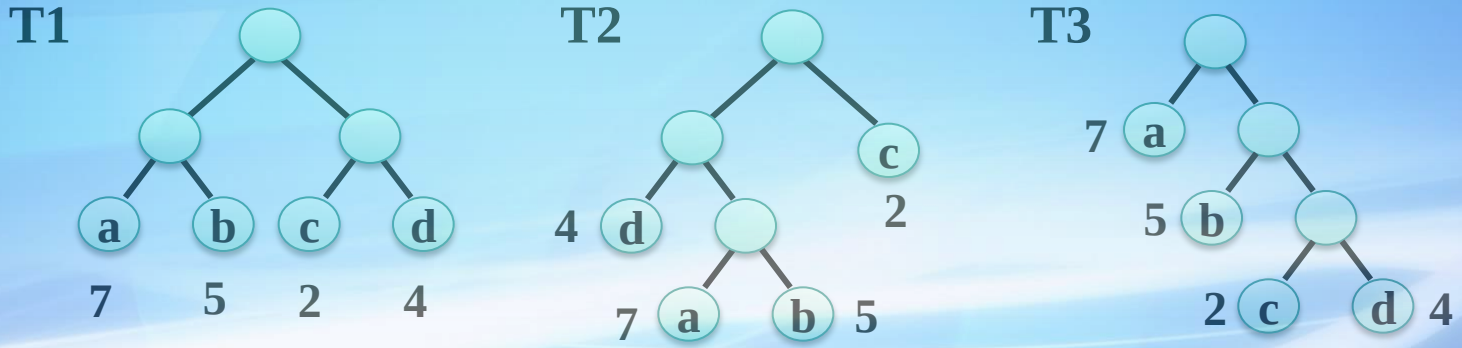
哈夫曼树

《数据结构》

主讲人：李清



什么是哈夫曼树？

编码

编码：是指用不同的0、1序列代表不同的信息。

- 西文信息（ASCII码）
- 汉字信息（GB2312编码、GBK、BIG5）
- 图像、图形、声音等信息

总的原则

- 要能唯一地译码
- 编码长度要尽量短

编码种类

等长编码

不等长编码

树型编码　哈夫曼编码

哈夫曼

戴维·哈夫曼（David A. Huffman 1925—1999）

美国计算机科学家，分别于1944年和1949年从俄亥俄洲立大学获得学士和硕士学位，1953年在麻省理工学院（MIT）获得博士学位。1962—1967年在MIT任教授，1967年到加洲大学圣克鲁斯分校创办计算机系。他将他全部的精力放在教学上，以他自己的话来说，“我所要带来的就是我的学生。”

1982年获得IEEE计算机先驱奖，所提出的哈夫曼编码方法被广泛应用于数据的压缩和传输。

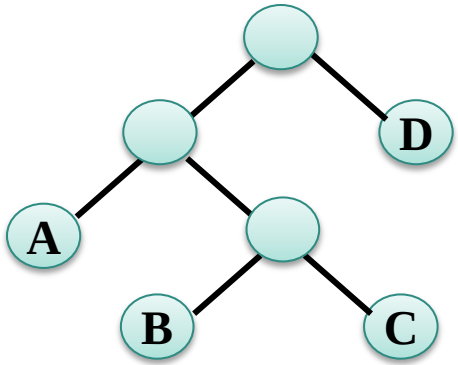
戴维·哈夫曼

教学目标和要求

1. 能够准确阐述哈夫曼树、哈夫曼编码的定义
2. 能够描述哈夫曼树的自底向上子树合并构造算法思想
3. 对于给定的编码频度，能够正确给出其哈夫曼编码
4. 能够举例说明哈夫曼编码应用
5. 能够编程实现哈夫曼树构造算法

叶的路径长度

从根结点到叶结点的路径上的边数。



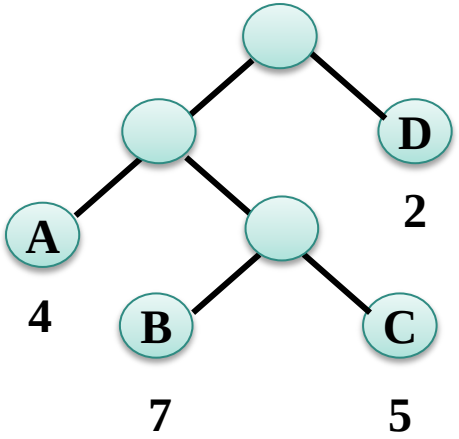
A的路径长度：2

B、C的路径长度：3

D的路径长度：1

叶的加权路径长度

设二叉树的叶子，其权值为正实数w，路径长度为*l*，那么，W***l* 称为该叶的加权路径长度。



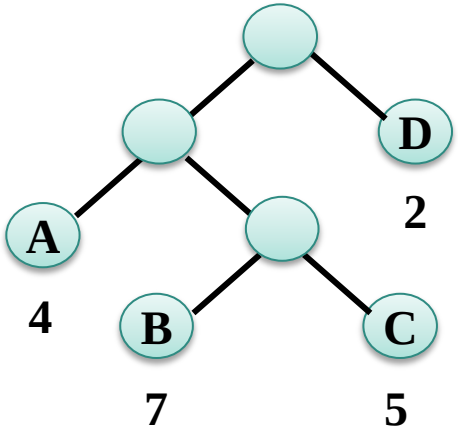
A的加权路径长度：4*2

B的加权路径长度：7*3

C的加权路径长度：5*3

D的加权路径长度：2*1

二叉树的权



设二叉树T具有n片叶子，n个正实数w_1至w_n作为各叶的权，根到第i（1≤i≤n）片叶子的路径长度（边数）为l_i，那么，**二叉树T的权**为所有叶的加权路径长度之和。即：

$$W(T)=\sum_{k=1}^{n} w_k l_k$$

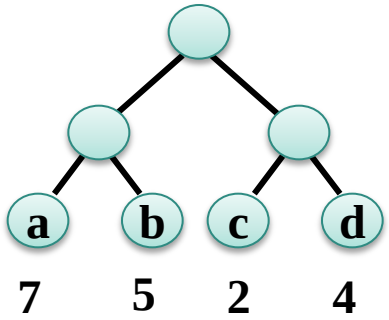
W(T)=4*2+7*3+5*3+2*1

=46

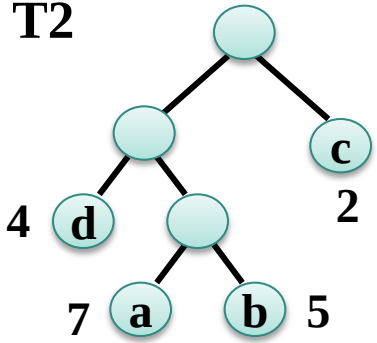
实例

已知叶结点a,b,c,d的权值分别为7，5，2，4。求下列三棵二叉树的权。

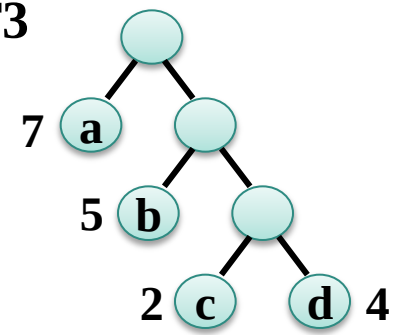
T1



T2



T3



W(T1)=7×2+5×2+2×2+4×2=36

W(T2)=4×2+7×3+5×3+2×1=46

W(T3)=7×1+5×2+2×3+4×3=35

哪种树的权最小？

哈夫曼树

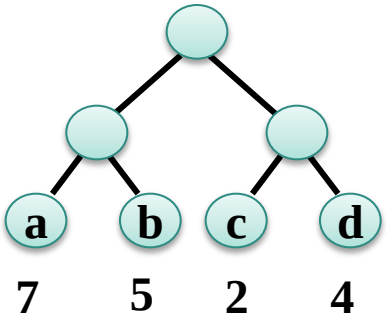
给定n个正实数w_1至w_n，分别为n片叶子的权，l_1至l_n为对应叶子的 路径长度，使权

$$W(T)=\sum_{k=1}^{n} w_k l_k$$

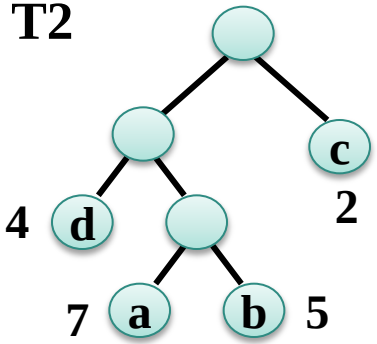
达最小值的树，称为**哈夫曼树**。

哈夫曼树

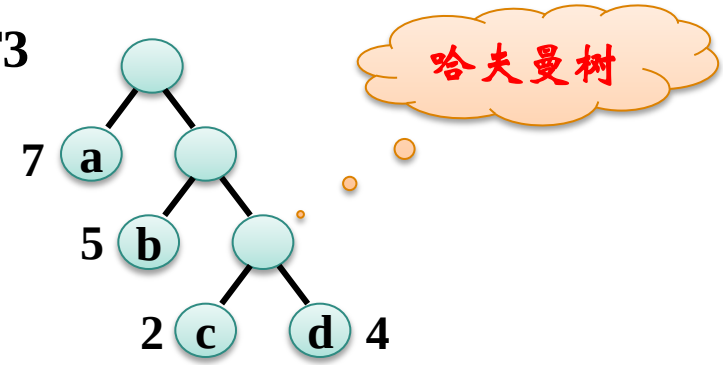
T1



T2



T3



哈夫曼树必是正则二叉树！

如何判定二叉树是否为哈夫曼树呢？

哈夫曼树的定义

哈夫曼树的构造

如何构造哈夫曼树呢？

哈夫曼算法----哈夫曼树的构造

“自底向上”逐步合并的方法

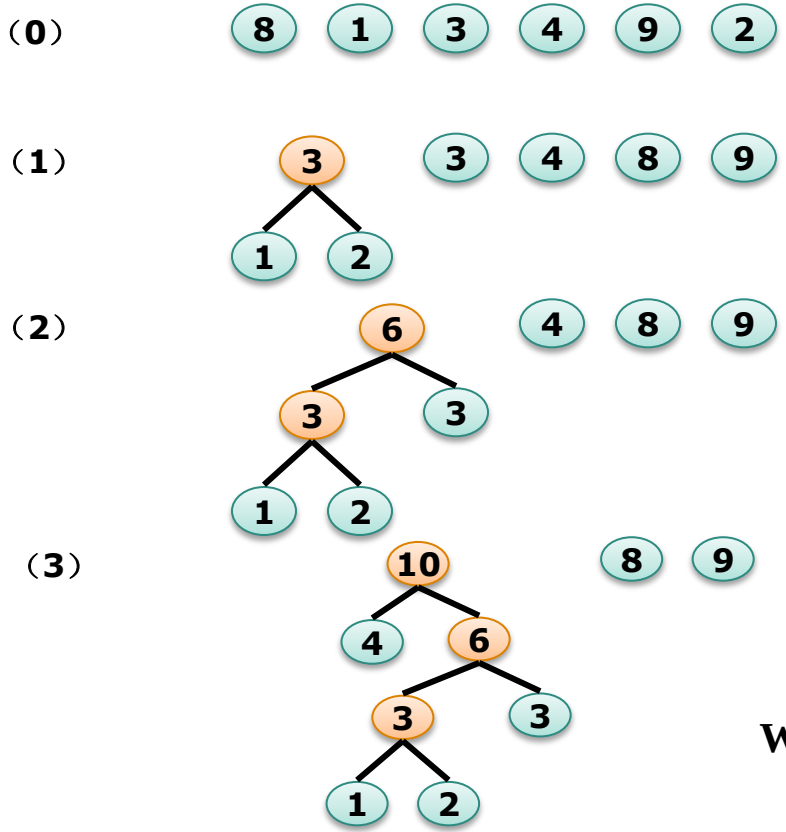
1、先构造n棵单叶的二叉树（n棵树组成森林），依次将权w_1，w_2，…w_n 赋给各个叶子；

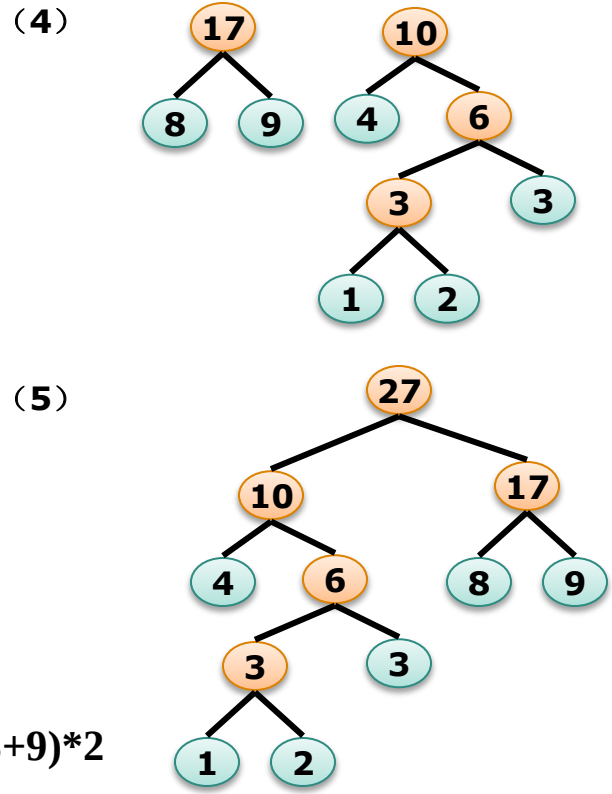
2、反复合并（n-1遍），直到森林中只有一棵树：

- 找出根的权最小的两棵树：T1和T2；
- 将T1，T2合并成一棵树T，使T1，T2分别作为T的左右子树，且使T之根的权等于T1，T2之根的权之和；
- 把合并树T加入森林中；

一叶一树，最小合并

实例：有6个数8，1，3，4，9，2，构造哈夫曼树。

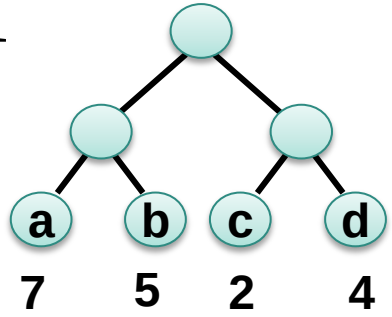




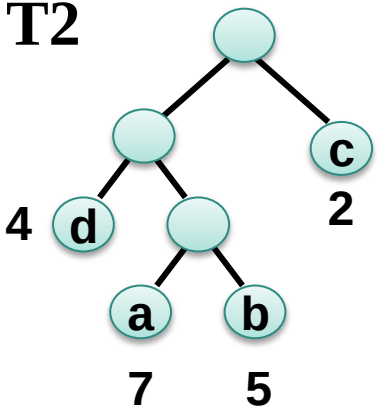
W(T)=(1+2)*4+3*3+(4+8+9)*2
=63

实例：判断下列三棵树是否为哈夫曼树，为什么？

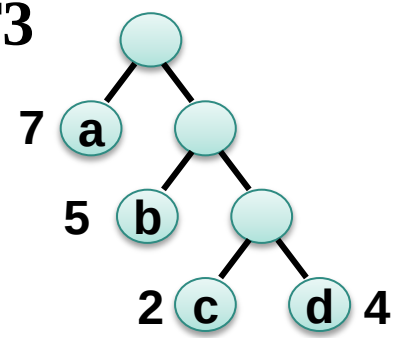
T1



T2



T3



T3为哈夫曼树

如何编程实现哈夫曼算法

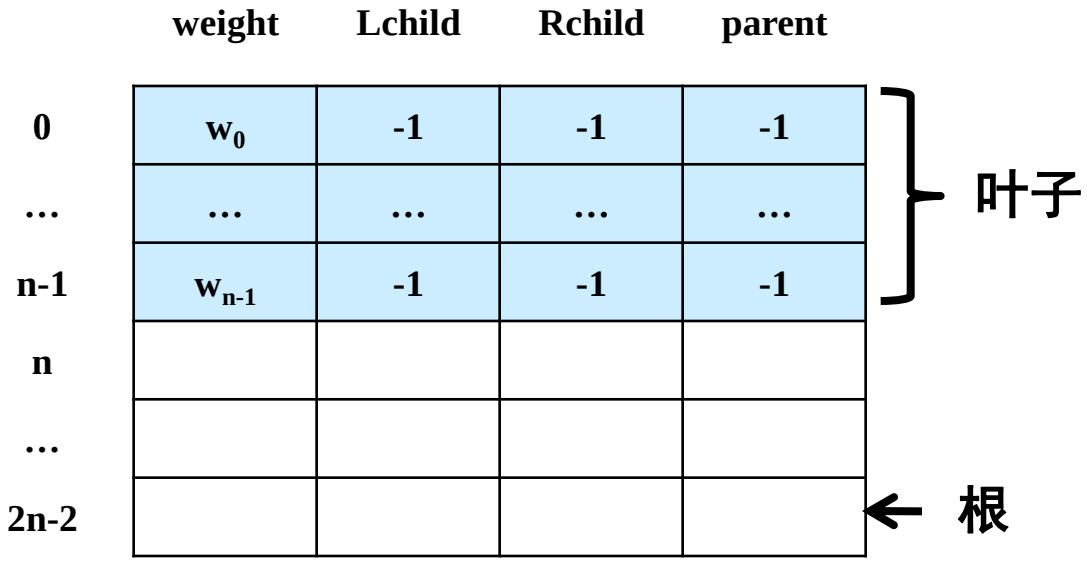
构造哈夫曼树？

哈夫曼算法----哈夫曼树的构造

数据结构描述

```
typedef struct Bhnode
{  int weight;
   int parent;
   int Lchild,Rchild;
}htnode;

htnode hf[2*n-1];
   //n片叶子的哈夫曼树有2n-1个结点
  //静态链表
```



哈夫曼算法----哈夫曼树的构造

```
void HUFM(int n,int w[n]） // n片叶，w[n]记录n个权值
{
 1. for(j=0;j<2*n-1;j++)//初始化
 2. { hf[j].parent=-1;
 3.   hf[j].Lchild=-1;
 4.   hf[j].Rchild=-1;
 5.  }
 6. for(j=0;j<n;j++) hf[j].weight=w[j];//赋权值
 7. for(k=n;k<2*n-1;k++)       //产生新树的根结点存放于hf[k]中
 8. {
 9.     select(k-1,i,j); //在前0..K-1个结点中选择两个双亲域为-1，而权值最小的结点i和j
10.     hf[i].parent=k; hf[j].parent=k;
11.      hf[k].Lchild=i; hf[k].Rchild=j;
12.      hf[k].weight=hf[i].weight+hf[j].weight;
13.  }
}
```

何为哈夫曼树编码？

哈夫曼编码

定义

字符作为叶子，字符的使用频度，作为权值构成的哈夫曼树，若左分枝以“0”，右分枝以“1”，则从根到叶子的路径分枝上的01字符串构成该叶子结点对应字符的编码。

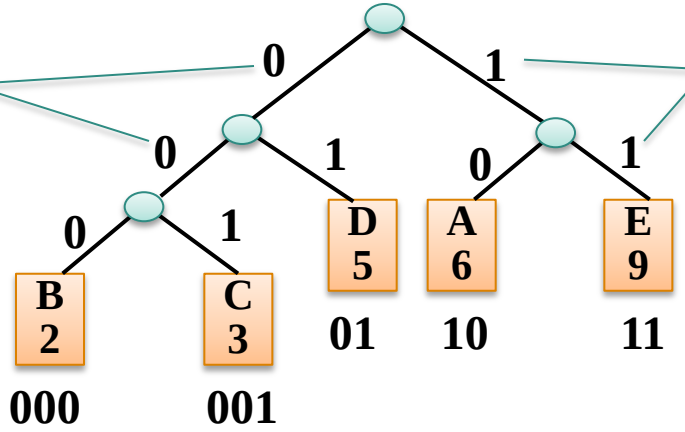
这种方法得到的字符编码称为**哈夫曼编码**。

保证能唯一译码，且编码方案最佳

哈夫曼编码----实例

向左的边标记为0

向右的边标记为1



对字符串　B A E D

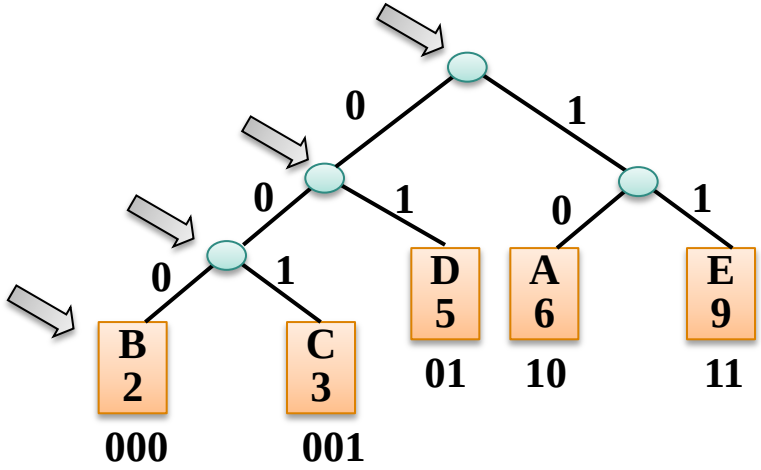
编码　　000 10 11 01

字符	哈夫曼编码
A	10
B	000
C	001
D	01
E	11

哈夫曼编码----译码实例

编码序列：0 0 0 1 1 1 0 0 1

编码、译码对应同一棵哈夫曼树

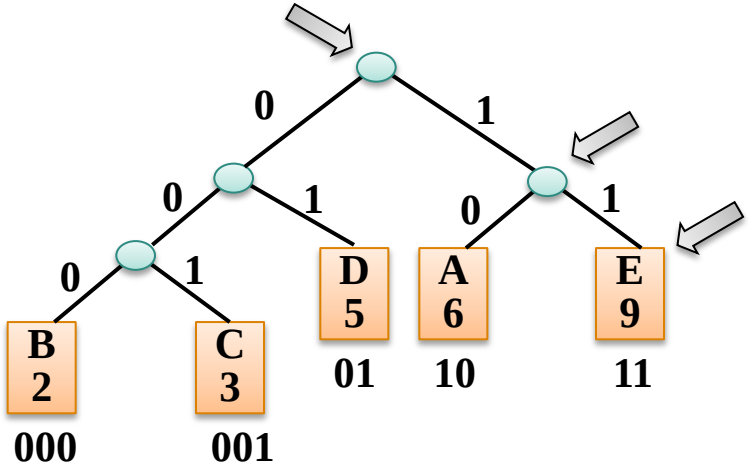


译码结果：B

哈夫曼编码----译码实例

编码序列：0 0 0 1 1 1 0 0 1

编码、译码对应同一棵哈夫曼树

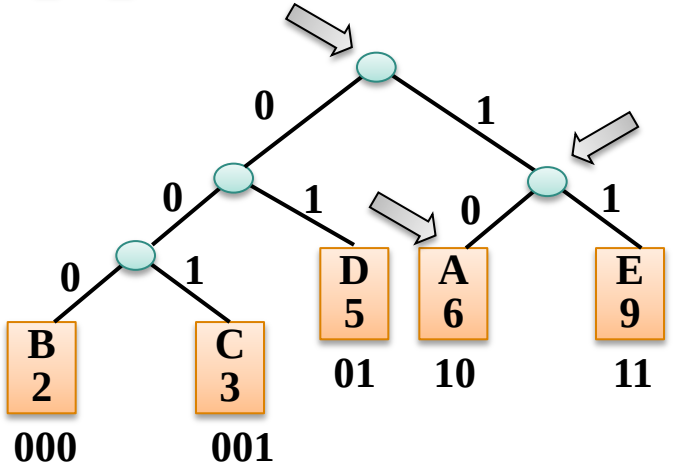


译码结果： B E

哈夫曼编码----译码实例

编码序列：0 0 0 1 1 1 0 0 1

编码、译码对应同一棵哈夫曼树

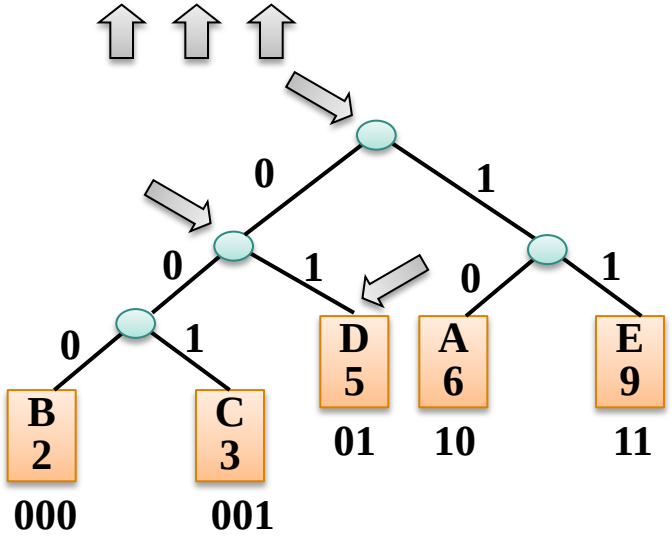


译码结果：B E A

哈夫曼编码----译码实例

编码序列：0 0 0 1 1 1 0 0 1

编码、译码对应同一棵哈夫曼树



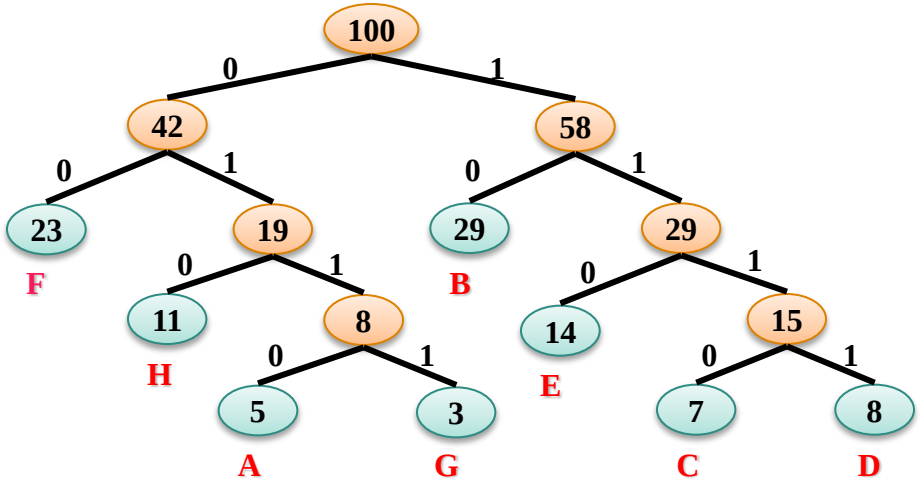
译码结果：B E A D

译码结束，唯一译码

实例

某通信系统中，通信联络只可能出现八种字符ABCDEFGH，各字符出现频度分别为： 0.05，0.29，0.07，0.08，0.14，0.23，0.03，0.11 。假设通信传输一个100,000个字符的文件。为了提高传输效率，试设计一编码方案，使得编码后的数据量最小。

实例：ABCDEFGH，频率为：0.05，0.29，0.07，0.08，0.14，0.23，0.03，0.11



字符	频率（权值）	编码
A	5	0110
B	29	10
C	7	1110
D	8	1111
E	14	110
F	23	00
G	3	0111
H	11	010

哈夫曼编码：100000*（（23+29）*2+（11+14）*3+（5+3+7+8）*4）/100=271000

ASCII码：100000*8=800000

实验数据

- 用《平凡的世界》做压缩测试，压缩前为文本文件，大小为1.7M，压缩后为二进制文件，大小接近1M(988,817byte)。

哈夫曼编码是无损压缩技术

- 另外，因为从Huffman压缩算法的原理可知，该算法对字符重复率高的文本最有效，比如长篇小说或者英文小说。

小结

- 哈夫曼编码是以哈夫曼树为编码树
- 哈夫曼编码是最短编码
- 哈夫曼编码译码唯一
- 通过哈夫曼编码可进行编码压缩