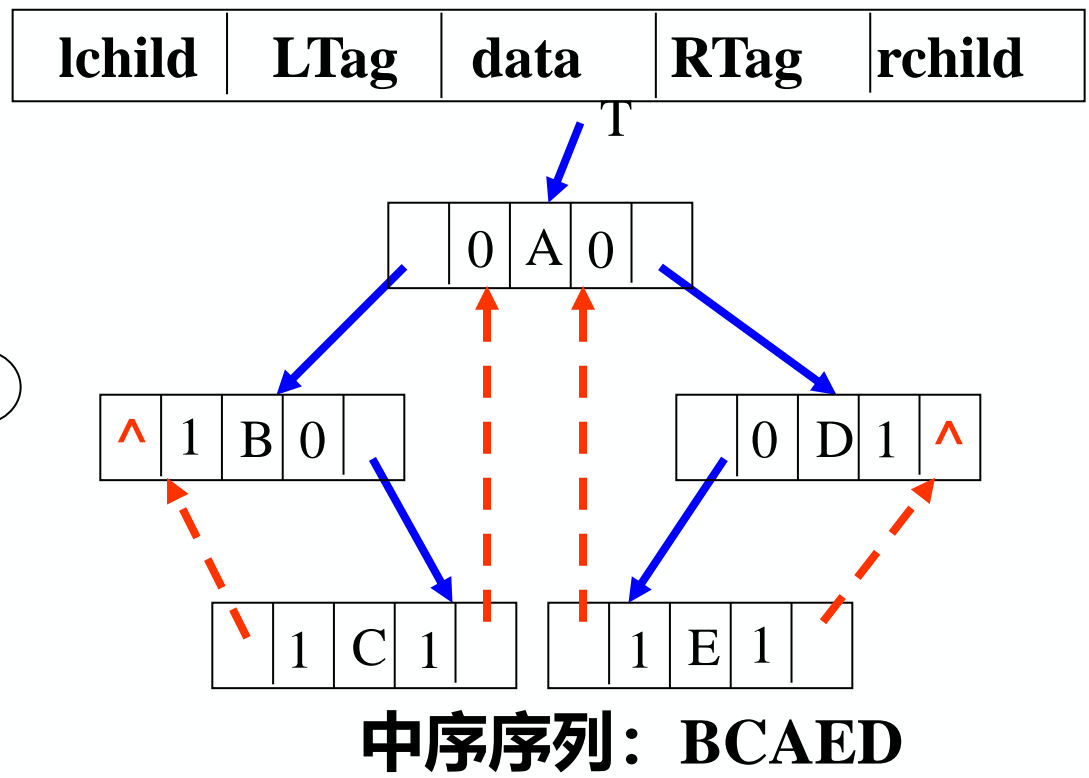
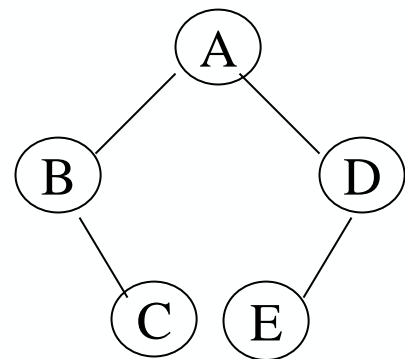
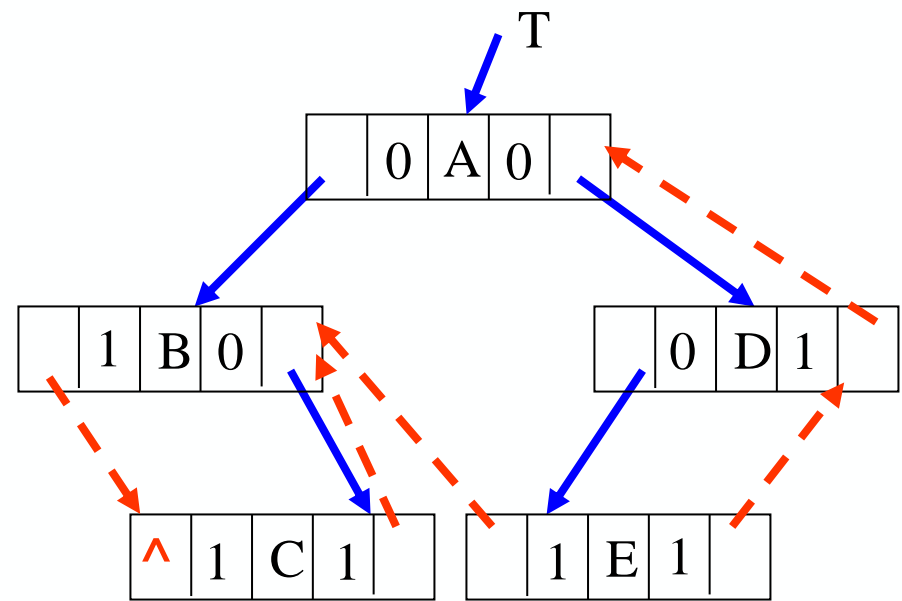
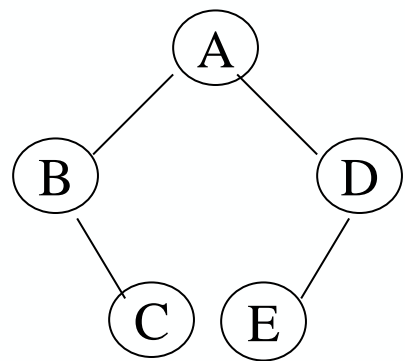


中序线索二叉树



后序线索二叉树

lchild	LTag	data	RTag	rchild
--------	------	------	------	--------



后序序列：CBEDA

线索化二叉树的几个术语

线索：指向结点前驱和后继的指针

线索链表：加上线索二叉链表

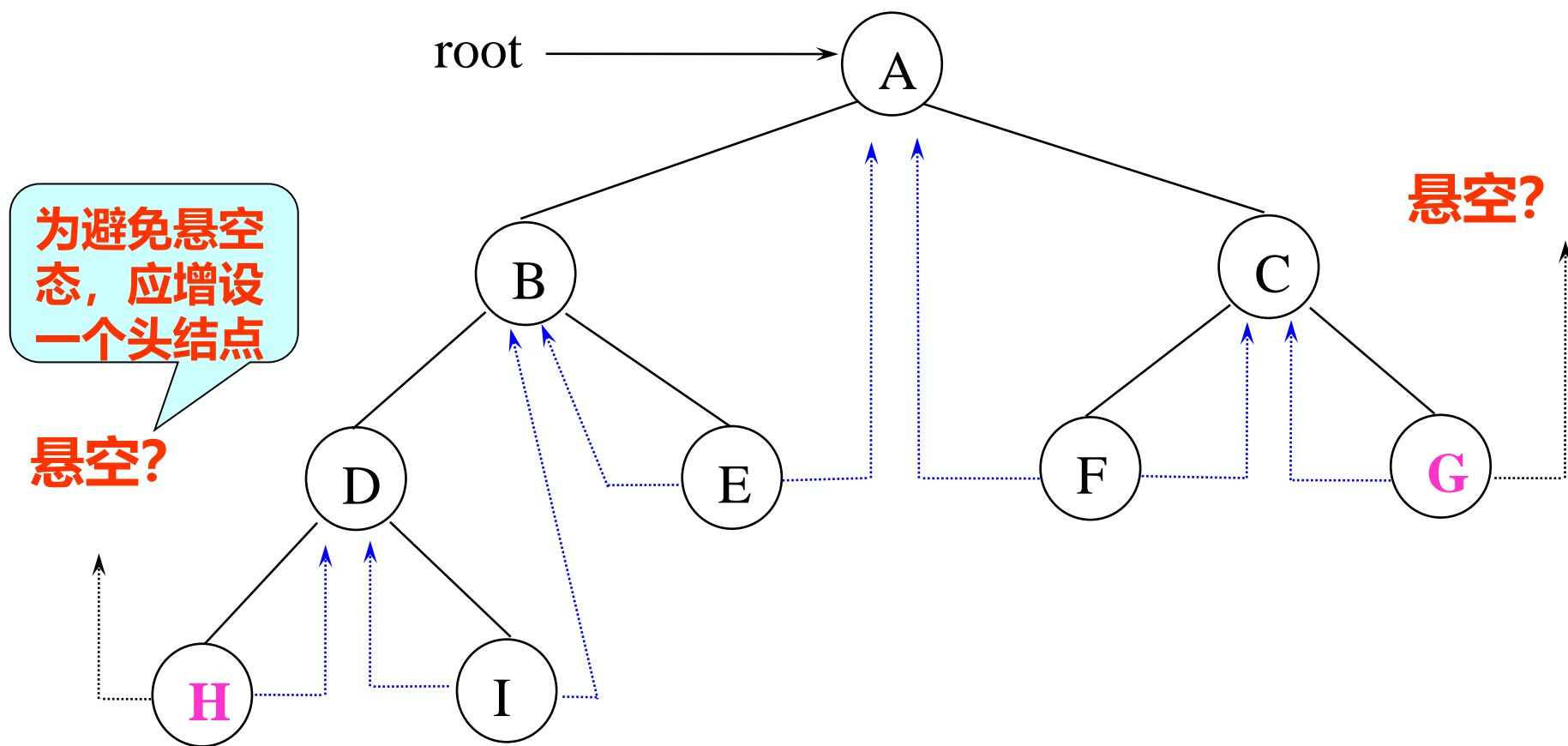
线索二叉树：加上线索的二叉树（图形式样）

线索化：对二叉树以某种次序遍历使其变为线索二叉树的过程

练习

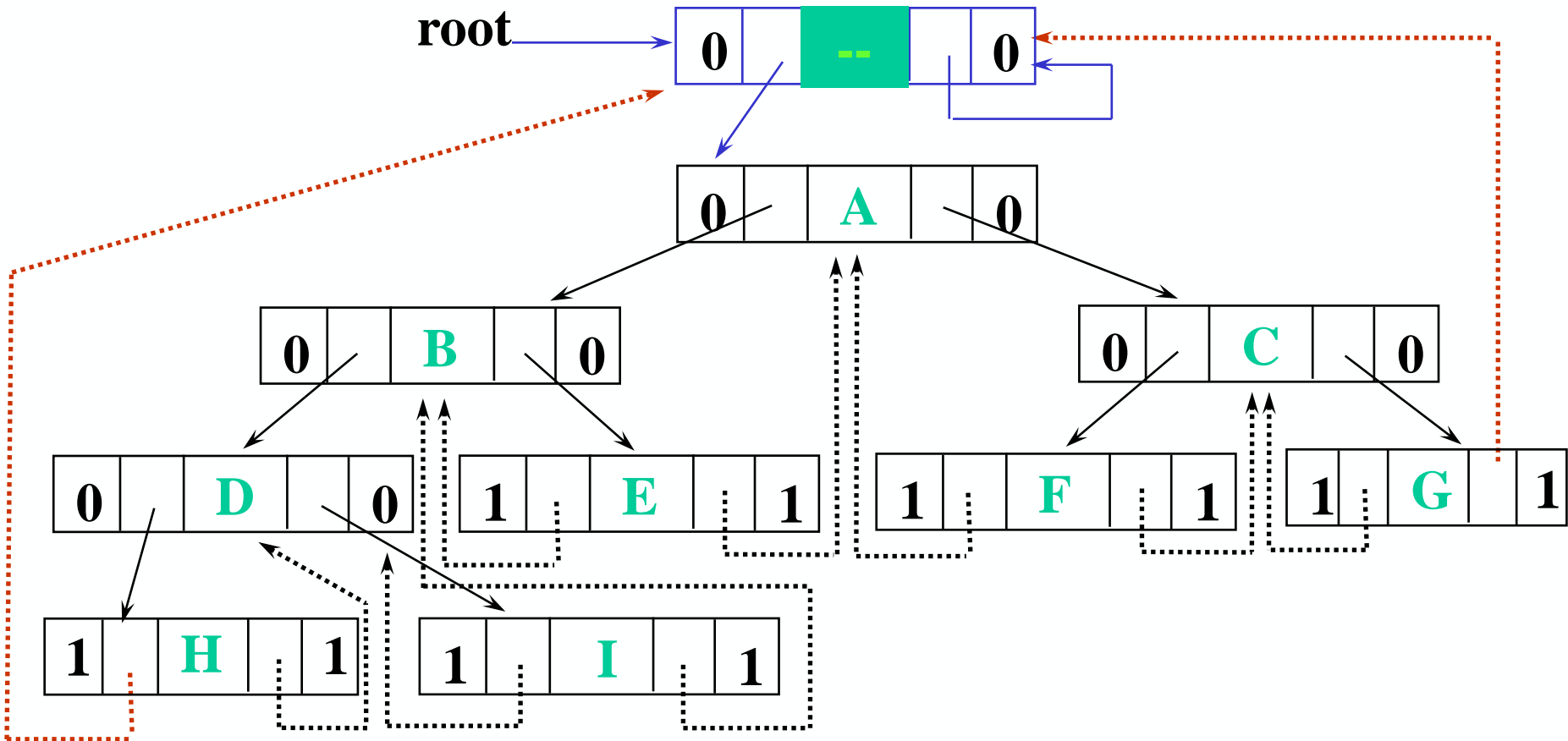
画出以下二叉树对应的中序线索二叉树。

该二叉树中序遍历结果为: **H**, **D**, **I**, **B**, **E**, **A**, **F**, **C**, **G**



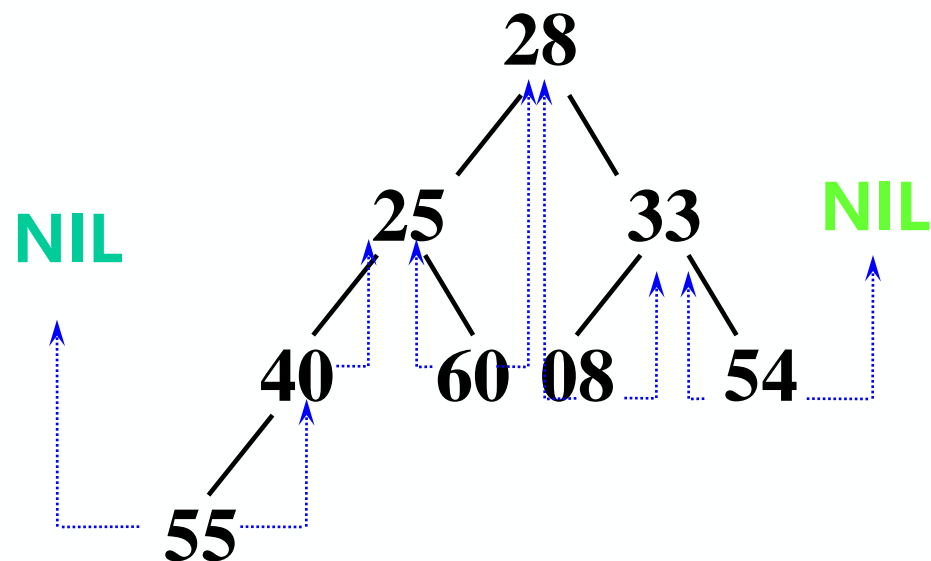
对应的中序线索二叉树存储结构如图所示：

注：此图中序遍历结果为：H, D, I, B, E, A, F, C, G



练习

画出与二叉树对应的中序线索二叉树



因为中序遍历序列是：**55** 40 25 60 **28** 08 33 **54**
对应线索树应当按此规律连线，即在原二叉树中添
加虚线。

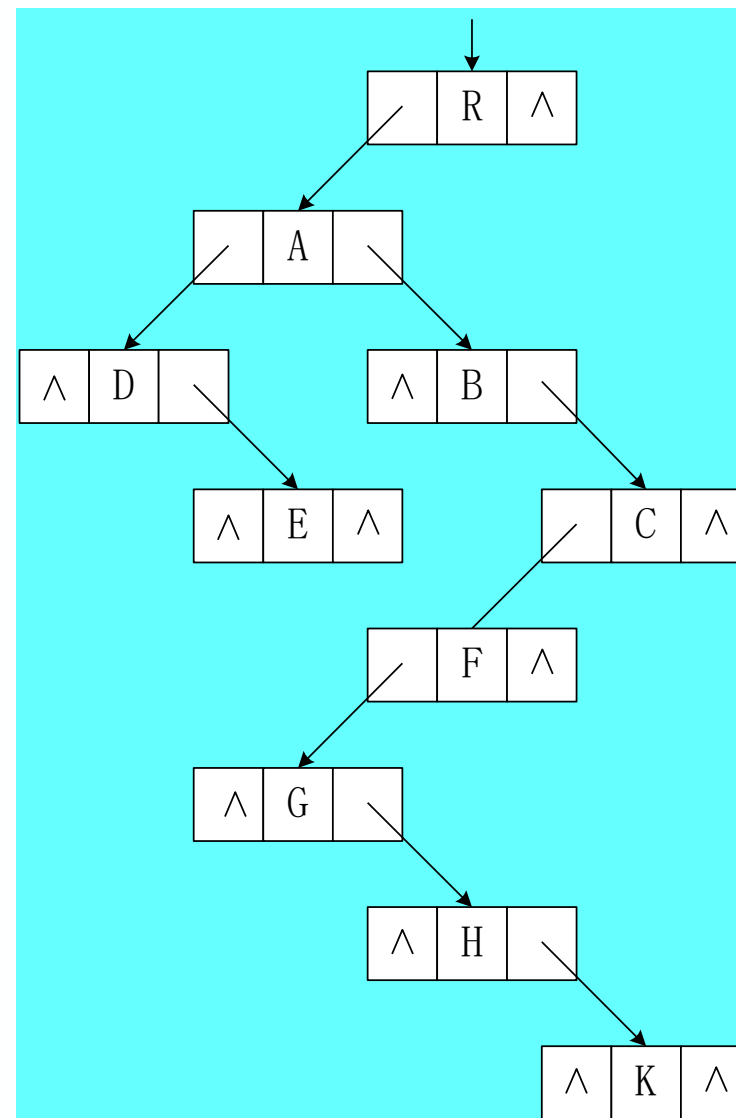
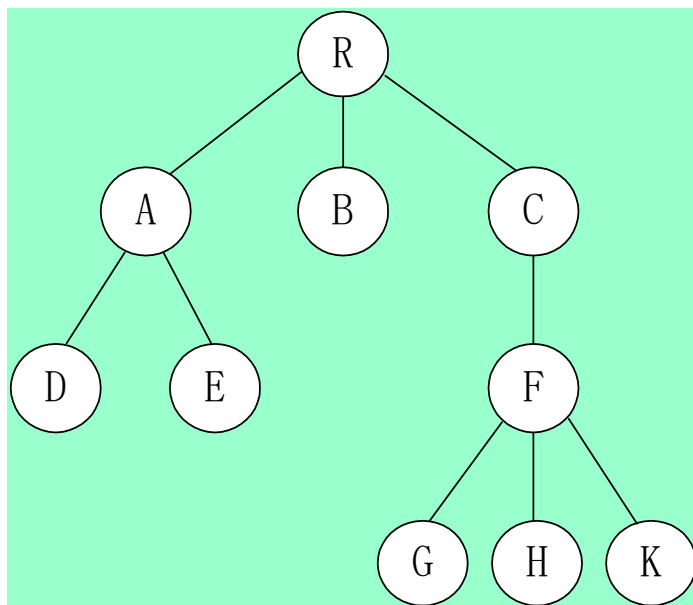
5.6 树和森林



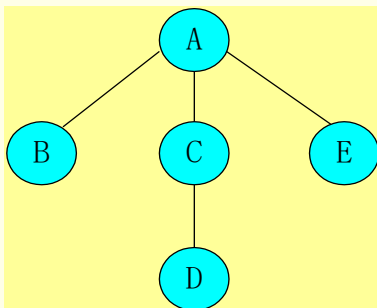
树的存储结构 - - 二叉链表表示法

```
typedef struct CSNode{  
    ElemType      data;  
    struct CSNode  *firstchild,*nextsibling;  
}CSNode,*CSTree;
```

树的存储结构 - - 二叉链表表示法



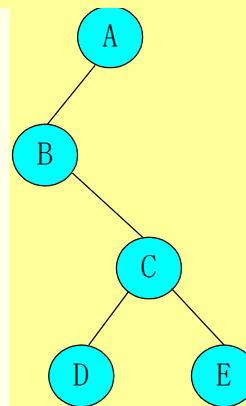
树



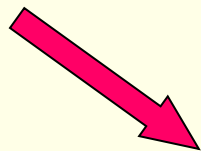
对应



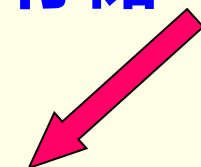
二叉树



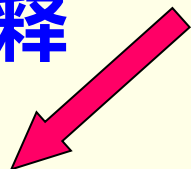
存储



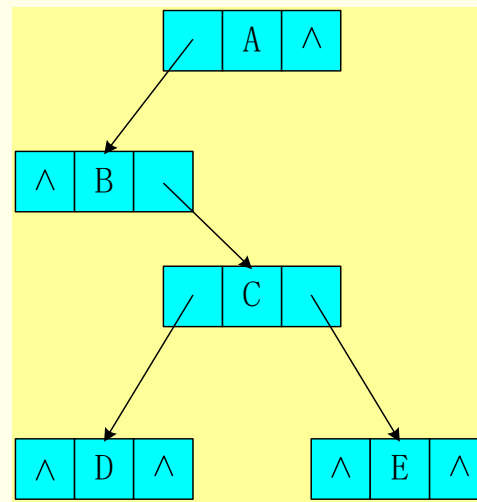
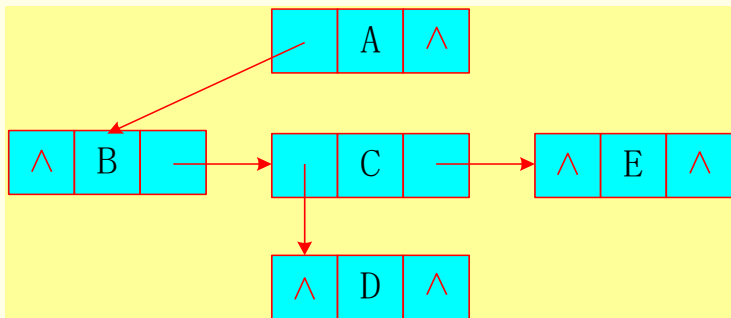
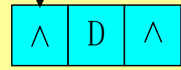
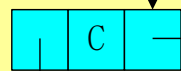
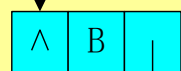
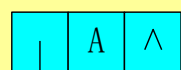
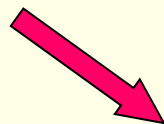
存储



解释




解释



5.7 哈夫曼树及其应用



- **游戏**中主角的生命值 d ，有这样的条件判定：当怪物碰到主角后，**怪物的反应**遵从下规则：



条件：	$d < 100$	$100 \leq d < 200$	$200 \leq d < 300$	$300 \leq d < 500$	$d > 500$
反应：	嘲笑，单挑	单挑	嗜血魔法	呼唤同伴	逃跑



条件:	$d < 100$	$100 \leq d < 200$	$200 \leq d < 300$	$300 \leq d < 500$	$d > 500$
反应:	嘲笑, 单挑	单挑	嗜血魔法	呼唤同伴	逃跑



```
if(d<100) state=嘲笑, 单挑;  
    else if(d<200) state=单挑;  
        else if(d<300) state=嗜血魔法;  
            else if(d<500) state=呼唤同伴;  
                else state=逃跑;
```

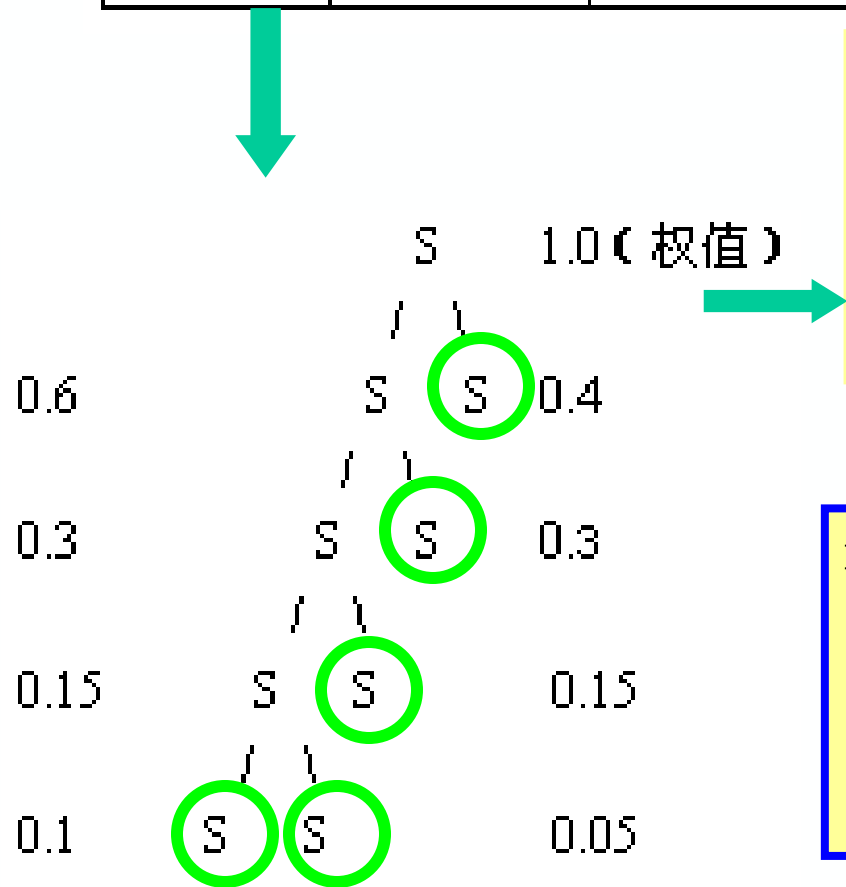


- 分析主角生命值 d 的特点，即预测出每种条件占总条件的**百分比**，将这些比值作为权值来构造最优二叉树（**哈夫曼树**），作为判定树来设定算法。

提高效率

条件：	$d \leq 100$	$100 \leq d \leq 200$	$200 \leq d \leq 300$	$300 \leq d \leq 500$	$d > 500$
比例：	5%	15%	40%	30%	10%

条件:	$d < 100$	$100 \leq d < 200$	$200 \leq d < 300$	$300 \leq d < 500$	$d \geq 500$
比例:	5%	15%	40%	30%	10%



if($d \geq 200$) && ($d < 300$) state=嗜血魔法;
 else if($d \geq 300$) && ($d < 500$) state=呼唤同伴;
 else if($d \geq 100$) && ($d < 200$) state=单挑;
 else if($d < 100$) state=嘲笑, 单挑;
 else state=逃跑;

if($d < 100$) state=嘲笑, 单挑;
 else if($d < 200$) state=单挑;
 else if($d < 300$) state=嗜血魔法;
 else if($d < 500$) state=呼唤同伴;
 else state=逃跑;

哈夫曼树应用实例 - - 哈夫曼编码

在远程通讯中，要将待传字符转换成二进制的字符串，怎样编码才能使它们组成的报文在网络中传得最快？

A	00
B	01
C	10
D	11

000110010101100

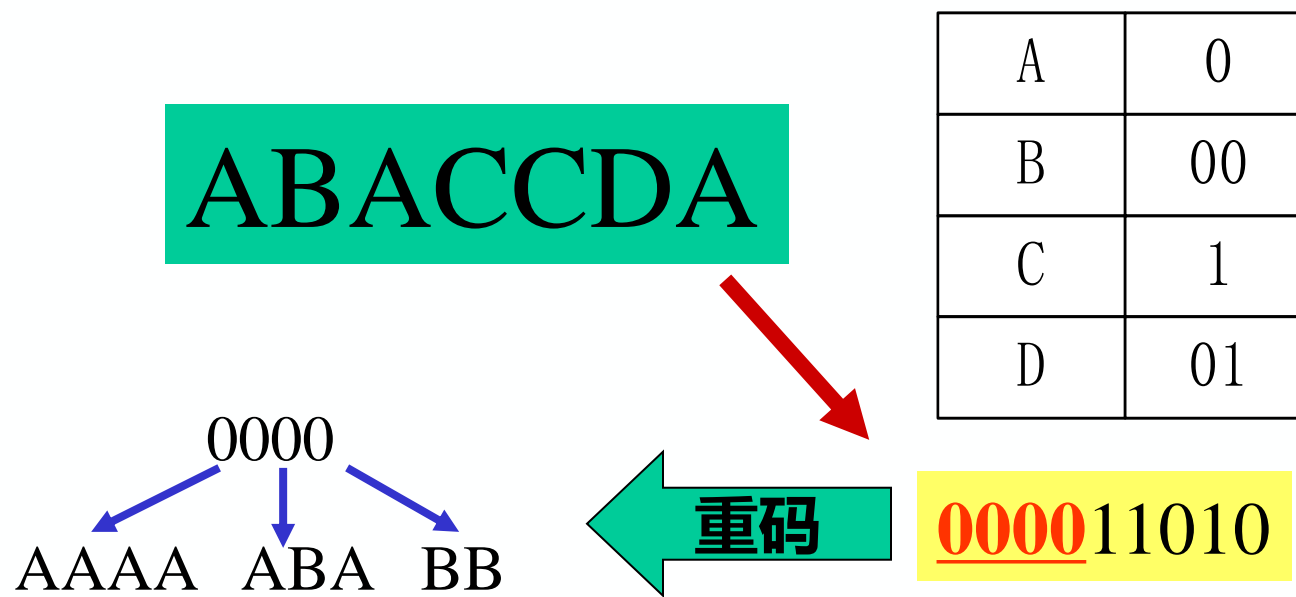
ABACCD A

A	0
B	00
C	1
D	01

000011010

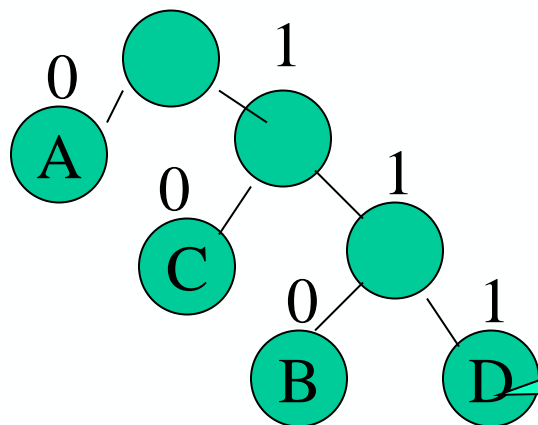
出现次数较多的字符采用尽可能短的编码

哈夫曼树应用实例 - - 哈夫曼编码



关键：要设计长度不等的编码，则必须使任一字符的编码都不是另一个字符的编码的**前缀** - **前缀编码**

采用二叉树设计
前缀编码



ABACCD A

A—0

B—110

C—10

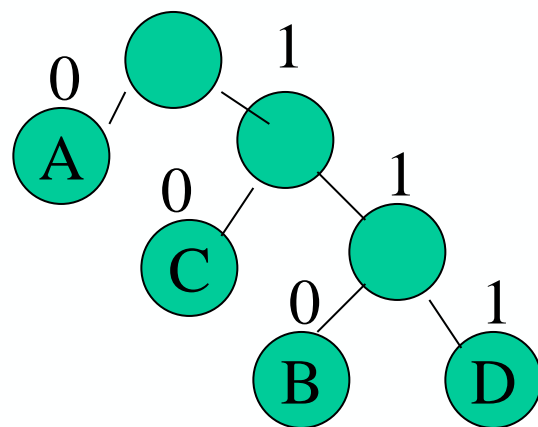
D—111

0110010101110

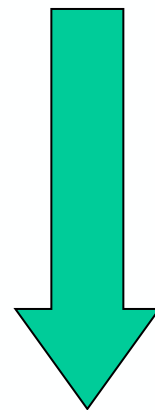
左分支用“0”
右分支用“1”

哈夫曼编码的译码过程

分解接收字符串：遇“0”向左，遇“1”向右；一旦到达叶子结点，则译出一个字符，反复由根出发，直到译码完成。



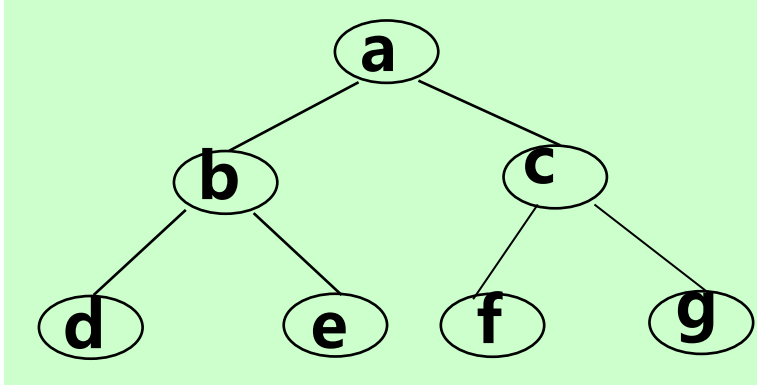
0110010101110



ABACCD A

特点：每一码都不是另一码的前缀，绝不会错译！称为前缀码

哈夫曼树的构造



路 径： 由一结点到另一结点间的分支所构成

路径长度： 路径上的分支数目 $a \rightarrow e$ 的路径长度 = 2

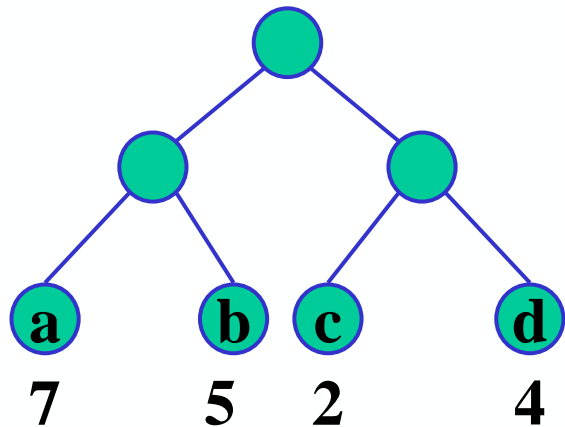
带权路径长度： 结点到根的路径长度与结点上权的乘积

树的带权路径长度： 树中所有叶子结点的带权路径长度之和

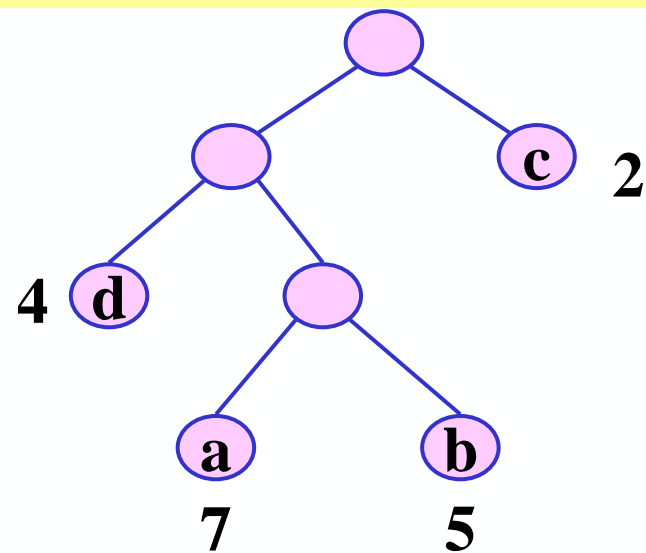
$$WPL = \sum_{k=1}^n w_k l_k$$

哈 夫 曼 树： 带权路径长度最小的树

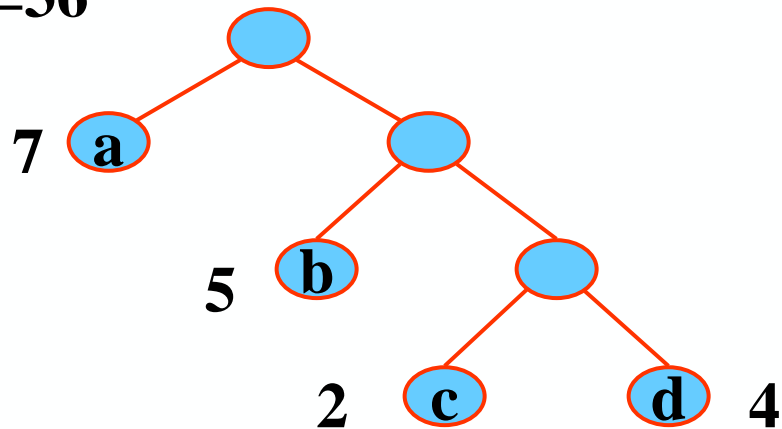
权值分别为7, 5, 2, 4, 构造有4个叶子结点的二叉树



$$WPL=7*2+5*2+2*2+4*2=36$$



$$WPL=7*3+5*3+2*1+4*2=46$$

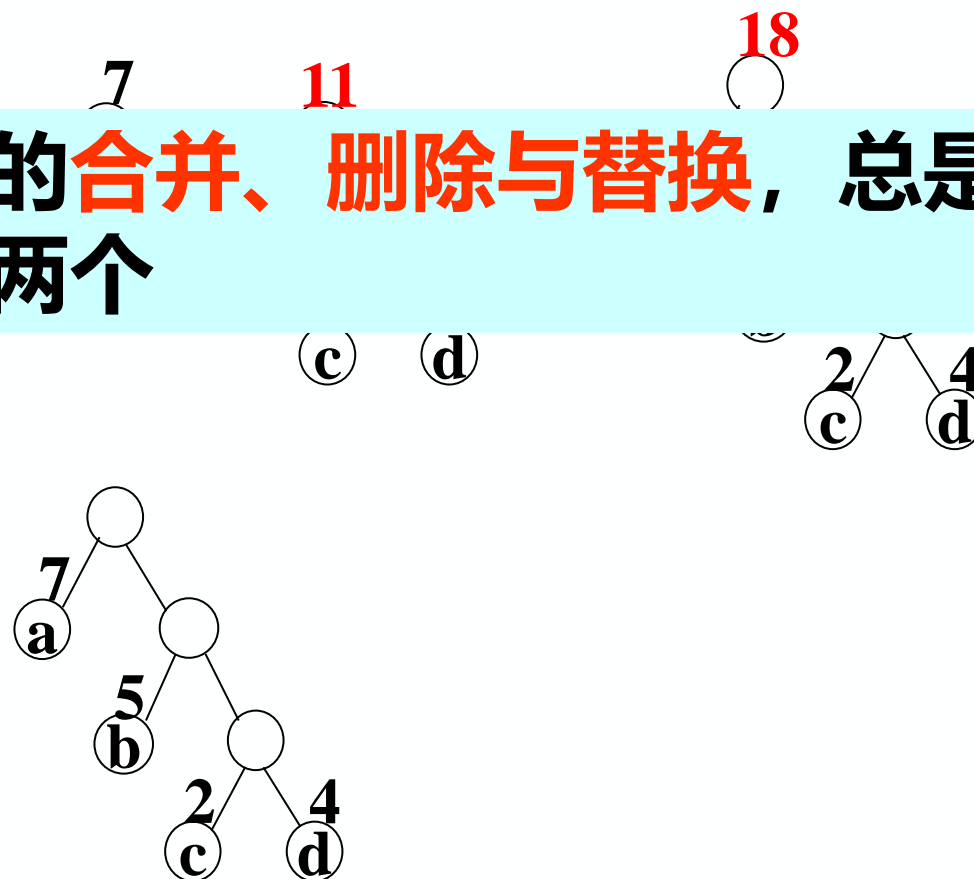


$$WPL=7*1+5*2+2*3+4*3=35$$

哈夫曼树的构造过程

基本思想：使权大的结点靠近根

操作要点：对权值的**合并、删除与替换**，总是合并当前值最小的两个

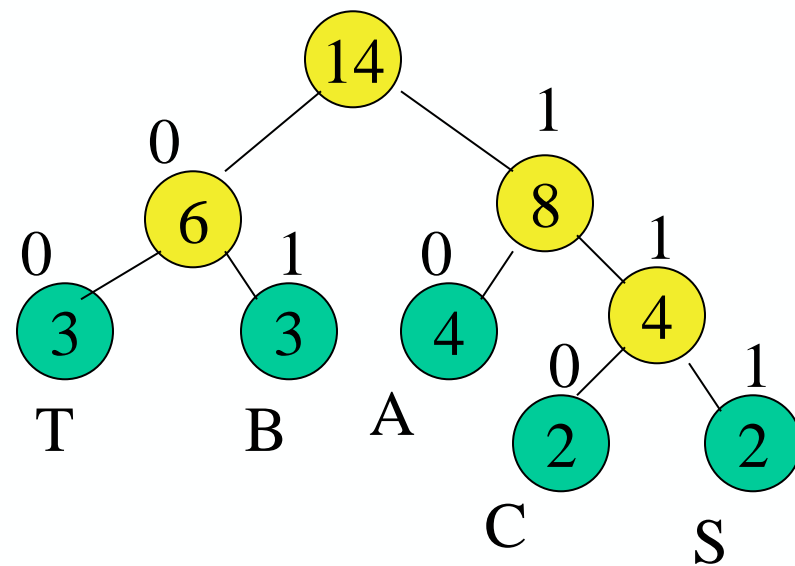


哈夫曼编码的构造

基本思想：概率大的字符用短码，小的用长码，构造哈夫曼树

例：某系统在通讯时，只出现C, A, S, T, B五种字符，其出现频率依次为2, 4, 2, 3, 3，试设计Huffman编码。

T	00
B	01
A	10
C	110
S	111



→ 例5.2

哈夫曼树的构造过程

- ✓ 根据给定的 n 个权值 $\{w_1, w_2, \dots, w_n\}$, 构造 **n 棵只有根结点的二叉树**。
- ✓ 在森林中选取两棵根结点**权值最小的树作左右子树**, 构造一棵新的二叉树, 置新二叉树根结点权值为其左右子树根结点权值之和。
- ✓ 在森林中**删除这两棵树**, 同时将新得到的二叉树加入森林中。
- ✓ 重复上述两步, **直到只含一棵树为止**, 这棵树即哈夫曼树。

哈夫曼树构造算法的实现（算法5.10）

一棵有n个叶子结点的Huffman树 **$2n-1$** 个结点

- ✓ 采用顺序存储结构——一维结构数组
- ✓ 结点类型定义

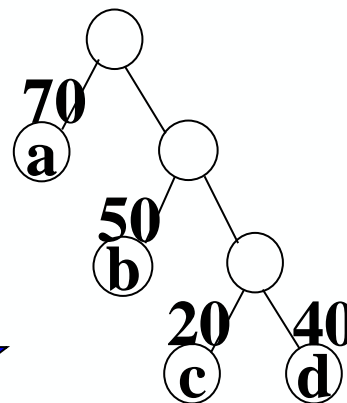
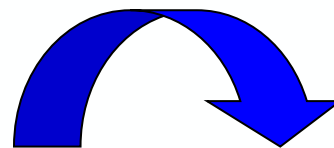
```
typedef struct  
{ int weght;  
  int parent,lch,rch;  
}*HuffmanTree;
```

哈夫曼树构造算法的实现

- 1) 初始化 $HT[1..2n-1]$: $lch=rch=parent=0$
- 2) 输入初始 n 个叶子结点: 置 $HT[1..n]$ 的 $weight$ 值
- 3) 进行以下 $n-1$ 次合并, 依次产生 $HT[i]$, $i=n+1..2n-1$:
 - 3.1) 在 $HT[1..i-1]$ 中选两个未被选过的 $weight$ 最小的两个结点 $HT[s1]$ 和 $HT[s2]$ (从 $parent = 0$ 的结点中选)
 - 3.2) 修改 $HT[s1]$ 和 $HT[s2]$ 的 $parent$ 值: $parent=i$
 - 3.3) 置 $HT[i]$: $weight=HT[s1].weight + HT[s2].weight$,
 $lch=s1, rch=s2$

例: 设 $n=4$, $w=\{70, 50, 20, 40\}$
 试设计 huffman code ($m=2*4-1=7$)

	weight	parent	lch	rch
1	70	0	0	0
2	50	0	0	0
3	20	0	0	0
4	40	0	0	0
5				
6				
7				



	weight	parent	lch	rch
1	70	7	0	0
2	50	6	0	0
3	20	5	0	0
4	40	5	0	0
5	60	6	3	4
6	110	7	2	5
7	180	0	1	6