6.7二叉树的应用

哈夫曼树(Huffman Tree)

- 结点的路径长度：该结点到树根之间的路径上，所经过的分支数目为此结点的路径长度。
- 树的路径长度：树根到每个结点的路径长度之和。
- 结点的带权路径长度：该结点到树根之间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，记为：$WPL = w_1k_1 + w_2k_2 + \ldots + w_ik_i + w_nk_n$。
- 最优二叉树（哈夫曼树）：设n个权值$\{w_1, w_2, \ldots, w_n\}$, 构造一棵有n个叶子结点的二叉树，每个叶子结点带权w_i, 则其中带权路径长度WPL最小的二叉树称为最优二叉树（哈夫曼树）。

6.7二叉树的应用（续）

哈夫曼算法(Huffman Algorithmic)

输入： n个权值$\{w_1, w_2,\ldots, w_n\}$；

输出： 一棵哈夫曼树

算法：

- **步骤一：** 根据给定的n个权值$\{w_1, w_2,\ldots, w_n\}$构成n棵二叉树的集合$F=\{T_1, T_2, \ldots, T_n\}$，其中每棵二叉树$T_i$中只有一个带权为$w_i$的根结点，其左右子树均为空。
- **步骤二：** 在F中选取两棵根结点的权值最小的树作为左、右子树构造一棵新的二叉树，且置新的二叉树的根结点的权值为其左、右子树上根结点的权值之和。
- **步骤三：** 在F中删除这两棵树，同时将新得到的二叉树加入F中。
- **步骤四：** 重复步骤二与三，直到F只含有一棵树为止。

6.7二叉树的应用（续）

DS/SS

哈夫曼算法(Huffman Algorithm)

给定权的集合： {15,3,14,2,6,9,16,17}

{15,3,14,2,6,9,16,17}

{15,5,14,6,9,16,17}

{15,11,14,9,16,17}

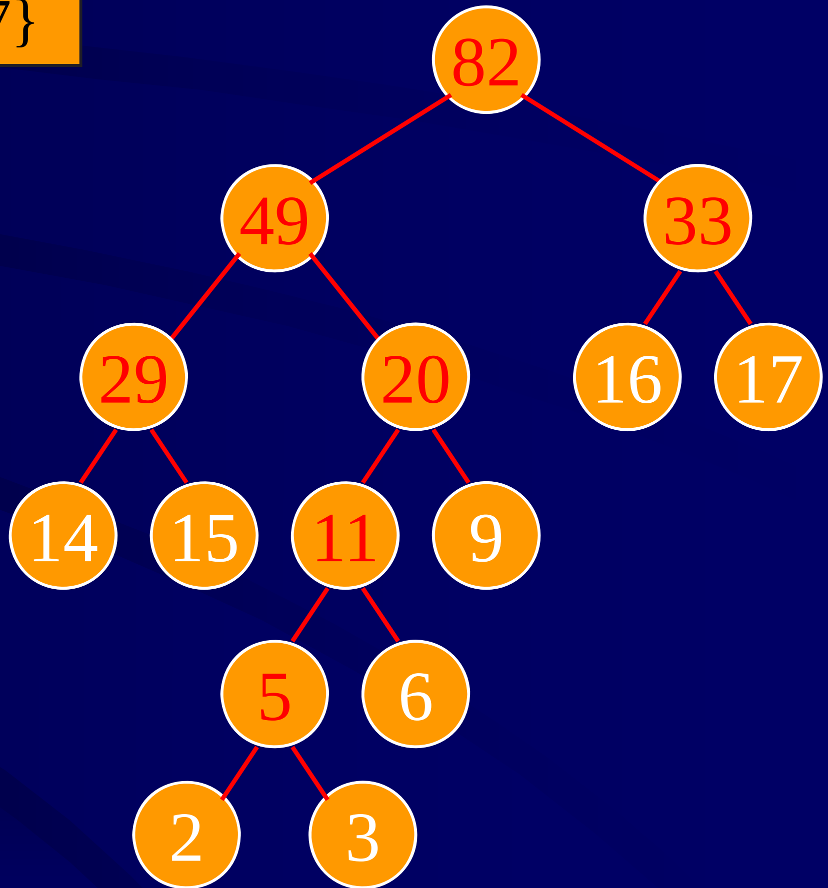
{15,14,20,16,17}

{29,20,16,17}

{29,20,33}

{49,33}

{82}



WPL =(2+3)*5+6*4+(9+14+15)*3+(16+17)*2 = 229

哈夫曼编码(Huffman Code)

- **等长编码**：每个被编码对象的码长相等。

  优点：码长等长，易于解码；

  缺点：被编码信息总码长较大，尤其是当存在许多不常用的被编码对象时

- **前缀编码**：任一个被编码对象的编码都不是另一个被编码对象的编码的前缀。

  优点：当存在被编码对象使用概率不同时，被编码信息总码长可以减小；

  缺点：码长不等长，不易于解码

6.7二叉树的应用（续）

DS/SS

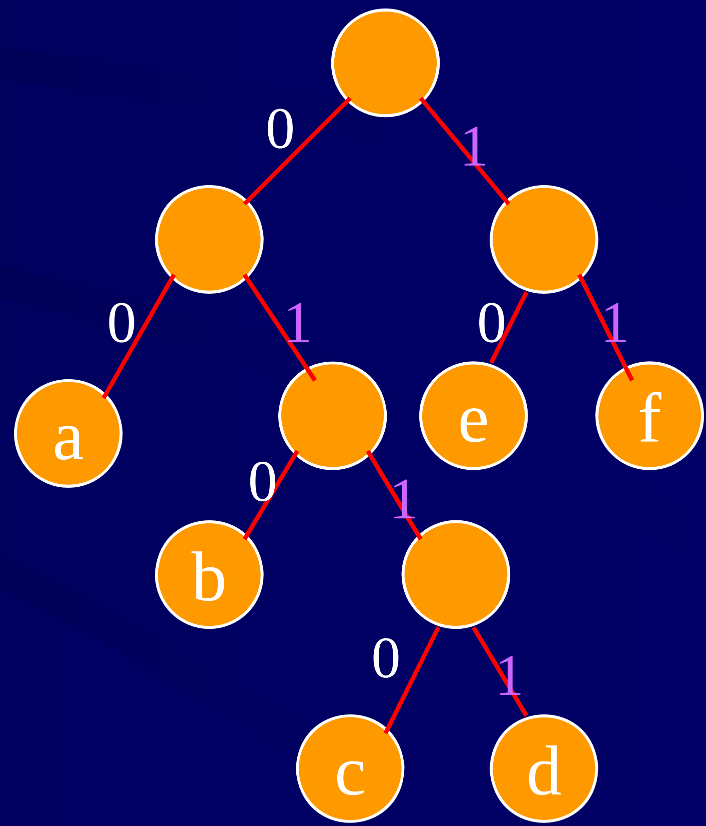
哈夫曼编码(Huffman Code)---前缀编码

利用二叉树来设计二进制的前缀编码：

被编码对象出现在二叉树的叶子结点。约定左分支表示字符“0”，右分支表示字符“1”，则可以从根结点到叶子结点的路径上分支字符组成的字符串作为该叶子结点对应对象的编码。

这种编码的译码过程从根开始。沿着某条分支走到叶子结点时，走过的二进制字符串即译为该叶子结点对应的对象。然后循环扫描总编码后的字符串。

a(00)
b(010)
c(0110)
d(0111)
e(10)
f(11)



6.7二叉树的应用（续）

DS/SS

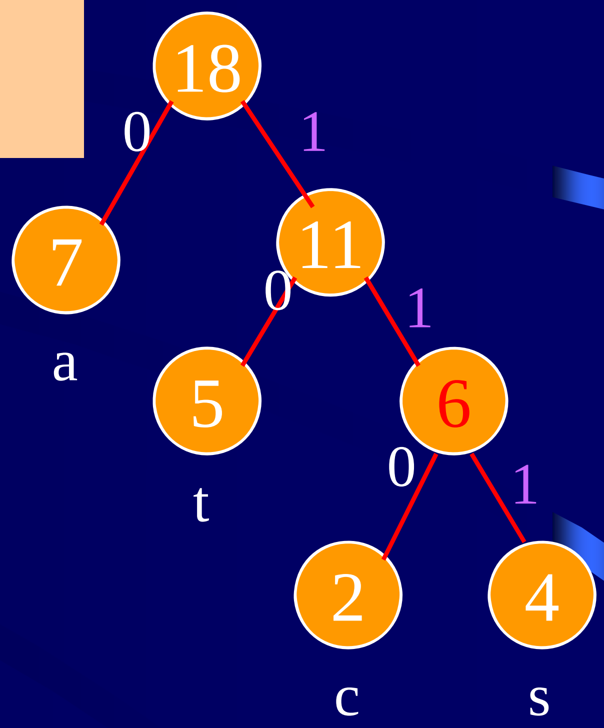
哈夫曼编码与译码

利用哈夫曼算法可以得到总码长最短的二进制前缀编码，即为哈夫曼编码。

设某次编码应用中不同被编码对象有n种，以此n种被编码对象出现的频率作权，构造出的哈夫曼树即为这n种对象的编码树，由此可得到其二进制的前缀编码。

Set = {c,a,s,t}

W ={2,7,4,5}



c(110)

a(0)

s(111)

t(10)

假定用于通讯的电文如下，现在要对其编码，采用哈夫曼编码。写出编码后的二进制串。

电文： castcatssatatatasa

电文编码：

11001111011001011111101001001001110