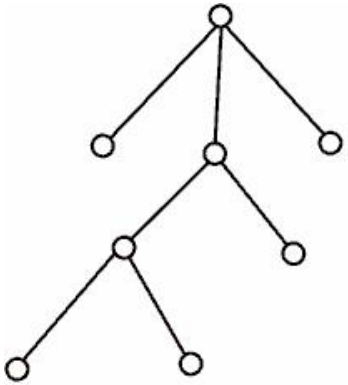
16.3 根树及其应用

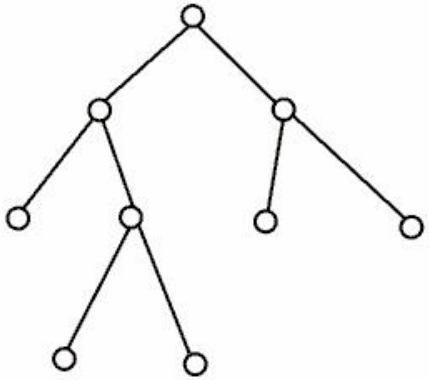
根树 及其定义

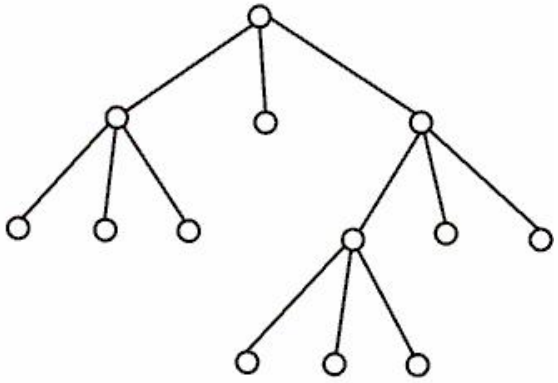
定义16.6 T是有向树（基图为无向树）

(1) T 为**根树**——T 中一个顶点入度为0，其余的入度均为1.

根树的画法——树根放上方，省去所有有向边上的箭头







根树 及其定义

(2) 树根——入度为0的顶点

(3) 树叶——入度为1，出度为0的顶点

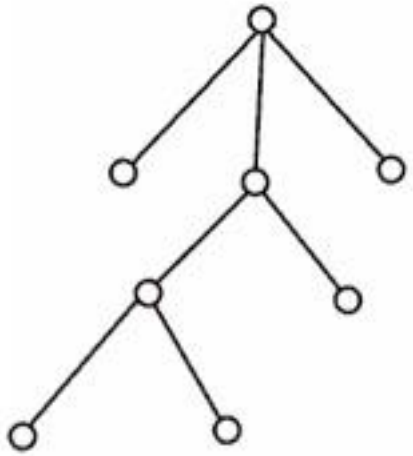
(4) 内点——入度为1，出度不为0的顶点

(5) 分支点——树根与内点的总称

(6) 顶点v的层数——从树根到v的通路长度

(7) 树高——T 中层数最大顶点的层数

(8) 平凡根树——平凡图



家族树与根子树

定义16.7 T 为非平凡根树

(1) 祖先与后代

(2) 父亲与儿子

(3) 兄弟

定义16.8 设v为根树T中任意一顶点，称v及其后代的导出子图为以v为根的根子树.

根树的分类

(1) T 为有序根树——同层上顶点标定次序的根树

(有序：每层的顶点，不论是分支点或者树叶，存在顺序)

(2) 分类

① r 叉树——每个分支点至多有r 个儿子

② r 叉有序树——r 树是有序的

③ r 叉正则树——每个分支点恰有r 个儿子

(正则r叉：儿子个数相同)

④ r 叉正则有序树

⑤ r 叉完全正则树——树叶层数相同的r叉正则树

(完全r叉：要求每个树叶的层数均为树高)

⑥ r 叉完全正则有序树——r叉正则 + 完全 +有序

最优二叉树

定义16.9 设2叉树T有t片树叶$v_1, v_2, \ldots, v_t$，权分别为$w_1, w_2, \ldots, w_t$，称 $W(t)=\sum_{i=1}^{t} w_i l(v_i)$ 为T的权，其中$l(v_i)$是v_i的层数.

在所有有t片树叶，带权$w_1, w_2, \ldots, w_t$的2叉树中，权最小的2叉树称为最优2叉树.

最优二叉树

求最优树的算法—— Huffman算法

给定实数$w_1, w_2, \ldots, w_t$，且$w_1 \le w_2 \le \ldots \le w_t$.

(1) 连接权为w_1, w_2的两片树叶，得一个分支点，其权为w_1+w_2.

(2) 在$w_1+w_2, w_3, \ldots, w_t$中选出两个最小的权，连接它们对应的顶点(不一定是树叶)，得新分支点及所带的权.

(3) 重复(2)，直到形成$t-1$个分支点，t片树叶为止.

最优二叉树

例 5　求带权为1, 1, 2, 3, 4, 5的最优树.

最佳前缀码

定义16.10 设$\alpha_1, \alpha_2, \ldots, \alpha_{n-1}, \alpha_n$是长度为 n 的符号串

(1) 前缀——α_1, $\alpha_1\alpha_2$, …, $\alpha_1\alpha_2\ldots\alpha_{n-1}$

(2) 前缀码——$\{\beta_1, \beta_2, \ldots, \beta_m\}$中任何两个元素互不为前缀

(3) 二元前缀码——β_i (i=1, 2, …, m) 中只出现两个符号，如0与1.

问题：如何产生二元前缀码?

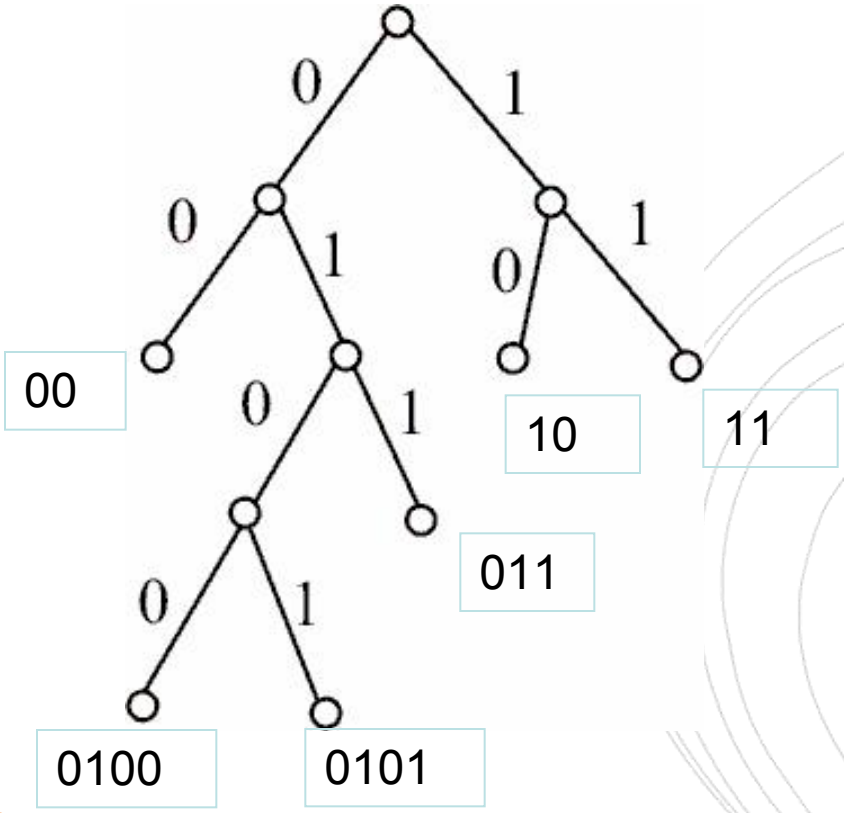
定理16.6 一棵2叉树产生一个二元前缀码.

推论 一棵正则2叉树产生惟一的前缀码（按左子树标0, 右子树标1）

最佳前缀码

图所示二叉树产生的前缀码为

{ 00, 10, 11, 011, 0100, 0101 }



用Huffman算法产生最佳前缀码

例6 在通信中，八进制数字出现的频率如下：

0：25%　　1：20%

2：15%　　3：10%

4：10%　　5：10%

6：5%　　7：5%

求传输它们的最佳前缀码，并求传输$10n$（$n\geq 2$）个按上述比例出现的八进制数字需要多少个二进制数字？若用等长的（长为3）的码字传输需要多少个二进制数字？

波兰符号法与逆波兰符号法

行遍或周游根树T——对T的每个顶点访问且仅访问一次.

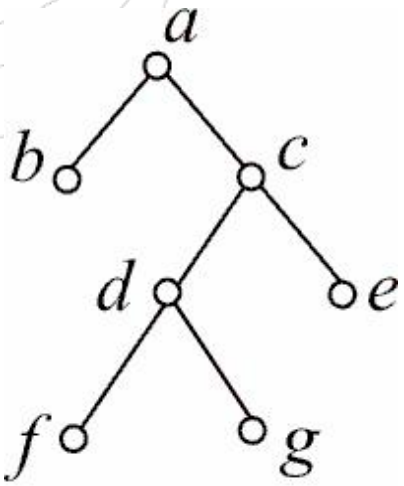
对2叉有序正则树的周游方式:

① 中序行遍法——次序为: 左子树、根、右子树

② 前序行遍法——次序为: 根、左子树、右子树

③ 后序行遍法——次序为: 左子树、右子树、根

(参考P339, 图16.12.)



练习:

对图所示根树按中序、前序、后序行遍法访问结果分别为:

$b\ \underline{a}\ (f\ \underline{d}\ g)\ \underline{c}\ e$,

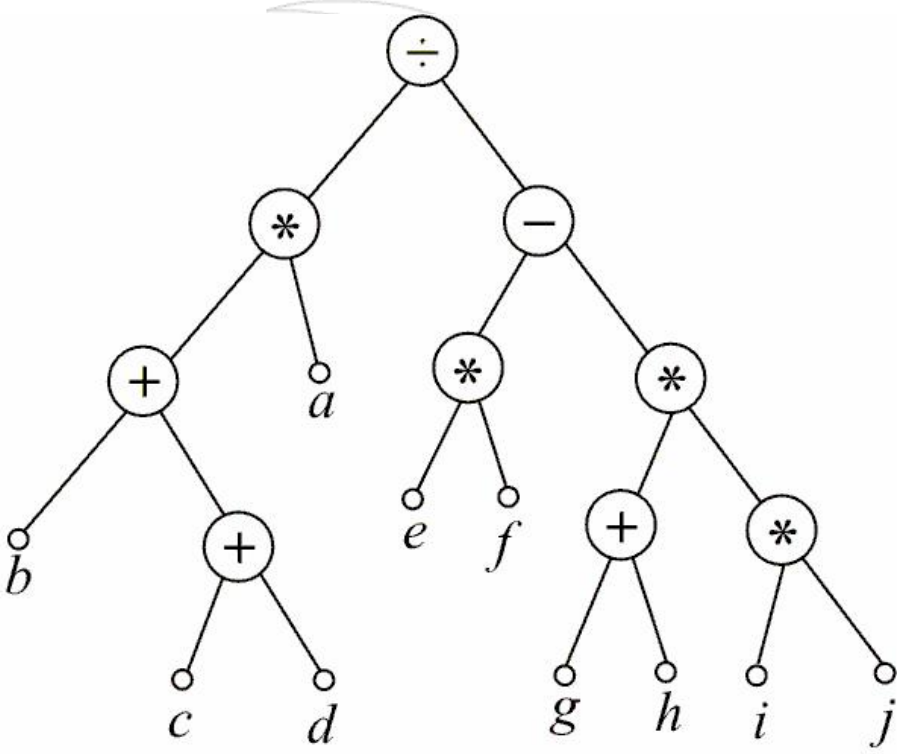
$\underline{a}\ b\ (\underline{c}\ (\underline{d}\ f\ g)\ e)$,

$b\ ((f\ g\ \underline{d})\ e\ \underline{c})\ \underline{a}$

用2叉有序正则树存放算式

存放规则

- 最高层次运算放在树根
- 后依次将运算符放在根子树的根上
- 数放在树叶上
- 规定：被除数、被减数放在左子树树叶上



算式 ((*b*+(*c*+*d*))**a*)÷((*e***f*)−(*g*+*h*)*(*i***j*))
存放在图所示2叉树上.

波兰符号法

1. 波兰符号法

按前序行遍法访问存放算式的2叉有序正则树，其结果不加括号，规定每个运算符号与其后面紧邻两个数进行运算，运算结果正确. 称此算法为波兰符号法或前缀符号法. 对上图的访问结果为

$$\div * + b + c\ d\ a - * e\ f * + g\ h * i\ j$$

2. 逆波兰符号法

按后序行遍法访问，规定每个运算符与前面紧邻两数运算，称为逆波兰符号法或后缀符号法. 对上图的访问结果为

$$b\ c\ d + + a * e\ f * g\ h + i\ j * * - \div$$