**						-1	6
7							0							-1

A(low，high)=min{A(low，high, r)|low≤r≤high}

A(i，i-1)代表空树

假定各元素的查找概率分别为A（0.20），B（0.24），C（0.16），D（0.28），E（0.04）和F（0.08），要得到最优二叉搜索树

p_1=0.20，p_2=0.24，p_3=0.16，p_4=0.28，p_5=0.04，p_6=0.08

	COST数组							ROOT数组						
	0	1	2	3	4	5	6	0	1	2	3	4	5	6
1	0	**0.2**	**0.64**					-1	1	2				
2		**0**	**0.24**	**0.56**					-1	2	2			
3			**0**	**0.16**	**0.6**					-1	3	4		
4				0	**0.28**	**0.36**					-1	4	4	
5					**0**	**0.04**						-1	5	
6						0	**0.08**						-1	6
7							0							-1

A(low，high)=min{A(low，high, r)|low≤r≤high}

A(i，i-1)代表空树

假定各元素的查找概率分别为A（0.20），B（0.24），C（0.16），D（0.28），E（0.04）和F（0.08），要得到最优二叉搜索树

p_1=0.20，p_2=0.24，p_3=0.16，p_4=0.28，p_5=0.04，p_6=0.08

	COST数组							ROOT数组						
	0	1	2	3	4	5	6	0	1	2	3	4	5	6
1	0	**0.2**	**0.64**					-1	1	2				
2		**0**	**0.24**	**0.56**					-1	2	2			
3			**0**	**0.16**	**0.6**					-1	3	4		
4				0	**0.28**	**0.36**					-1	4	4	
5					**0**	**0.04**	**0.16**					-1	5	6
6						0	**0.08**						-1	6
7							0							-1

A(low，high)=min{A(low，high, r)|low≤r≤high}

A(i，i-1)代表空树

假定各元素的查找概率分别为A（0.20），B（0.24），C（0.16），D（0.28），E（0.04）和F（0.08），要得到最优二叉搜索树

p_1=0.20，p_2=0.24，p_3=0.16，p_4=0.28，p_5=0.04，p_6=0.08

	COST数组							ROOT数组						
	0	1	2	3	4	5	6	0	1	2	3	4	5	6
1	0	**0.2**	**0.64**	**0.96**				-1	1	2	2			
2		**0**	**0.24**	**0.56**					-1	2	2			
3			**0**	**0.16**	**0.6**					-1	3	4		
4				0	**0.28**	**0.36**					-1	4	4	
5					**0**	**0.04**	**0.16**					-1	5	6
6						0	**0.08**						-1	6
7							0							-1

A(low，high)=min{A(low，high, r)|low≤r≤high}

A(1，3, r)=p(1,3) + A(1,r-1)+A(r+1,3)=0.6+ A(1,r-1)+A(r+1,3)

r=1: 0.6+A(1,0)+A(2,3)=1.16; r=2: 0.6+A(1,1)+A(3,3)=0.96;

r=3: 0.6+A(1,2)+A(4,3)=1.24

A(i，i-1)代表空树

假定各元素的查找概率分别为A（0.20），B（0.24），C（0.16），D（0.28），E（0.04）和F（0.08），要得到最优二叉搜索树

p_1=0.20，p_2=0.24，p_3=0.16，p_4=0.28，p_5=0.04，p_6=0.08

	COST数组							ROOT数组						
	0	1	2	3	4	5	6	0	1	2	3	4	5	6
1	0	**0.2**	**0.64**	**0.96**				-1	1	2	2			
2		**0**	**0.24**	**0.56**					-1	2	2			
3			**0**	**0.16**	**0.6**					-1	3	4		
4				0	**0.28**	**0.36**					-1	4	4	
5					**0**	**0.04**	**0.16**					-1	5	6
6						0	**0.08**						-1	6
7							0							-1

A(low，high)=min{A(low，high, r)|low≤r≤high}

A(1，3, r)=p(1,3) + A(1,r-1)+A(r+1,3)=0.6+ A(1,r-1)+A(r+1,3)

r=1: 0.6+A(1,0)+A(2,3)=1.16; r=2: 0.6+A(1,1)+A(3,3)=0.96;

r=3: 0.6+A(1,2)+A(4,3)=1.24

A(i，i-1)代表空树

假定各元素的查找概率分别为A（0.20），B（0.24），C（0.16），D（0.28），E（0.04）和F（0.08），要得到最优二叉搜索树

p_1=0.20，p_2=0.24，p_3=0.16，p_4=0.28，p_5=0.04，p_6=0.08

	COST数组							ROOT数组						
	0	1	2	3	4	5	6	0	1	2	3	4	5	6
1	**0**	**0.2**	**0.64**	**0.96**	**1.68**	**1.8**	**2.08**	**-1**	**1**	**2**	**2**	**2**	**2**	**2**
2		**0**	**0.24**	**0.56**	**1.2**	**1.32**	**1.52**		**-1**	**2**	**2**	**2**	**3**	**4**
3			**0**	**0.16**	**0.6**	**0.68**	**0.88**			**-1**	**3**	**4**	**4**	**4**
4				**0**	**0.28**	**0.36**	**0.56**				**-1**	**4**	**4**	**4**
5					**0**	**0.04**	**0.16**					**-1**	**5**	**6**
6						**0**	**0.08**						**-1**	**6**
7							**0**							**-1**

从空树开始，依次构造1个节点的最优二叉搜索树，2 个节点的最优二叉搜索树，……

假定各元素的查找概率分别为A（0.20），B（0.24），C（0.16），D（0.28），E（0.04）和F（0.08），要得到最优二叉搜索树

p_1=0.20，p_2=0.24，p_3=0.16，p_4=0.28，p_5=0.04，p_6=0.08

	COST数组							ROOT数组						
	0	1	2	3	4	5	6	0	1	2	3	4	5	6
1	**0**	**0.2**	**0.64**	**0.96**	**1.68**	**1.8**	**2.08**	**-1**	**1**	**2**	**2**	**2**	**2**	**2**
2		**0**	**0.24**	**0.56**	**1.2**	**1.32**	**1.52**		**-1**	**2**	**2**	**2**	**3**	**4**
3			**0**	**0.16**	**0.6**	**0.68**	**0.88**			**-1**	**3**	**4**	**4**	**4**
4				**0**	**0.28**	**0.36**	**0.56**				**-1**	**4**	**4**	**4**
5					**0**	**0.04**	**0.16**					**-1**	**5**	**6**
6						**0**	**0.08**						**-1**	**6**
7							**0**							**-1**

从空树开始，依次构造1个节点的最优二叉搜索树，2 个节点的最优二叉搜索树，……

```
optimalBST(p[ ], n, cost[ ][ ], root[ ][ ]){
    for(i=n+1; i≥1; i--)
        {cost[i][i-1]=0;  root[i][i-1]=-1;}
    for(i=1; i<=n; i++)
        { cost[i][i]=p[i];   root[i][i]=i;}
    for(L=2; L≤n; L++)
        for (low= 1; low <= n - L+1; low++) {
            high=low+L-1
            bestCost=p(low,high)+cost[low][low-1]+cost[low+1][high];
            for (r = low+1; r <=high; r++){
                rCost=p(low,high)+cost[low][r-1]+cost[r+1][high];
                if (rCost<bestCost)
                        { bestCost=rCost;    bestRoot=r;}}
            cost[low][high]=bestCost;
            root[low][high]=bestRoot;}
    return;    }
```

- $\Theta(n^3)$