哈夫曼树与哈夫曼编码

主讲人：邓哲也

大纲

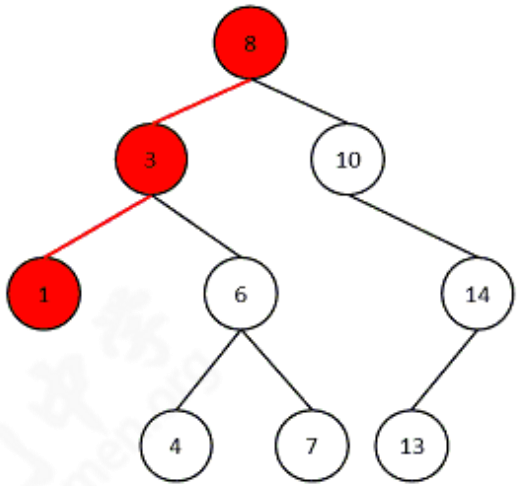
- 哈夫曼树
- 哈夫曼树的构建
- 哈夫曼树到哈夫曼编码
- 哈夫曼树的实现

哈夫曼树

哈夫曼（Huffman）树是一种特殊结构的二叉树，由Huffman树设计的二进制前缀编码，也称为Huffman编码在通信领域有着广泛的应用。

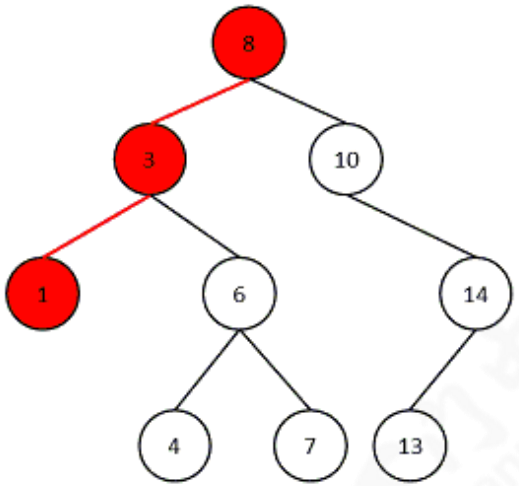
哈夫曼树的基本概念

- 路径是指在一棵树中，从一个节点到另一个节点之间的分支构成的通路。
- 对于如图所示二叉树，红色的是从节点 8 到节点 1 的路径。



哈夫曼树的基本概念

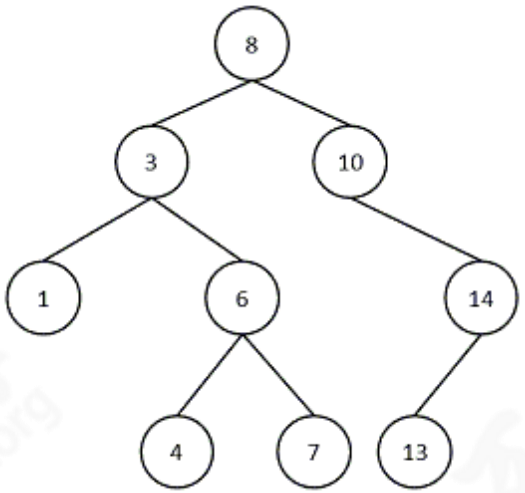
- 路径长度是指路径上边的数目，如图，路径长度为 2。
- 节点的权指的是为树中的每一个节点赋予的一个非负的值，如图中每一个节点中的值。
- 节点的带权路径长度指的是从根节点到该节点之间的路径长度与该节点权的乘积：如对于1节点的带权路径长度为 2，对于7节点的带权路径长度为 21。
- 树的带权路径长度指的是所有叶子节点的带权路径长度之和。

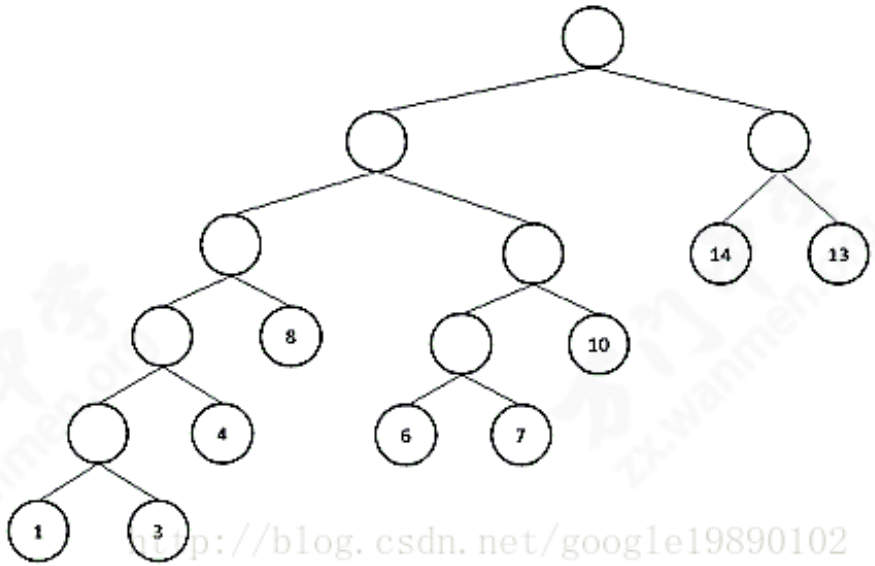


哈夫曼树的基本概念

Huffman树的定义：

给定 n 个带权值的点作为叶子节点，构造一颗二叉树。若这颗二叉树的带权路径长度达到最小值，则称这颗二叉树为最优二叉树，也称为 Huffman 树。





哈夫曼树的构建

由上述的Huffman树可知：节点的权越小，其离树的根节点越远。那么应该如何构建Huffman树呢？

假设传输的报文为：“AFTERDATAEARAREARTAREA”，现在需要对该报文进行编码。

以上述报文为例，首先需要统计出每个字符出现的次数作为节点的权:

A-8，R-5，E-4，T-3，F-1，D-1

哈夫曼树的构建

哈夫曼算法：

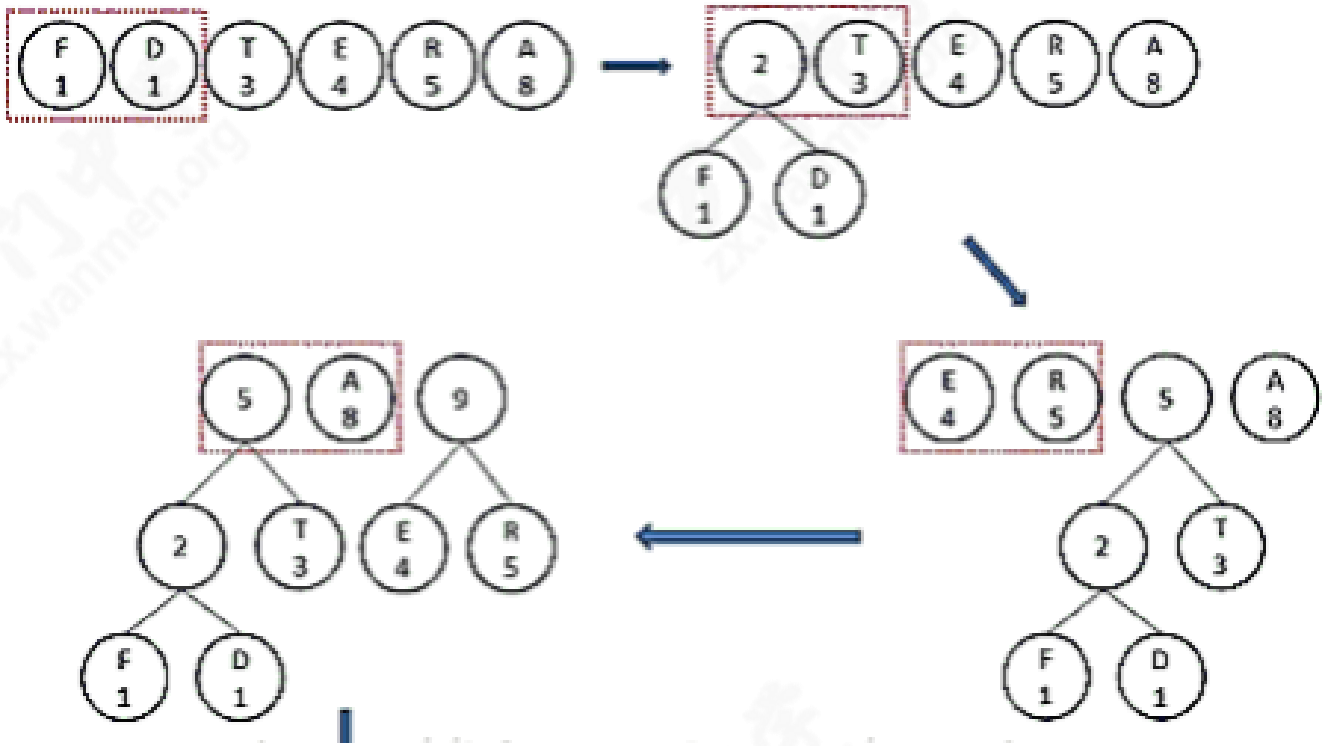
1. 根据给定的 n 个权值 {w1, w2, …, wn}，构造 n 棵二叉树的集合 F = {T1, T2, …, Tn}，其中每颗二叉树都只有一个节点。

2. 在 F 中选取其根节点权值最小的两颗二叉树，分别作为左、右子树构造一颗新的二叉树，并将这颗二叉树的根节点的根设为左右子树根节点的权值之和。

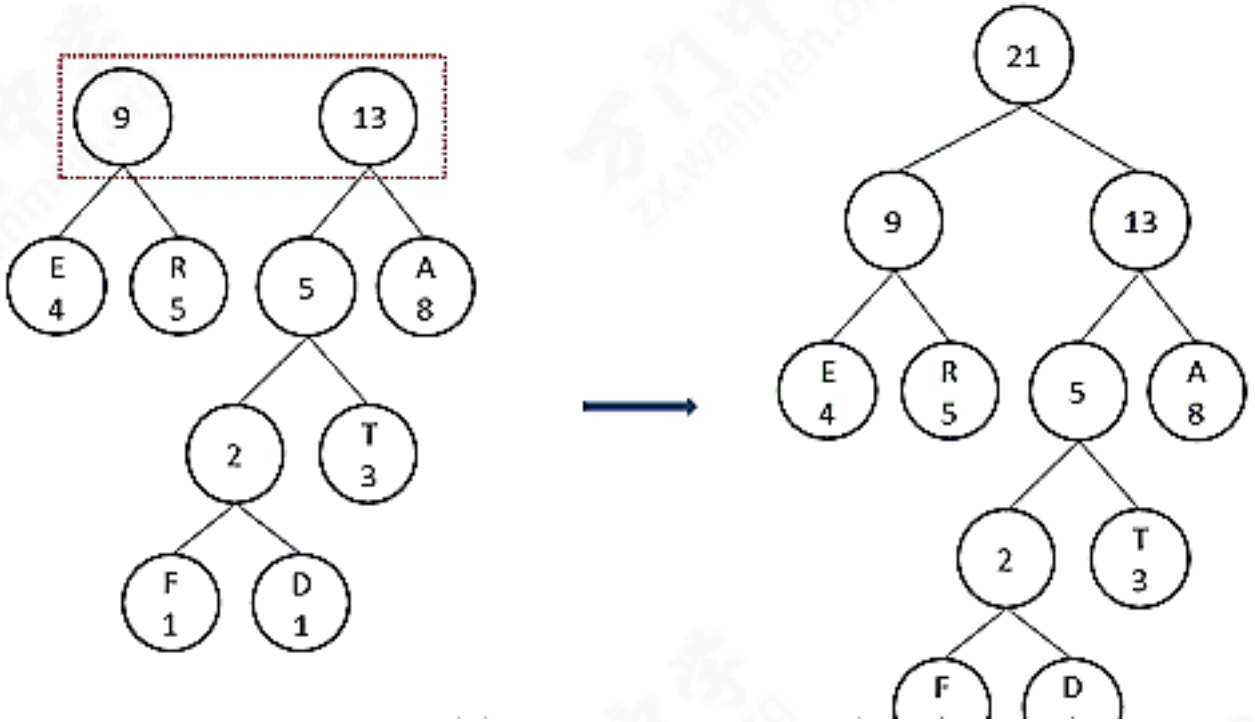
3. 从 F 中删去这两颗树，同时加入刚生成的新树。

4. 重复 2 和 3 两步，直到 F 中只有一棵树为止。

哈夫曼树的构建


F 1
D 1
T 3
E 4
R 5
A 8
2
T 3
E 4
R 5
A 8
F 1
D 1
E 4
R 5
S
A 8
2
T 3
F 1
D 1
S
A 8
9
2
T 3
E 4
R 5
F 1
D 1

哈夫曼树的构建



哈夫曼树到哈夫曼编码

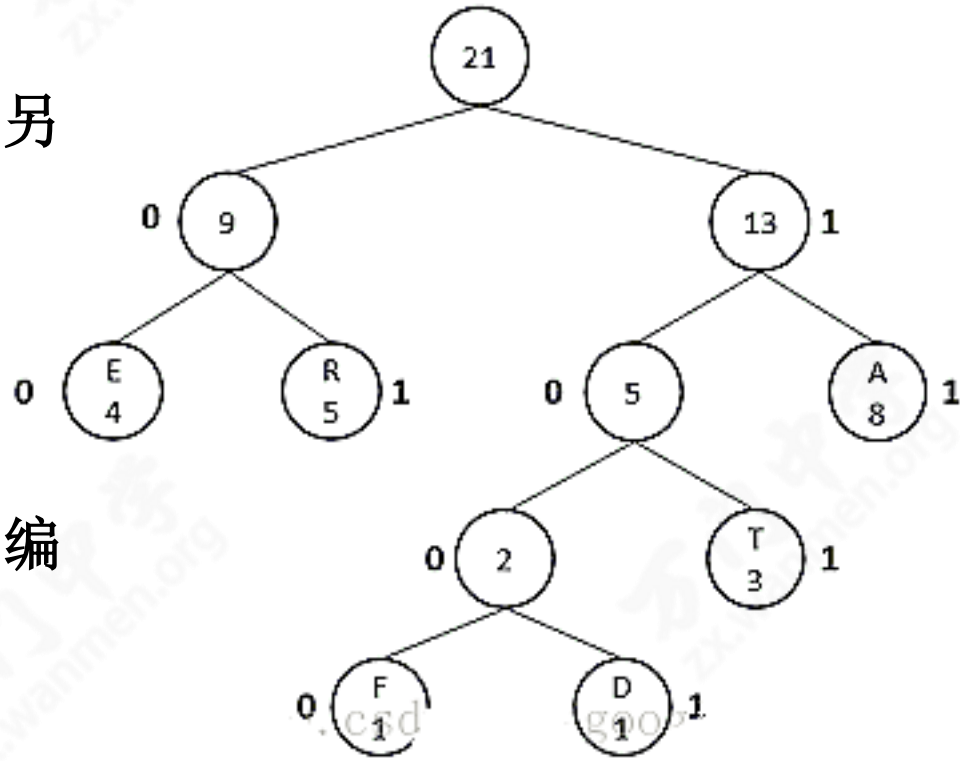
有了上述的Huffman树的结构，现在我们需要利用Huffman树对每一个字符编码，该编码又称为Huffman编码。

Huffman编码是一种前缀编码，即一个字符的编码不是另一个字符编码的前缀。在这里约定：

将权值小的最为左节点，权值大的作为右节点

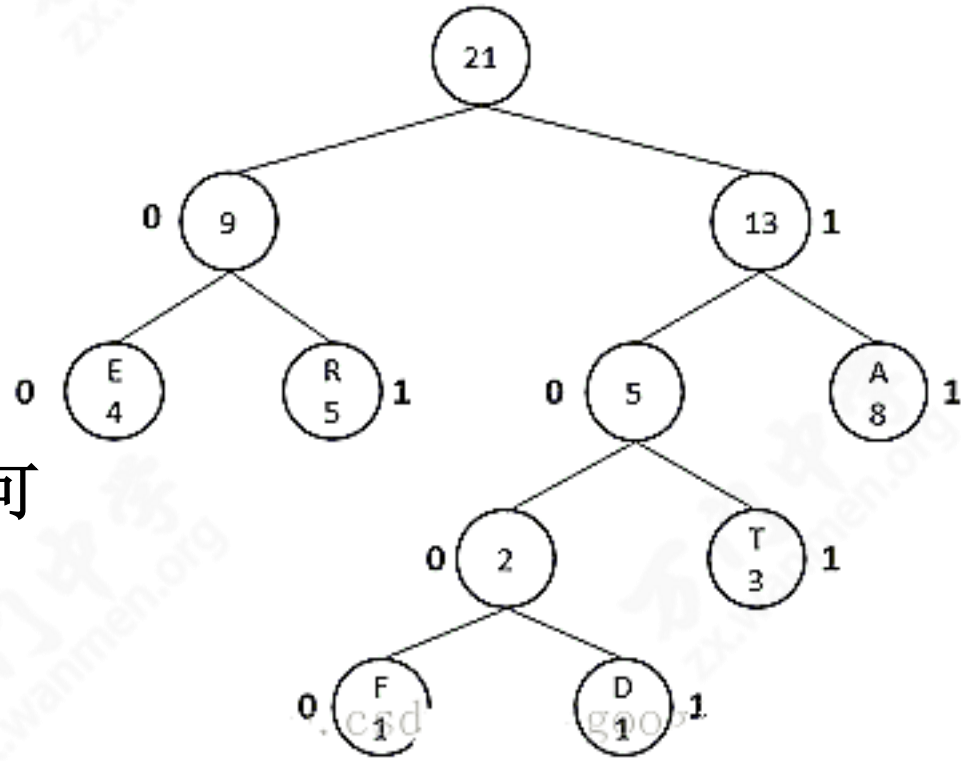
左孩子编码为0，右孩子编码为1

从这张图中可以得到，E 节点的编码为 00，T 节点的编码为 101.



哈夫曼树的实现

- 实现哈夫曼树，用一个堆即可。
- 用堆来维护 F 中每一个根节点的权值。
- 每次从堆中取出最小的两个元素。
- 把两颗树合并后，将新的权值加入堆中。
- 直到堆中只剩一个元素。
- 此时哈夫曼树的信息也维护好了，遍历一遍即可得到所有词的编码。



下节课再见