



# 数据结构

## Data Structure

张先宜

手机: 18056307221 13909696718

邮箱: [zxianyi@163.com](mailto:zxianyi@163.com)

QQ: 702190939

QQ群: XC数据结构交流群 **275437164**

# 第5章 树 (Tree)

- 5.1 树
- 5.2 二叉树
- 5.3 二叉树的遍历
- 5.4 线索二叉树
- 5.5 树和森林
- **5.6 哈夫曼树 ( Huffman Tree )**

## 5.6 哈夫曼树 (Huffman Tree)

- 哈夫曼树 – Huffman Tree
- 又叫最优二叉树
- Huffman 1952 年提出。

# 5.6.1 哈夫曼树基本概念

## 1. 引言

☞ 视频、音频、文本、数据等，任何形式的内容在计算机内部以何种形式存储？

✦ **编码：0、1编码**

☞ 听说过文件压缩吗？你知道那些压缩方式？

✦ **zip、JPG、MPG、Huffman... , 事实是压缩编码。**



👉听说过“密电码”吗？如何使用？

✦编码、译码。

- 接下来要介绍的Huffman树是一种压缩编码技术。

## 2. 几个概念

### ☞ (1) 路径

✦ 从一个结点到另一个结点所经过的结点和边。

### ☞ (2) 路径长度

✦ 路径上的边数。

### ☞ (3) 结点的带权路径长度

✦ 结点到树根的路径长度和结点权值的乘积。

#### ☞ (4) 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。
- ✦ 设有  $n$  个叶子结点， $w_i$  为第  $i$  个叶子结点的权值， $L_i$  为第  $i$  个叶子结点的路径长度，树的带权路径长度记作：

$$WPL = \sum_{i=1}^n w_i L_i$$

### 3. 哈夫曼树的定义

👉 给定一组数值  $\{w_1, w_2, \dots, w_n\}$  作为叶子结点的权值，构造一棵二叉树。


$$WPL = \sum_{i=1}^n w_i L_i$$

👉 若二叉树满足：**WPL 为最小**（其中  $L_i$  为  $w_i$  对应的叶子结点到根结点的路径长度），则称此二叉树为**最优二叉树**，也称**哈夫曼树**，并称**WPL**为树的带权路径长度。



## 5.6.2 哈夫曼树的构造

- 对给定的  $w=\{w_1, w_2, \dots, w_n\}$  , 以此作为叶子结点的权值, 如何构造哈夫曼树 ?
- 构造方法 :
  - ① 根据给定的  $n$  个权值  $w=\{w_1, w_2, \dots, w_n\}$  , 构成  $n$  棵二叉树的集合  $T=\{T_1, T_2, \dots, T_n\}$  , 其中每个  $T_i$  只有一个权值为  $w_i$  的根结点, 其左右子树均空。

- 
- ② 从 $T$ 中选两棵根结点权值最小的二叉树  
(不妨设为  $T_1$ 、 $T_2$ )，作为左右子树构成  
一棵新二叉树  $T_1'$ ，并置新二叉树  $T_1'$  的  
根的权值为其左右子树 (即  $T_1$ 、 $T_2$ ) 的根  
结点的权值之和。
- ③ 将新二叉树  $T_1'$  并入到  $T$  中，同时从  $T$  中  
删除  $T_1$ 、 $T_2$ 。
- ④ 重复①、②，直到  $T$  中只有一棵树为止。  
这棵树便是哈夫曼树。

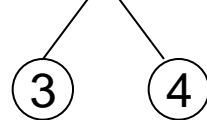
**【例】**以集合{3,4,5,6,8,10,12,18}为叶子结点的权值构造哈夫曼树，并计算其带权路径长度。

**【解】**

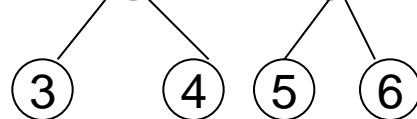
☞ 按构造算法，首先将这些数变成单结点的二叉树集合：

$T = \{ \textcircled{3}, \textcircled{4}, \textcircled{5}, \textcircled{6}, \textcircled{8}, \textcircled{10}, \textcircled{12}, \textcircled{18} \}$

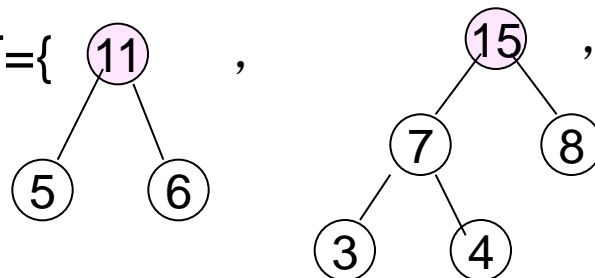
$T = \{ \textcircled{7}, \textcircled{5}, \textcircled{6}, \textcircled{8}, \textcircled{10}, \textcircled{12}, \textcircled{18} \}$

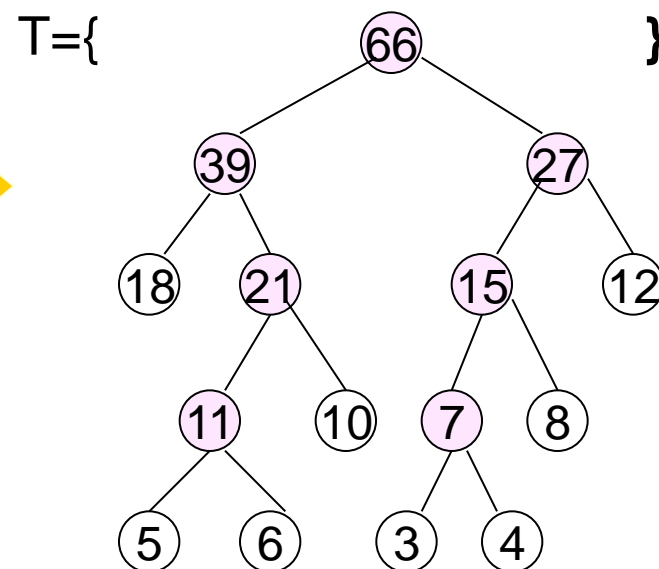
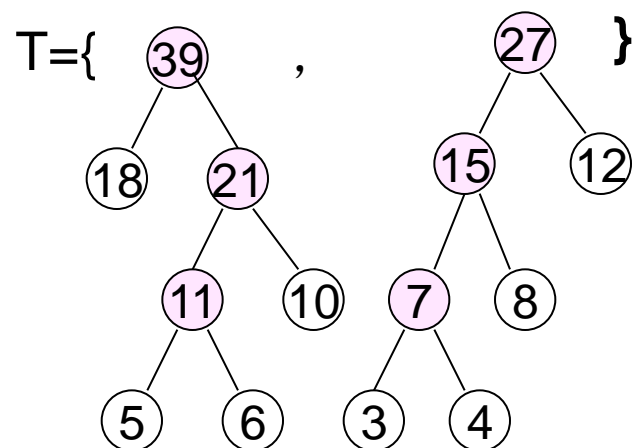
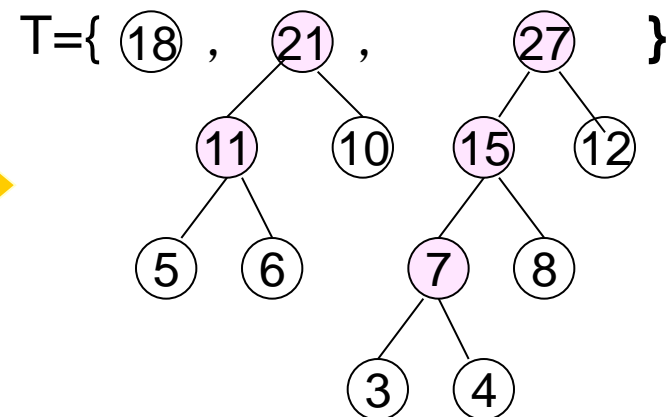
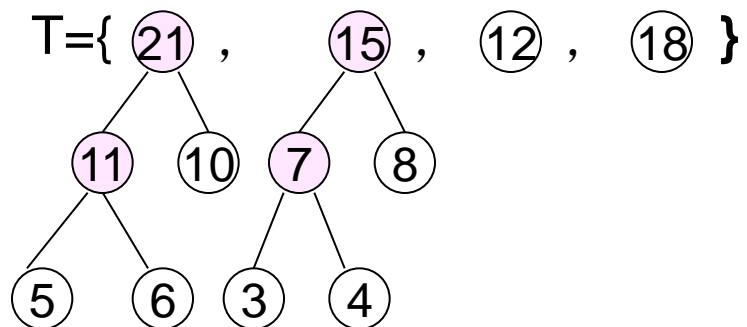


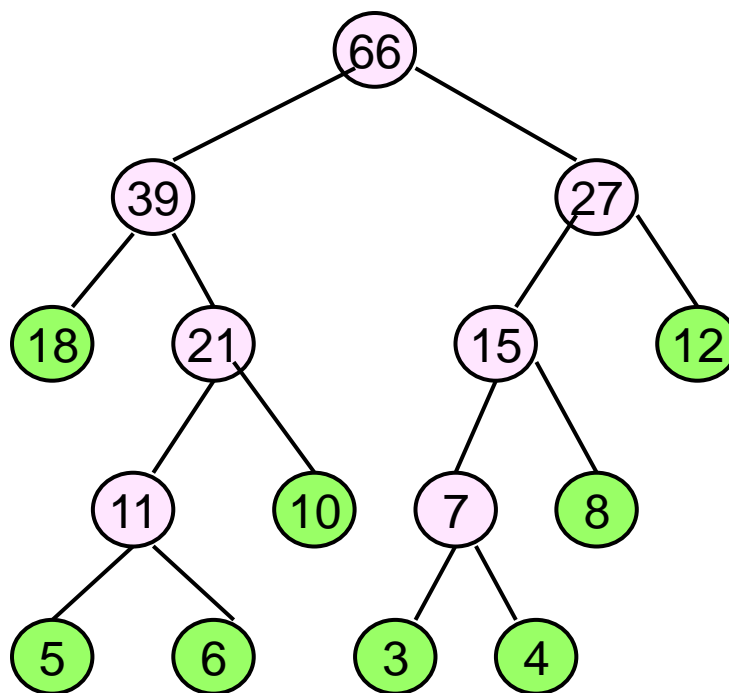
$T = \{ \textcircled{7}, \textcircled{11}, \textcircled{8}, \textcircled{10}, \textcircled{12}, \textcircled{18} \}$



$T = \{ \textcircled{11}, \textcircled{15}, \textcircled{10}, \textcircled{12}, \textcircled{18} \}$







### ■ 带权路径长度

$$\text{WPL} = (3 + 4 + 5 + 6) \times 4 + (8 + 10) \times 3 + (12 + 18) \times 2$$

$$= 186$$

$$= 66 + 39 + 27 + 21 + 15 + 11 + 7 \quad (\text{分支结点权值之和})$$

$$= 186$$

**Huffman树的WPL=分支结点权值之和。**

- ❏ 可以证明，任意一棵哈夫曼树的带权路径长度均具有这一性质。
- ❏ 根据这一性质求解**WPL**要快捷得多。

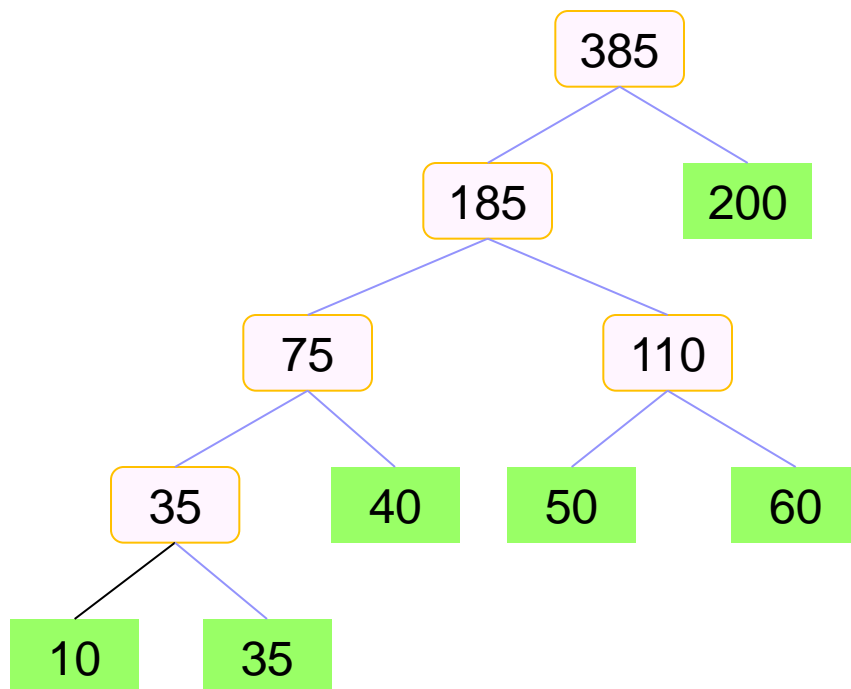
## ■ 哈夫曼树应用实例

(2012) (10分) 设有6个有序表A、B、C、D、E、F，分别有10、35、40、50、60和200个数据元素，各表中元素按升序排序。要求通过5次两两合并，将6个表最终合并成一个升序表，并在最坏情况下比较的总次数达到最小。请回答下列问题：

- (1) 请写出合并方案，并求出最坏情况下比较的总次数。
- (2) 根据你的合并过程，描述 $N$  ( $N \geq 2$ ) 个不等长升序表的合并策略，并说明理由。



- 总比较次数： $wpl=35+75+110+185+385=790$



## 5.6.3 哈夫曼编码

- 压缩编码（节省存储空间）
- 报文的编码、译码
  - ☞ 将报文的字符进行二进制编码、译码

## ■ 等长编码

☞ 对每个字符作长度相等的编码。

☞ 例：报文 “**ABAACCBADCA**”，有四个不同字符**A**、**B**、**C**、**D**，可用 **2 位(bit)**二进制编码，如：

✦ 编译码字典：**A-00、B-01、C-10、D-11**

✦ 密文：“**0001000010100100111000**”，总长为 **22 位**；

☞ 译码时，从左往右，每 **2位(bit)**为进行翻译，只要有编码字典，即可译出原文。

## ■ 不等长编码

☞ 对字符采用不等长的编码

✦ 比如上例中可以分别采用 1 位和 2 位编码，  
0、1、00、01；

☞ 为减少总码长度，容易想到，出现频率高的字符采用短的编码，频度低的字符采用长的编码，

✦ 如上例中 A、C 出现的频度高，进行下列编码：

✦ 字典：A--0、B--00、C--1、D--01

✦ 密文：“00000110000110”，

✦ 码长为 14 位，比等长编码短。

但是，不等长编码在译码时，存在歧义解释问题：

✦ 上例中，接收方在收到“00000”时，既可以译为“AAAAA”，也可译为“ABB”、或“BAB”、或“BAAA”等等。

怎么解决这个问题呢

– 前缀编码

## ■ 前缀编码

- ✎ 不等长编码中，任一个编码不是另一个编码的前缀。即：短编码不能是任何长编码的前缀（长编码切割不出段编码）。
- ✎ 前缀编码可以保证译码的唯一性。

## ■ 哈夫曼编码（一种前缀压缩编码）

- ☞ 设电文使用  $n$  个不同字符；
- ☞ 统计电文中各种字符（ $n$  种）出现的次数（频率）；
- ☞ 用这  $n$  个字符作为根结点，出现的次数（频率）作为权值，构成  $n$  棵二叉树（只有根结点）的森林；
- ☞ 用这  $n$  棵二叉树，构造一棵哈夫曼树；

☞ 哈夫曼树中，所有左分支对应的边（结点  
到其左孩子的边）上标记‘0’，右分支对应  
的边（结点到其右孩子的边）上标记‘1’；  
（反之亦可）

☞ 从根结点到每个叶子结点，经过边的 0、1  
组合起来即为此叶结点（字符）的编码。

■ 报文前缀码长度 = 哈夫曼树的带权路径长度 WPL

☞ 路径长度（边数）= 编码位数



## 【例】报文“ABAACCBADCA”

👉 A、B、C、D 的出现频率分别为：5、2、3、1

A	B	C	D
5	2	3	1

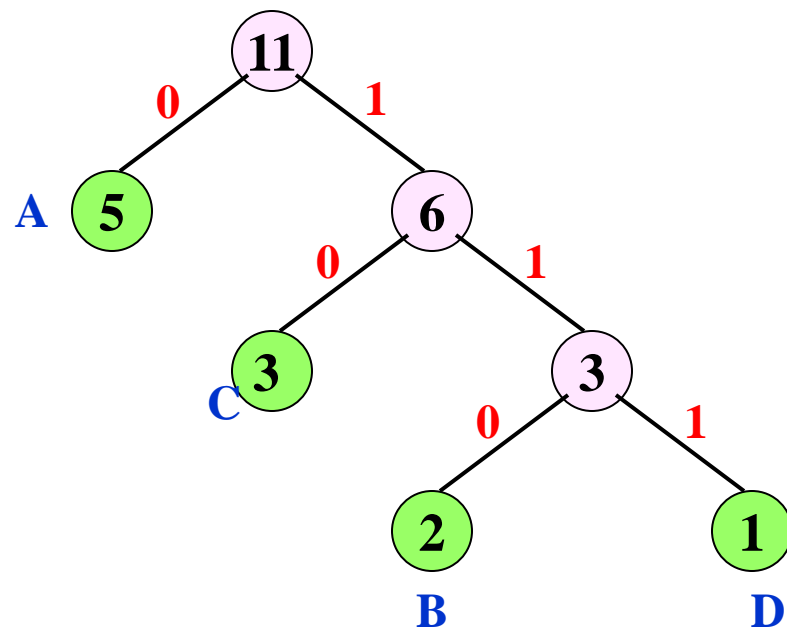
Huffman 编码

A: 0

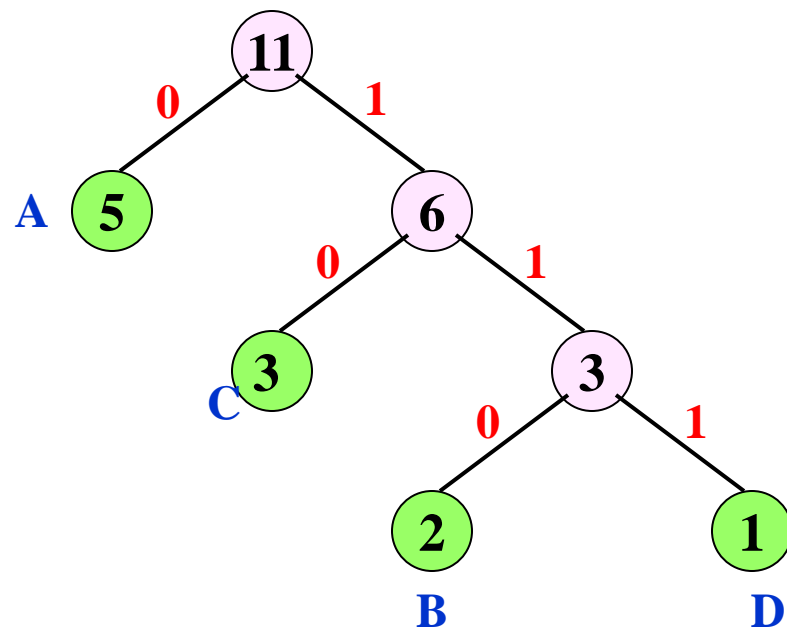
B: 110

C: 10

D: 111



- 等长编码总长度：22bit
- 报文前缀码长度 = 哈夫曼树的带权路径长度WPL  
= 11 + 6 + 3  
= 20bit
- 压缩比  
= 20/22 = 91%



## 【性质1】

**Huffman树根结点的权值=报文的字符总数。**

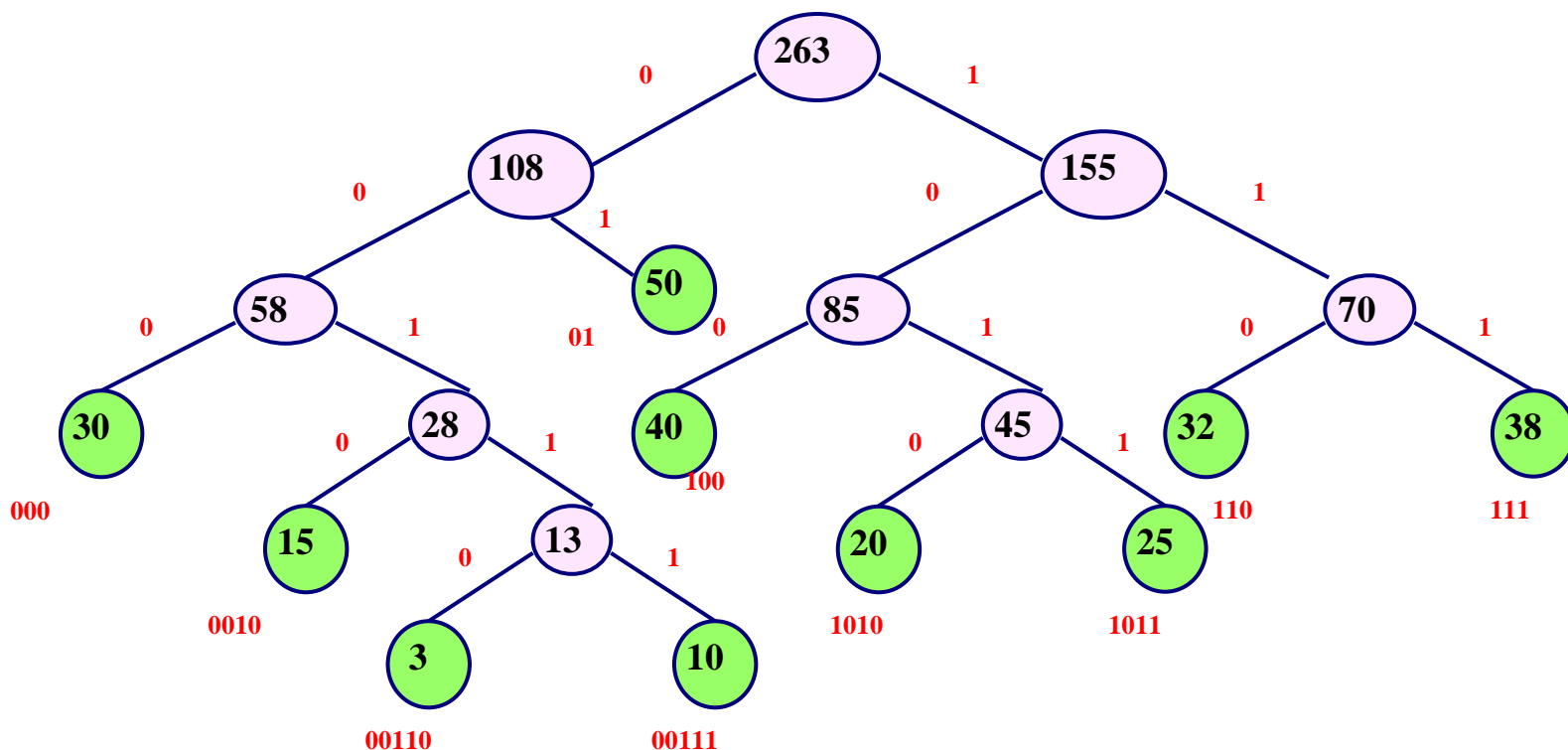
## 【性质2】

**Huffman的WPL=Huffman编码总长度。  
=分支结点权值之和**

## 【性质3】

**Huffman树没有 1 度结点。**（国际满二叉树）

**【例】**已知一个文件中共有**10**个不同的字符，各字符出现的频率分别是{**50** , **40** , **38** , **32** , **30** , **25** , **20** , **15** , **10** , **3**}。试重新为各字符编码，使文件最短，以节省存储空间。



- 文件等长码长度 ( 4bit等长码 )

$$263 \times 4 = 1052$$

- 文件前缀码长度 = 哈夫曼树的带权路径长度WPL ( 边数=编码位数 )

$$\begin{aligned} &= ( 3 + 10 ) \times 5 + ( 15 + 20 + 25 ) \times 4 + \\ &\quad ( 30 + 32 + 38 + 40 ) \times 3 + 50 \times 2 \\ &= 825 \end{aligned}$$

- 压缩比

$$= 825 / 1052 = 78\%$$

编码

50: 01

40: 100

38: 111

32: 110

30: 000

25: 1011

20: 1010

15: 0010

10: 00111

3: 00110

## ■ 二叉树的一维数组存储—哈夫曼树结点

- ☞ 对元素进行封装：数据元素、左右孩子指针（下标）、父结点指针（下标）。
- ☞ 将封装好的结点存入一维数组（顺序表）。

```
typedef struct hfmNode
```

```
{
```

```
    elementType data;           //元素值
```

```
    weightType weight;         //结点权值
```

```
    int parent,lChild,rChild;   //父结点、左右孩子结点指针
```

```
}hfmNode;
```

## 【思考问题】

- ➡  $n$ 个权值构造的哈夫曼树共有多少个结点？
- ➡ 哈夫曼树如何存储？
- ➡ 如何实现构造哈夫曼树？
- ➡ 读入一个文本文件，如何进行哈夫曼编码、译码？--课程设计课题。

# 本章小结

- 二叉树、树、森林、哈夫曼树基本概念
- 二叉树的5个性质
- 二叉树、树、森林的各种存储结构
- 二叉树遍历
  - ☞ 遍历是其它运算的基础，要熟练掌握
  - ☞ 进一步理解递归
  - ☞ 层次遍历与其他遍历策略不同



## ■ 线索二叉树

- ☞ 实质是建立结点之间的前驱与后继的关系
- ☞ 线索化
- ☞ 找前驱、后继算法
- ☞ 遍历实现

## ■ 树、森林与二叉树的转换

## ■ 树、森林各种算法的实现

## ■ 哈夫曼树的特性、建立、哈夫曼编码

Thank you !

