

树结构习题

蒋雨芮

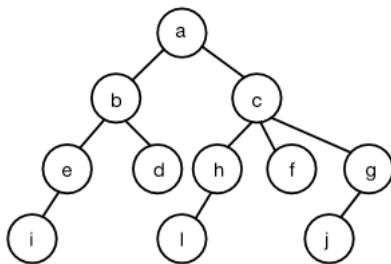
2020 年 10 月 16 日

1

1、假设在树中，结点 x 是结点 y 的双亲时，用 (x,y) 来表示树边。已知一棵树的树边集合为 $\{(e,i), (b,e), (b,d), (a,b), (g,i), (c,g), (c,f), (h,i), (c,h), (a,c)\}$ ，用树型表示法表示该树，并回答下列问题：

- 1) 哪个是根结点? 哪些是叶子结点? 哪个是 g 的双亲? 哪些是 g 的祖先? 哪些是 g 的孩子? 哪些是 e 的子孙? 哪些是 e 的兄弟? 哪些是 f 的兄弟?
- 2) b 和 n 的层次各是多少? 树的深度是多少? 以结点 c 为根的子树的深度是多少?

该树型如下：



1.1

a 是根节点； i,d,l,f,j 是叶子节点； c 是 g 的双亲； c,a 是 g 的祖先； j 是 g 的孩子； i 是 e 的子孙； d 是 e 的兄弟； h,g 是 f 的兄弟。

1.2

b 的层次是第二层；树深度为4；以 c 为根的子树深度为3。

2

2、一棵深度为h的满k叉树有如下性质：第h层上的结点都是叶子结点，其余各层上每个结点都有k棵非空子树。如果按层次顺序(同层自左至右)从1开始对全部结点编号，问：

- 1) 各层的结点数是多少?
- 2) 编号为i的结点的双亲结点(若存在)的编号是多少?
- 3) 编号为i的结点的第j个孩子结点(若存在)的编号是多少?
- 4) 编号为i的结点的有右兄弟的条件是什么? 其右兄弟的编号是多少?

2.1

$$k^{n-1}$$

2.2

假设该编号为i的结点的双亲节点j在第n层，有前n-1层节点总数 $S_{n-1} = \frac{k^{n-1}-1}{k-1}$ ， $S_n = \frac{k^n-1}{k-1}$ ；由此可知，第n层在j之前的节点数目 $N = j - S_{n-1} - 1$ ，所以节点j-1的最右孩子节点的编号为 $k \times N + S_n = k \times j - k + 1$

所以j的孩子的编号i的范围为：

$$k \times j - k + 2 \leq i \leq k \times j + 1$$

$$k \times j - k \leq i - 2 \leq k \times j - 1$$

$$k \times (j - 1) \leq i - 2 \leq k \times (j - 1) + k + 1$$

$$i - 1 \leq \frac{i-2}{k} \leq i - 1 + \frac{k-1}{k}$$

$$j = \lfloor \frac{i-2}{k} \rfloor + 1$$

2.3

由(2)中推出的结论知，结点i的第j个孩子的编号为 $k \times i + k + 1 + j$

2.4

由(2)中推出的结论

条件为 $k > 1, i > 1, \frac{i-1}{k} \neq 0$

右兄弟的编号为 $[i + 1, k \times (\lfloor \frac{i-2}{k} \rfloor + 1) + 1]$

3

3、设有如图1所示的二叉树。

1) 分别用顺序存储方法和链接存储方法画出该二叉树的存储结构。

2) 写出该二叉树的先序、中序、后序遍历序列。

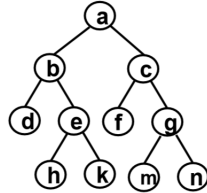


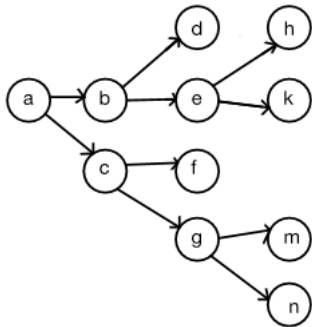
图1 二叉树

3.1

顺序存储:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	a	b	c	d	e	f	g			h	k			m	n

链接存储:



3.2

先序: a,b,d,e,h,k,c,f,g,m,n

中序: d,b,h,e,k,a,f,c,m,g,n

后序: d,h,k,e,b,f,m,n,g,c,a

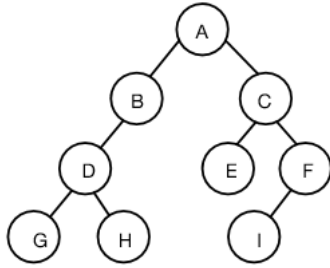
4

4、已知一棵二叉树的先序遍历序列和中序遍历序列分

别为ABDGHCEFI和GDHBAECIF, 请画出这棵二叉树,

然后给出该树的后序遍历序列。

该树型如下：

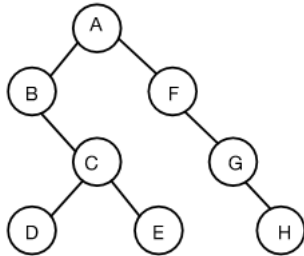


后序：G,H,D,B,E,I,F,C,A

5


5、设一棵二叉树的中序遍历序列和后序遍历序列分别为BDCEAFHG和DECBHGFA，请画出这棵二叉树，然后给出该树的先序序列。

该树型如下：

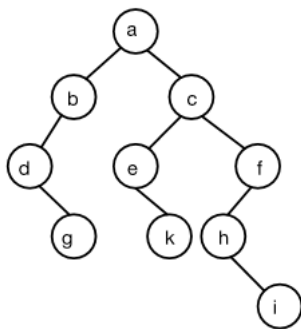


先序：A,B,C,D,E,F,G,H

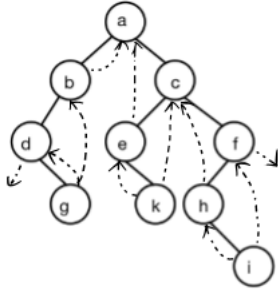
6

6、已知一棵二叉树的中序遍历序列和后序遍历序列分别为  dgbaekchif和gdbkeihfca，请画出这棵二叉树对应的中序线索树和后序线索树。

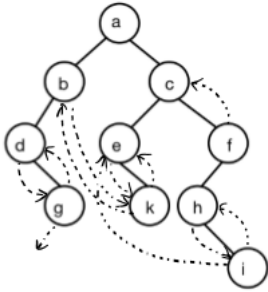
该树型如下：



中序线索树如下：



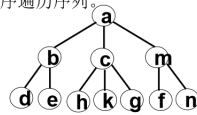
后序线索树如下：



7

7、设有一棵树，如图2所示。

- 1) 请分别用双亲表示法、孩子表示法、孩子兄弟表示法给出该树的存储结构。
- 2) 请给出该树的先序遍历序列和后序遍历序列。
- 3) 请将这棵树转换成二叉树。



7.1

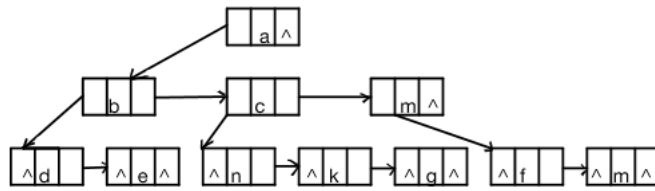
双亲表示法:

孩子表示法:

下标	info	paren
0	a	-1
1	b	0
2	c	0
3	d	1
4	e	1
5	f	9
6	g	2
7	h	2
8	k	2
9	m	0
10	n	9

下标	info	
0	a	-> 1 -> 2 -> 9
1	b	-> 3 -> 4
2	c	-> 7 -> 8 -> 6
3	d	-> ^
4	e	-> ^
5	f	-> ^
6	g	-> ^
7	h	-> ^
8	k	-> ^
9	m	-> 5 -> 10
10	n	-> ^

孩子兄弟表示法:



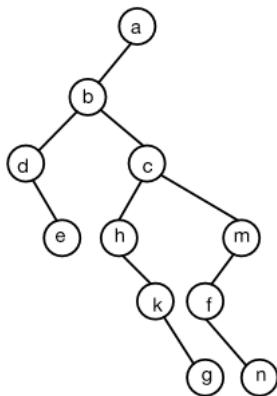
7.2

先序遍历: a,b,d,e,c,h,k,g,m,f,n

后序遍历: d,e,b,h,k,g,c,f,n,m,a

7.3

由于该树的先序遍历序列和后序遍历序列分别是对应二叉树的先序遍历和中序遍历, 故由上述序列可唯一确定一颗二叉树如下图所示:



8

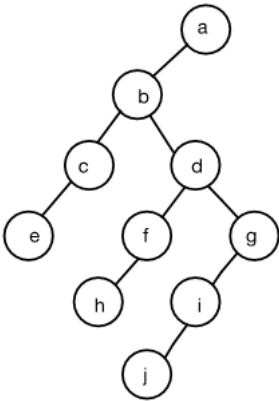
8、设二叉树t的存储结构如图3所示。其中t为树根结点的指针，Left和Right分别为结点的左、右孩子指针域，data为结点的数据域，请完成下列各题：

- 1) 画出二叉树t的逻辑结构
- 2) 写出按前序、中序和后序遍历二叉树t所得到的结点序列。

	1	2	3	4	5	6	7	8	9	10
Left	0	0	2	3	7	5	8	0	10	1
Data	j	h	f	d	b	a	c	e	g	i
Right	0	0	0	9	4	0	0	0	0	0

8.1

该树的逻辑结构如下：



8.2

前序： a,b,c,e,d,f,h,g,i,j
中序： e,c,b,a,h,f,d,j,i,g
后序： e,c,h,f,j,i,g,d,b,a

9

9、表1中m,n分别是一棵二叉树中两个结点，行号i=1,2,3,4分别表示四种m,n的相对关系，列号j=1,2,3分别表示在前序、中序和后序遍历中m,n之间的先后次序关系，要求在i,j所表示的关系能够同时发生的方格内打“√”。

i \ j	1	2	3
1 n在m的左边			
2 n在m的右边			
3 n是m的祖先			
4 n是m的儿子			

	前序遍历n先被访问	中序遍历n先被访问	后序遍历n先被访问
n在m的左边	√	√	√
n在m的右边			
n是m的祖先	√	√	
n是m的儿子		√	√

10

10、假设二叉树采用二叉链表存储，编写一个后序遍历二叉树的非递归算法。



后序遍历实际上就是遇到一个结点，先访问它的左右子树，相当于当该结点被“遇见”两次之后即输出。故可用栈结构来辅助，并引入一个标志来标记访问次数，访问两次即输出。

```

struct Node{           //定义一个结构 Node,成员为二叉树的节点 treeNode 和一个标志 flag
    TreeNode *tree;
    int flag;
    Node(TreeNode* t):tree(t),flag(0){}
};

void postOrderTraversal(TreeNode* root) {
    stack<Node> s;
    if(root)             //将根节点入栈
        s.push(Node(root));

    while(!s.empty()){   //取出栈顶元素，如果该元素已“出栈”两次，则输出
        if(s.top().flag==2){
            cout<<s.top().tree->val;
            s.pop();
            continue;
        }

        if(!s.empty())   //栈顶元素“出栈”
            s.top().flag++;

        Node temp=s.top();
        if(temp.flag==1){ //若是该元素第一次“出栈”，则访问它的左孩子
            if(temp.tree->left) //左孩子不为空，则将左孩子入栈
                s.push(Node(temp.tree->left));
        }
        else if(temp.flag==2){ //若是该元素第二次“出栈”，则访问它的右孩子
            if(temp.tree->right) //右孩子不为空，则将右孩子入栈
                s.push(Node(temp.tree->right));
        }
    }
}

```


11

11、在二叉树中查找值为x的结点，设计打印值为x的结点的双亲的算法。

```
Node* fun(Node* T,int x){
    if(T==NULL)                //若该树为空，则返回 NULL
        return NULL;

    bool f1= true,f2=true;      //判断该结点的左右孩子是否等于 x
    if(T->left==NULL||T->left->value!=x)
        f1=false;
    if(T->right==NULL||T->right->value!=x)
        f2=false;
    if(f1||f2){
        cout<<T->value<<endl;    //若某孩子等于 x,则打印并返回该节点
        return T;
    }
    else{                        //若该节点的孩子都不等于 x，则递归调用该函数
        Node* T1,*T2;
        T1=fun(T->left,x);
        T2=fun(T->right,x);

        return T1?T1:T2;        //返回两孩子中不为空的结点,若都为空，则返回 NULL
    }
}
```

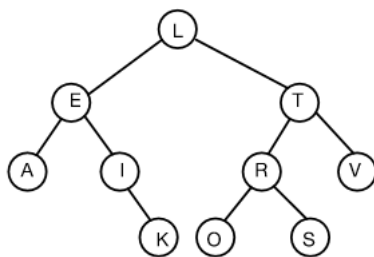
该算法有一个问题是，默认了根节点的值不是要找的x，若是则出现错误。

12

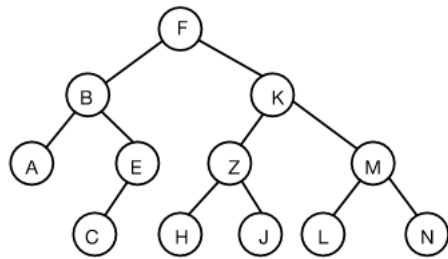
12、以下列顺序插入数据元素，并用边建立边平衡的方式建立AVL二叉排序树。

- 1) A, V, L, T, R, E, I, S, O, K
- 2) A, Z, B, Y, C, X, D, W, E, V, F

12.1



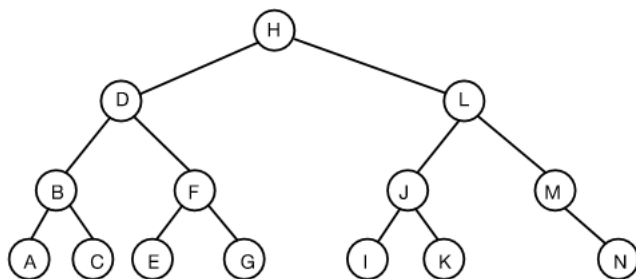
12.2



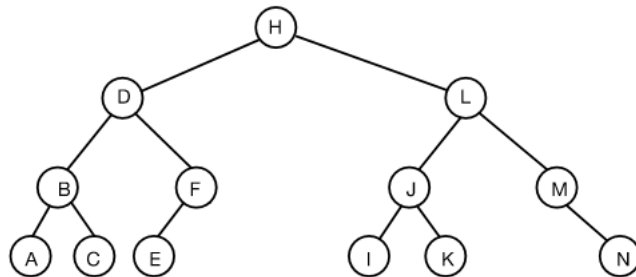
13

13、以A-N为关键字，建立AVL树，并对建立好的树，图示依次删除G, D, N 结点。

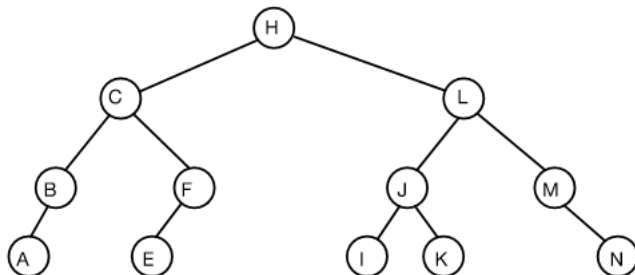
该AVL树型如下：



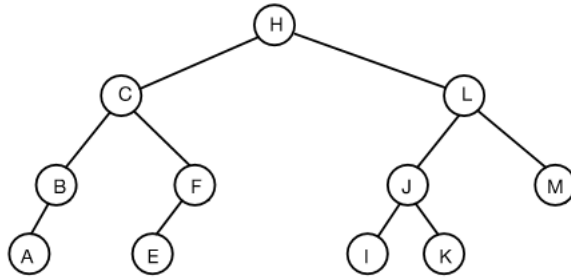
删除结点G后为：



删除结点D后为：



删除结点N后为：

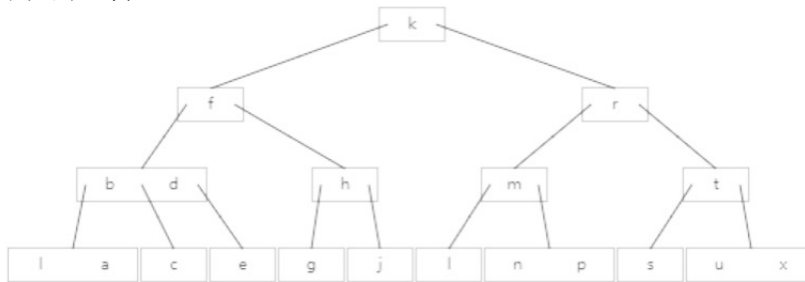


14

14、将下列元素以给定的顺序插入到初始为空的阶为3，7的B-树中。

a g f b k d h m j e s l r x c l n t u p

阶为3的B-树：



阶为7的B-树：



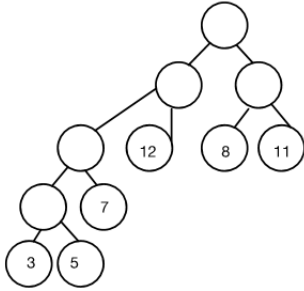
15

15、当以适当的次序插入元素时，产生的高度为3（即为3层）的5阶B-树的最少元素数目是多少？

最少17个元素

16

16、设给定权值集合 $w=\{3, 5, 7, 8, 11, 12\}$ ，请构造关于 w 的一棵huffman树，并求其加权路径长度WPL。



由画出的Huffman树得 $WPL = 3 \times 4 + 5 \times 4 + 7 \times 3 + 12 \times 2 + 8 \times 2 + 11 \times 2 = 115$

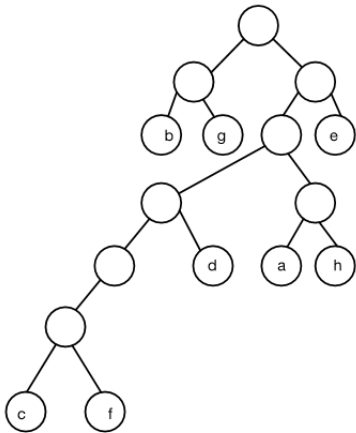
17

17、假设用于通信的电文是由字符集 $\{a, b, c, d, e, f, g, h\}$ 中的字符构成，这8个字符在电文中出现的概率分别为 $\{0.07, 0.19, 0.02, 0.06, 0.32, 0.03, 0.21, 0.10\}$ 。

- 1) 请画出对应的huffman树(按左子树根结点的权小于等于右子树根结点的权的次序构造)。
- 2) 求出每个字符的huffman编码。

17.1

对应的Huffman树如下：



17.2

由Huffman树得：

a: 1010	b: 00	c: 10000	d: 1001
e: 11	f: 10001	g: 01	h: 1011