

大家好，我是蓝蓝，这是我们一期数据结构应用题专题的第一天。day08/15

蓝蓝B站首页: [蓝蓝希望你上岸呀B站首页](#)

蓝蓝公众号: [应用题训练营专题](#)

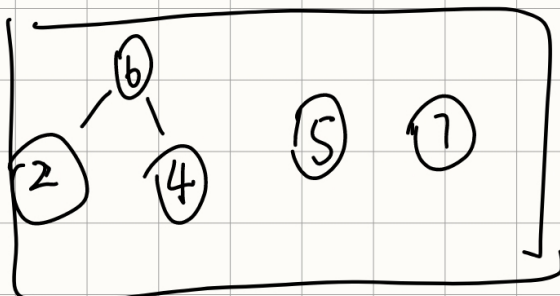
01、哈夫曼树的构造，写出构造过程

知识星球专属

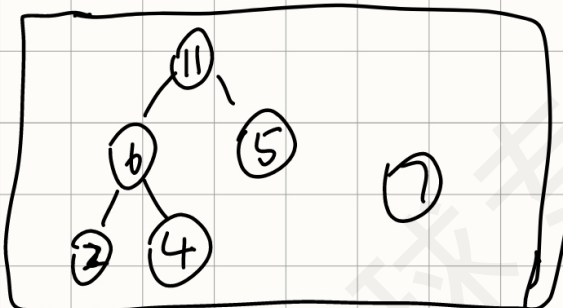
7, 5, 2, 4

哈夫曼树的构造，每次从集合中选择最小代  
价的两颗树，作为新结点的子树，新结点的  
权值等于两颗子树的权值之和，最后将  
新结点加入集合，直到集合内只剩一棵树为  
止

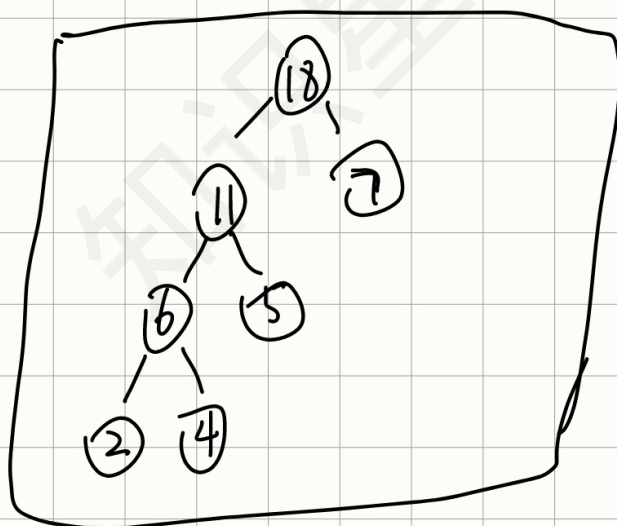
(1)



(2)



(3)



以2012年真题举例



## 哈夫曼树与并查集

### 1. 构造哈夫曼树的过程

① 选择  $n$  个结点中权值最小的两个结点, 优先组成哈夫曼树的左右子树, 根为二者权值之和

② 将①中根结点加入结点队列, 再次选择权值最小的两个结点, 重复①的过程, 直到队列中结点全部归并结束

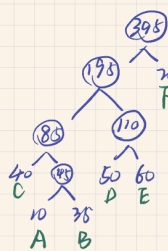
eg:

02. 【2012 统考真题】设有 6 个有序表 A, B, C, D, E, F, 分别含有 10, 35, 40, 50, 60 和 200 个数据元素, 各表中的元素按升序排列。要求通过 5 次两两合并, 将 6 个表最终合并为 1 个有序表, 并使最坏情况下比较的总次数达到最小。请回答下列问题:

- 1) 给出完整的合并过程, 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  $n$  ( $n \geq 2$ ) 个不等长有序表的合并策略, 并说明理由。

两个有序表  $m, n$  的最坏比较次数为  $m+n-1$   
最好比较次数为  $\min(m, n)$

要使最坏情况下比较总次数最小, 则优先构造哈夫曼树

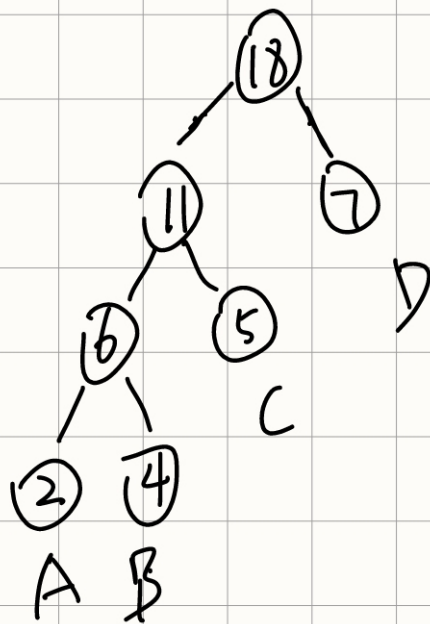


- ① A 与 B 合并 10+35=45
- ② C 与 45 合并 40+45=85
- ③ D 与 E 合并 50+60=110
- ④ 85 与 110 合并 85+110=195
- ⑤ 195 与 F 合并 195+20=395

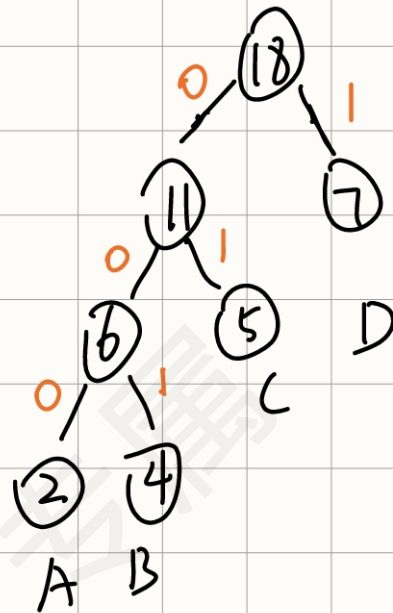
总比较次数为 815 次

②  $n$  个不等长有序表两两合并时, 表长不同, 最坏情况下差的比较次数, 依表长的合并次数, 所以用哈夫曼树的构造思想来依次合并最短的两表, 此时合并效率最高

02、根据哈夫曼树, 写出哈夫曼编码



根据哈夫曼树我们可以规定0为左子树，1为右子树，或者0为右子树，1为左子树，下面给出示例

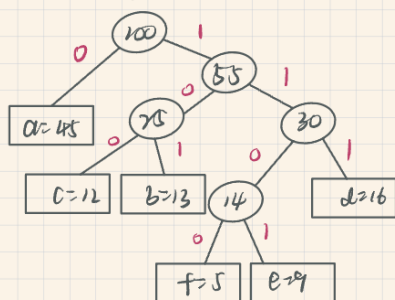


因此，可以得到哈夫曼编码

A: 000  
B: 001  
C: 01  
D: 1

## 2. 哈夫曼编码 又称 前缀编码

若构造哈夫曼树为



则各子树编码为

a 0  
b 101  
c 100  
d 111  
e 1101  
f 1100

$$\begin{aligned} \text{平均WPL} &= \frac{1}{8} [1 \times 45 + 3 \times 13 + 2 \times 16 + 4 \times (5+9)] \\ &= \frac{1}{8} \times 210 \\ &= \frac{105}{4} \end{aligned}$$

$$\begin{aligned} \text{平均哈夫曼长度为} &= \frac{1}{8} (1 + 3 \times 2 + 2 \times 3) \\ &= \frac{13}{8} \end{aligned}$$

由于0和1为左右子树的哪一个并未明确规定故哈夫曼编码不为1但WPL为相同且最优

### 03、并查集的基本操作

#### 并查集的基本操作

构造并查集可以  
降低树高

3 并查集

用树的关系表示作为并查集的存储结构, 每个集合以一棵树表示

eg  $S = \{0, 1, 2, 3, 4, 5, 6\}$  则  $S_1$   $S_2$   $S_3$

初始状态值均为 -1

0	1	2	3	4	5	6
-1	-1	-1	-1	-1	-1	-1

若  $S_1 = \{0, 3, 4\}$   $S_2 = 2$   $S_3 = \{1, 5, 6\}$

代表该树的根节点取总号

代表 6 指向 1

#### 并查集的定义与查找合并代码

① 并查集定义

```
#define SIZE 1000
int ufsets[SIZE]; // 双亲指针数组
```

② 初始化操作

```
void Initial(int sz){
    for(int i=0; i<sz; i++)
        sz[i] = -1;
}
```

③ Find 操作

(在查集 S 中找并返回 编号 x 的根)

```
int Find(int sz, int x){
    while (sz[x] >= 0)
        x = sz[x]; // 一路向上
    return x;
}
```

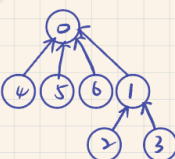
最好时间复杂度  $O(1)$ , 最坏  $O(n)$

④ Union 操作

求两个不相交并查集

```
void Union(int sz, int Root1, int Root2){
    if (Root1 == Root2) return;
    sz[Root2] = Root1; // 将根 Root2 接到另根 Root1 下
}
```

eg



0	1	2	3	4	5	6
-1	0	1	1	0	0	0

$sz[1] = 0;$

