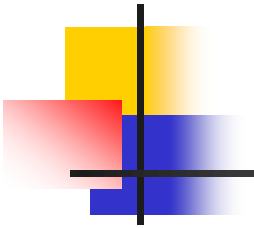


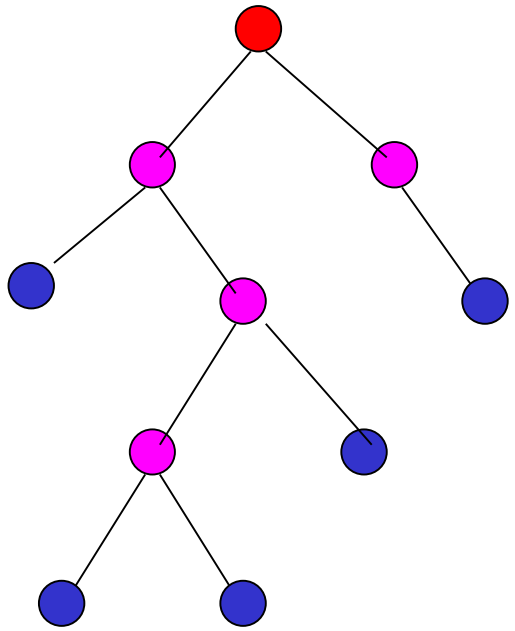
根树

- 1. 根树
- 2. 根树的周游
- 3. 最优树, Huffman算法
- 4. 最佳前缀码

根树(rooted tree)

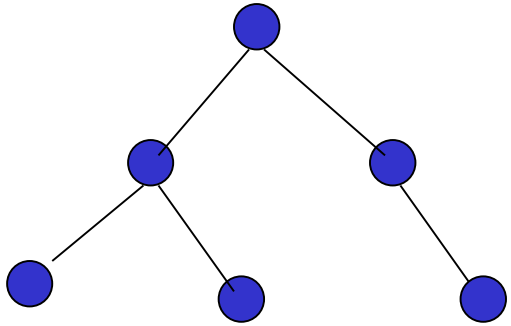
- 有向树: 基图是树的有向图
- 根树(rooted tree): 具有以下特点的有向树

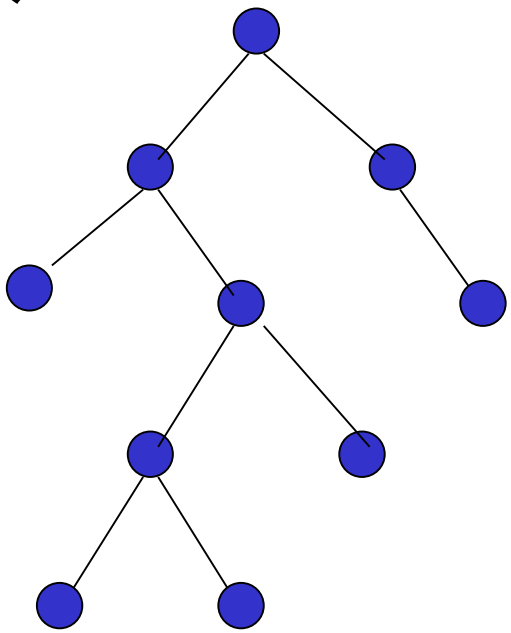
名称		入度	出度	个数
分支点	树根	0	>0	1
	内点	1	>0	≥0
树叶		1	0	≥0



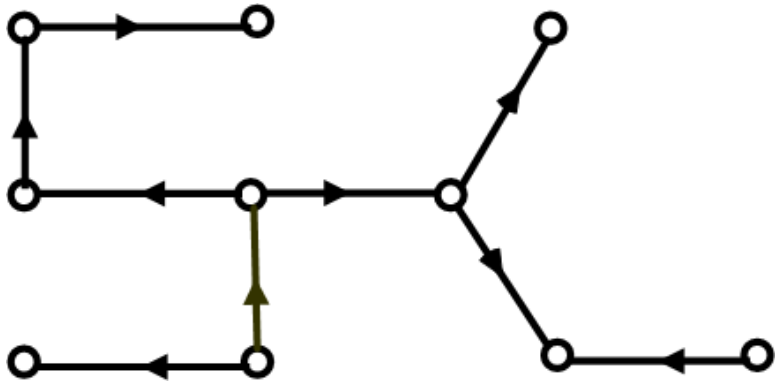
层数与树高

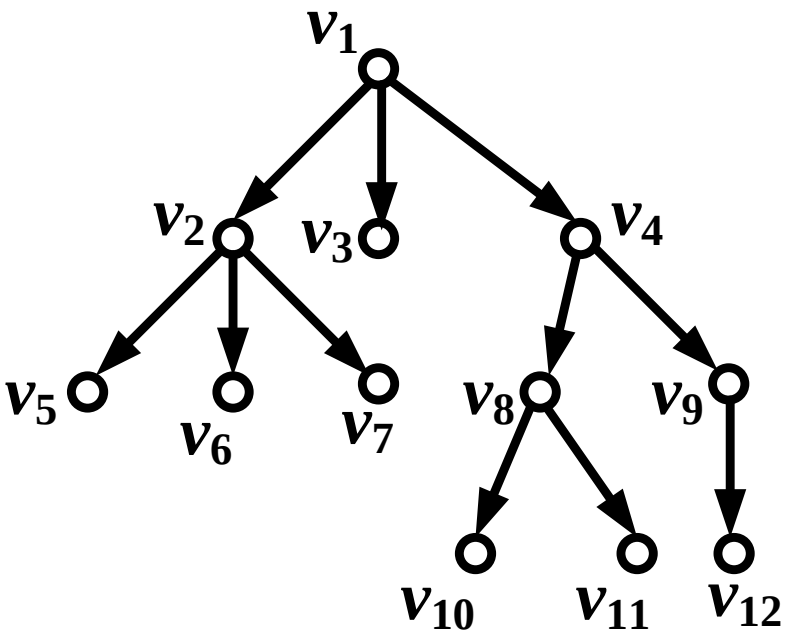
- **画法:** 树根在最上方, 省略边的方向(从上到下)
- **顶点*v*的层数**(level): L(*v*)=从树根到*v*路径长度
- **树高**(height): *h*(T)=顶点的最大层数





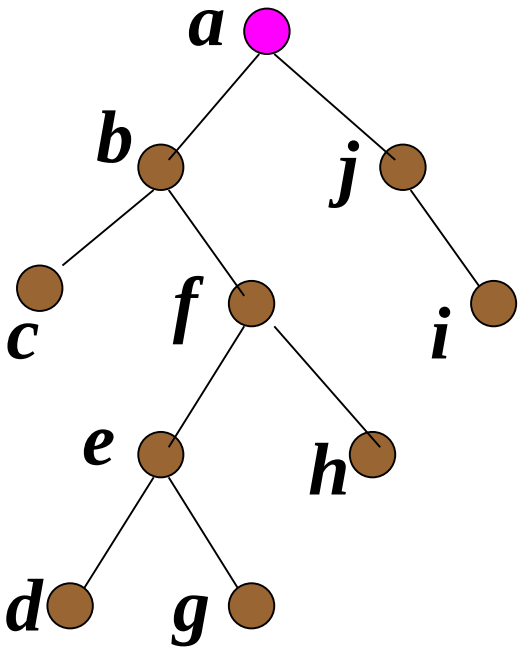
举例





家族树

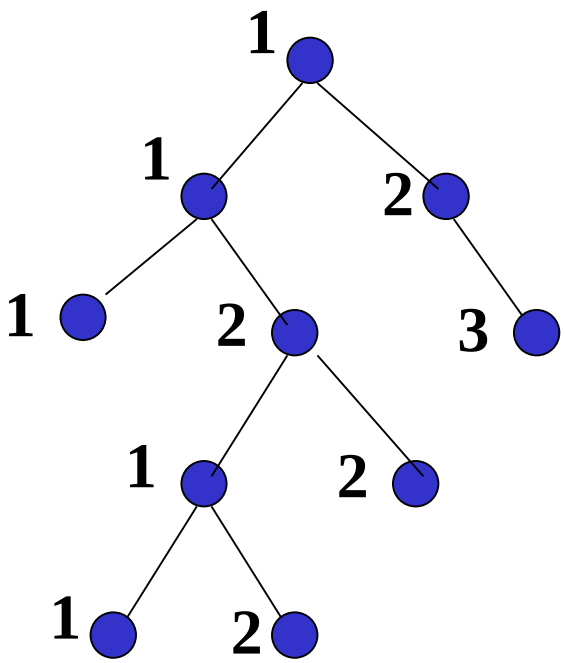
- 儿子: u在上方与v相邻, v是u的儿子
- 父母: u在上方与v相邻, u是v的父母
- 兄弟: u与v有相同父母, u是v的兄弟
- 祖先: 从u可达v, u是v的祖先
- 后代: 从u可达v, v是u的后代
- 根子树T_v:设v是根树的一个顶点且不是树根, 称v及其所有后代的导出子图为以v为根的根子树。



例 在右图的根数中, 求c的父母, f的子女, f的兄弟, e的所有祖先, b的所有后代, 所有内点以及所有树叶,以 f 为根的子树是什么?

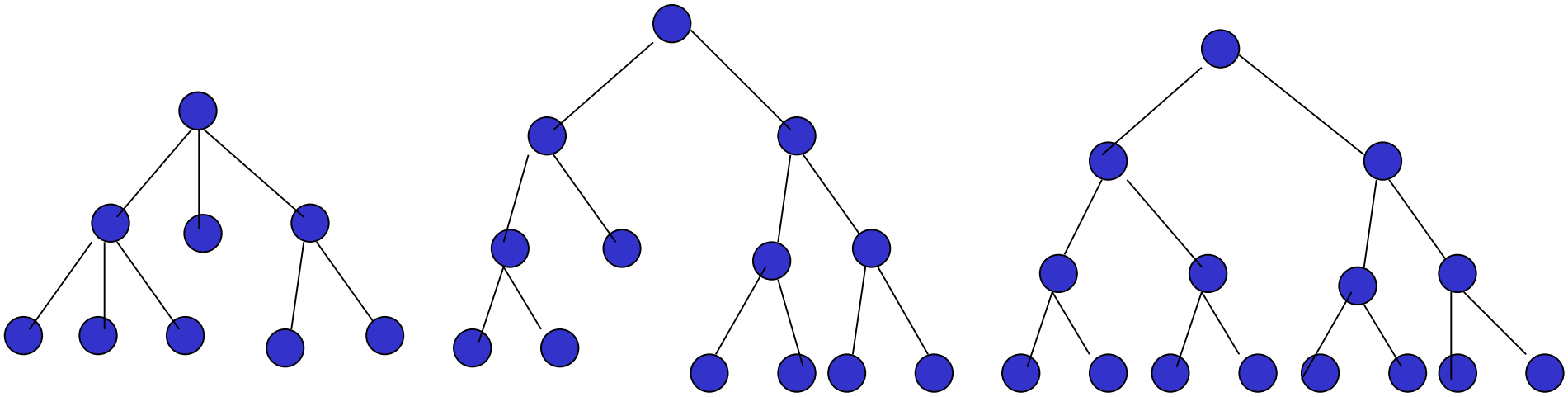
有序树(ordered tree)

- 有序树: 给相同层数的顶点标上次序的根树



r叉树(t-ary tree)

- *r*叉树: $\forall v \in V(T)$, $d^+(v) \le r$
- 正则(regular)*r*叉树: $\forall v \in V(T)$, $d^+(v) = r$或0
- 完全(complete)树: ∀树叶v,$L(v) = h(T)$
- 有序*r*叉树,有序正则*r*叉树,有序完全正则*r*叉树



定理14.13

- **定理14.13**: 设正则$r(\geq 2)$叉树T有i个分支点和t个树叶, 则$(r-1)i=t-1$.

证明:正则r叉树的每个分枝结点的出度均是r,树叶的出度为零,除了根结点,其余每个结点的入度都为1,根据有向图的握手原理得

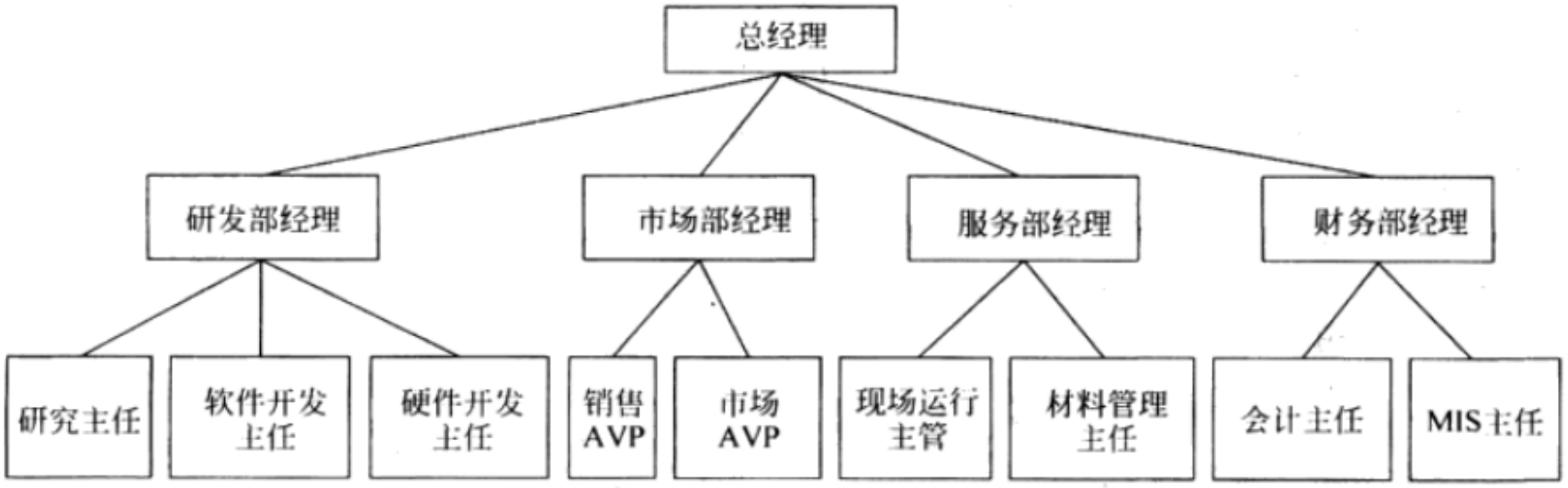
$$m=t+i-1=ri$$

解得 $(r-1)i=t-1$

- 思考：如果树T是r叉树，结论是什么呢？

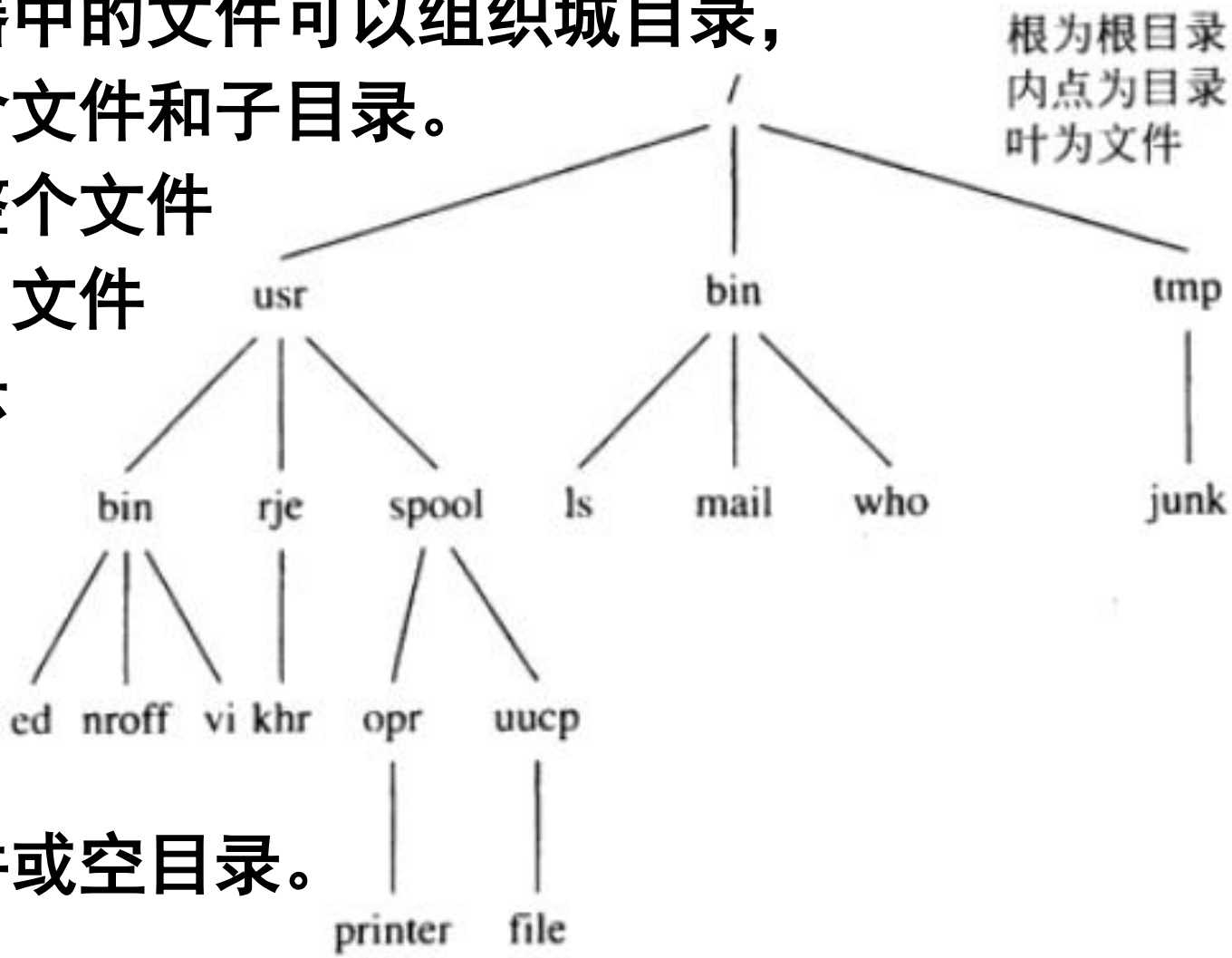
树作为模型—表示组织机构

大的组织机构的结构可以用根树来建模。在这个树中每个顶点表示机构里的一个职务。从一个顶点到另外一个顶点的边的起点所表示的人是终点所表示的人的（直接）上司。如下图所示



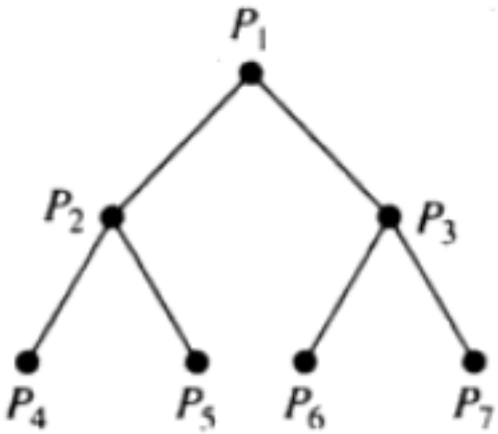
树作为模型—计算机文件系统

计算机存储器中的文件可以组织城目录，目录可以包含文件和子目录。根目录包含整个文件系统。因此，文件系统可以表示成根数，其中根表示根目录，内点表示子目录，树叶表示文件或空目录。



树作为模型—并行处理系统

树形连接网络是把处理器互相连接的一种重要方式，一般采用正则二叉树，能把2^k-1个处理器互相连接起来(k为正整数)。一个非根也非树叶的顶点v所表示的处理器有3个双向连接，根所表示的处理器有2个双向连接，树叶所表示的处理器有1个双向连接。



带 7 个处理器的树形连接网络

如右图中的处理器完成8个数相加的步骤：

步骤1: P_4完成x_1和x_2相加，P_5完成x_3和x_4相加，P_6完成x_5和x_6相加，P_7完成x_7和x_8相加；

步骤2： P_2完成x_1+x_2和x_3+x_4相加，P_3完成x_5+x_6和x_7+x_8相加；

步骤3: P_2完成$x_1+x_2+x_3+x_4$和$x_5+x_6+x_7+x_8$相加

应用实例

例题 设有28盏灯，拟公用一个电源插座，问至少需用多少块具有四插座的接线板。

解 将每个四插座的接线板看作分枝结点，每盏灯看作是树叶，公用的电源插座为根，可以构成4叉树，那么 $t=28$，$r=4$

由$(r-1)i \geq t-1$，可得$i \geq 9$.

所以至少需要9块具有四插座的接线板。

应用实例

例题 假设一台计算机有一条加法指令，可计算3个数的和,如果要计算9个数的和，至少要执行几次加法指令。

解： 把9个数看成正则3叉树的树叶，加法指令则是分枝点,所以$r=3,t=9$, 因此有

$$(3-1)i \geq 9-1$$

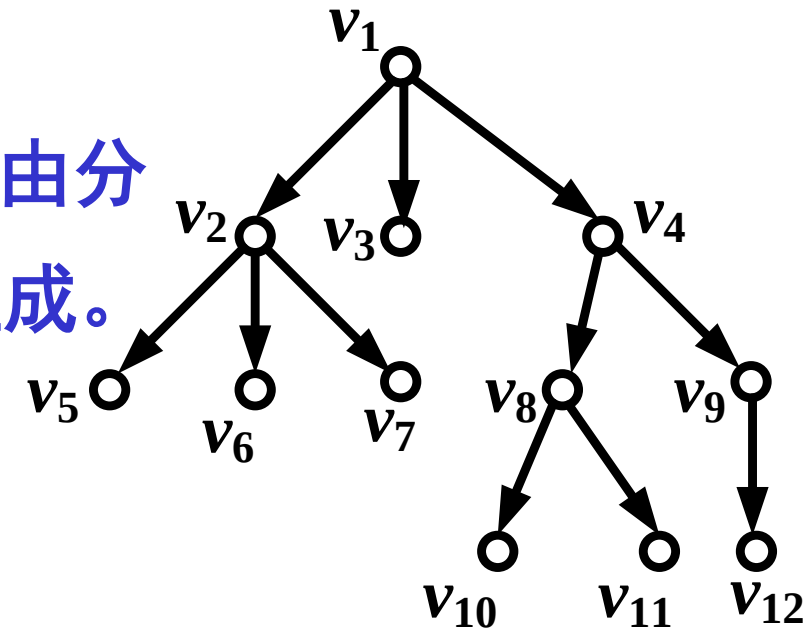
得 $i \geq 4$

故至少需要执行4次加法指令。

根子树

- 根树中除树根v外,其它所有结点被分成有限个子根树.
- 左(右)子树:2叉正则有序树的每个分支点的左右两个儿子导出的根子树.

例 在右图所示的根树中，根树由分别以v_2,v_3,v_4为根的3个子根树组成。以v_4为根的子树由分别以v_8,v_9为根的两个根子树构成。

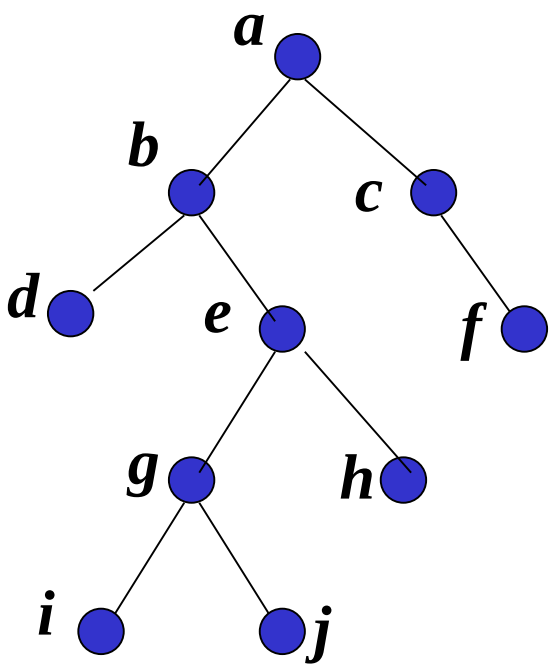


根树的周游(travesal)

- **根树的周游**: 列出根树的所有顶点, 每个顶点恰好出现一次
- **中序行遍**: 左子树, 根, 右子树
- **前序行遍**: 根, 左子树, 右子树
- **后序行遍**: 左子树, 右子树, 根
- **例**: 中序: *dbigjehacf*

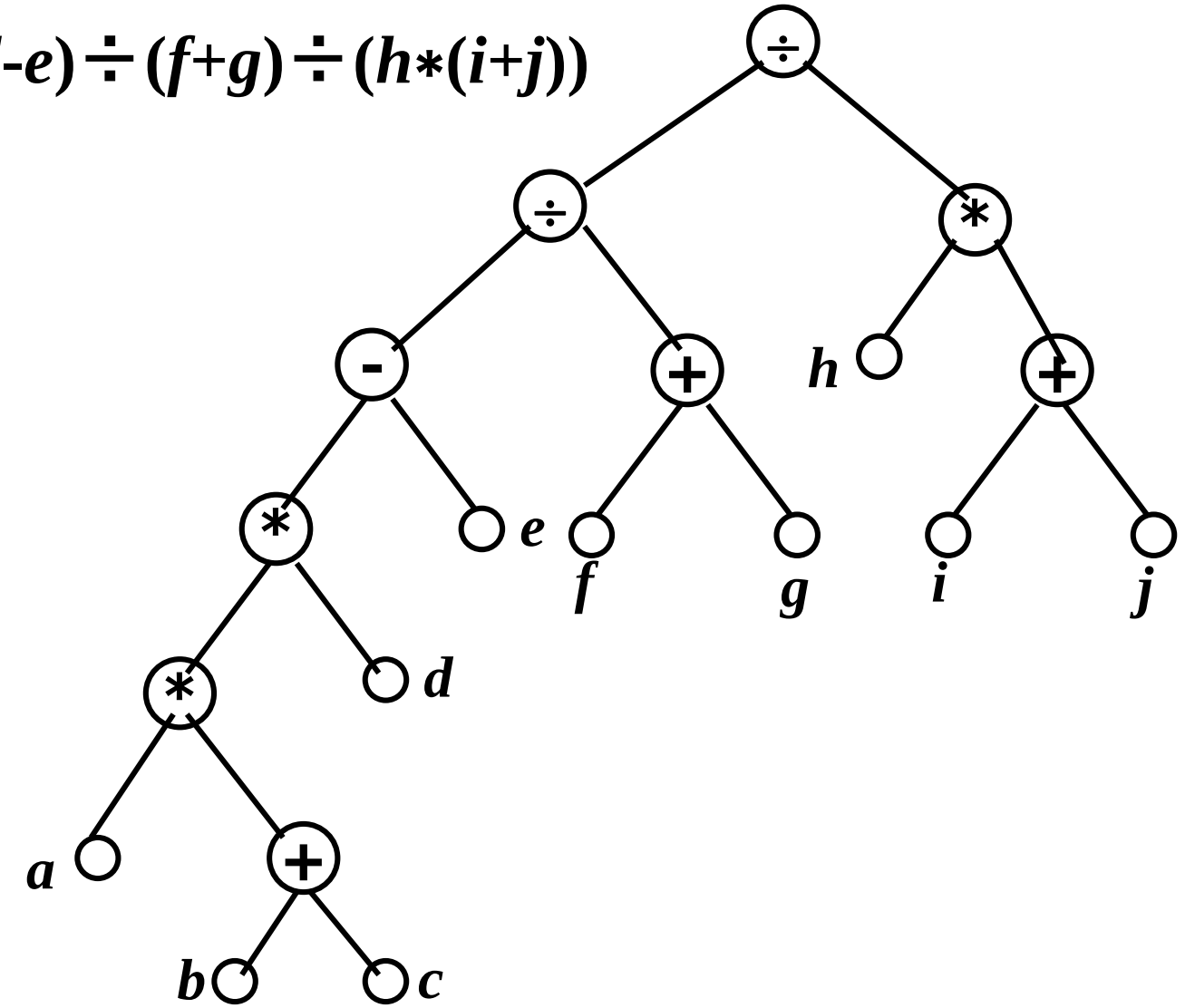
  前序: *abdegijhcf*

  后序: *dijghebfca*



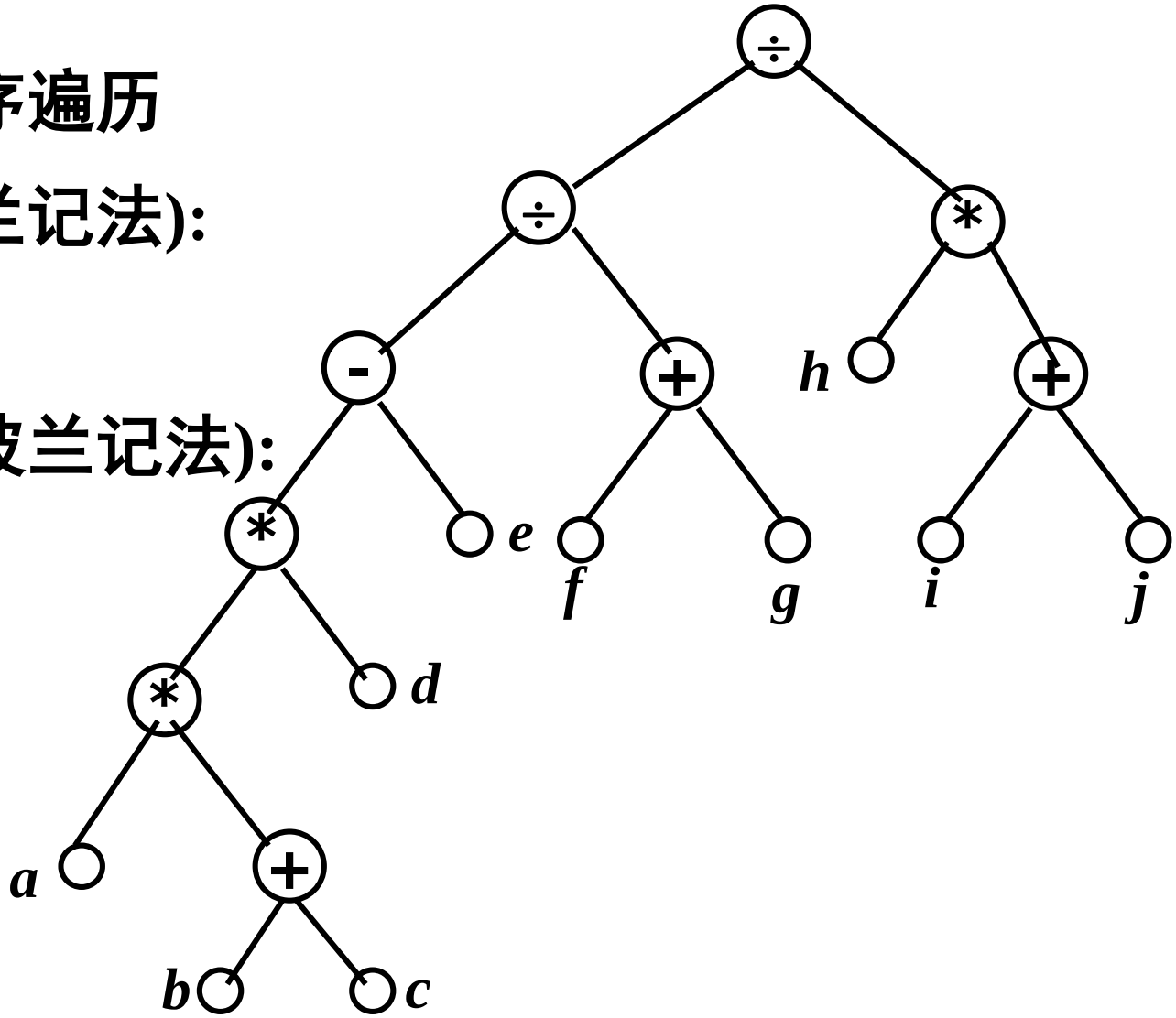
四则运算式与二叉树

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



中缀法,前缀法,后缀法

- 中缀记法:中序遍历
- 前缀记法(波兰记法):
  前序遍历
- 后缀记法(逆波兰记法):
  后序遍历



举例

- 中缀:

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

- 前缀(波兰):

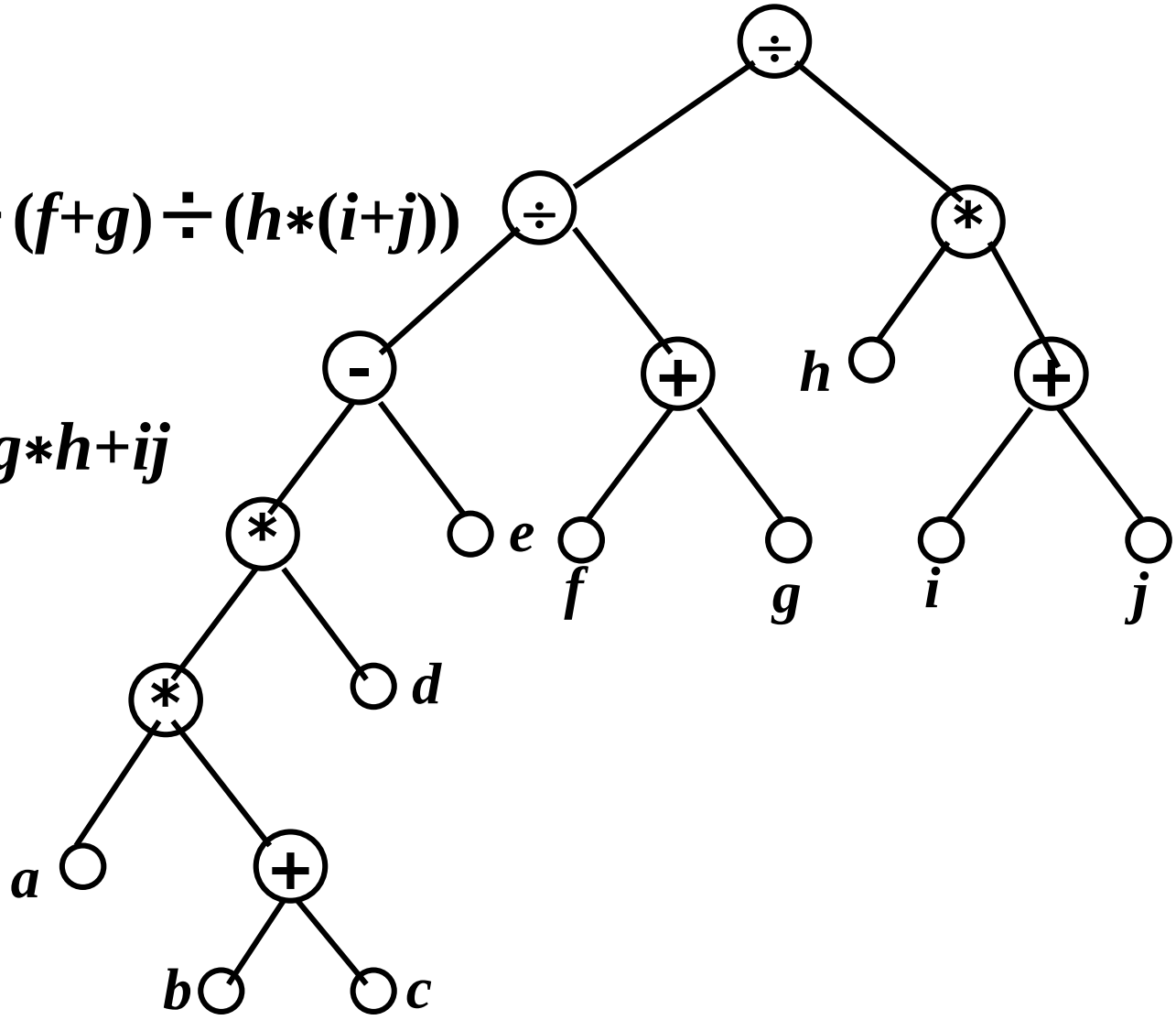
$\div\div-**a+bcde+fg*h+ij$

- 后缀(逆波兰):

$abc+*d*e-fg+\div$

$hij+*\div$

(可以用栈实现）



通讯编码

- Shannon, Hamming, Sudan
- 有噪声信道的可靠通信: 编码
- 信息就是不确定性的消除: 熵(entropy)
- 比特(bit): binary information unit
- 例: {0,1,2,…,7}, $\log_2 8=3$, 编码为000,001,010,…,111

  000111010101译为0725

不等长编码

- 若{0,1,2,…,7}出现频率不一样,则出现频率高的用短码字
- 例: 频率递减: 0,1,2,3,4,5,6,7, 编码为

$0 \to 0,\ 1 \to 1,\quad 2 \to 00,\ 3 \to 01,\ 4 \to 10,$

$5 \to 11,\ 6 \to 000,\ 7 \to 001.$

若收到000111, 不能唯一解码:

651, 235, 075,…等.

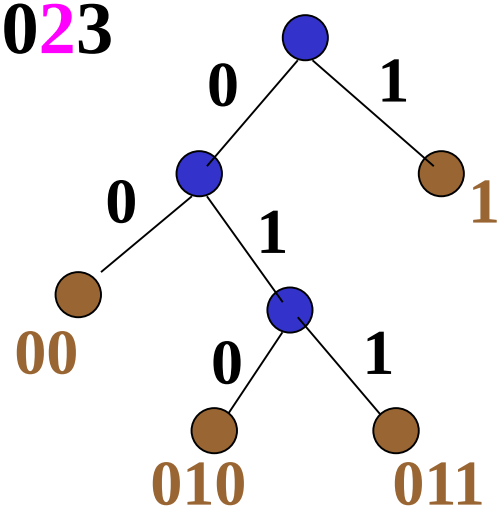
原因: 有码字是其它码字的前缀,如00是001的前缀

前缀码(prefix code)

- **前缀码**: 码字互相不为前缀的不等长编码。
- 前缀码与二叉树对应
- **例**:{0,1,2,3}编码为{00,010,011,1},

  即0→00, 1→010, 2→011, 3→1

  收到00011 1,译为023



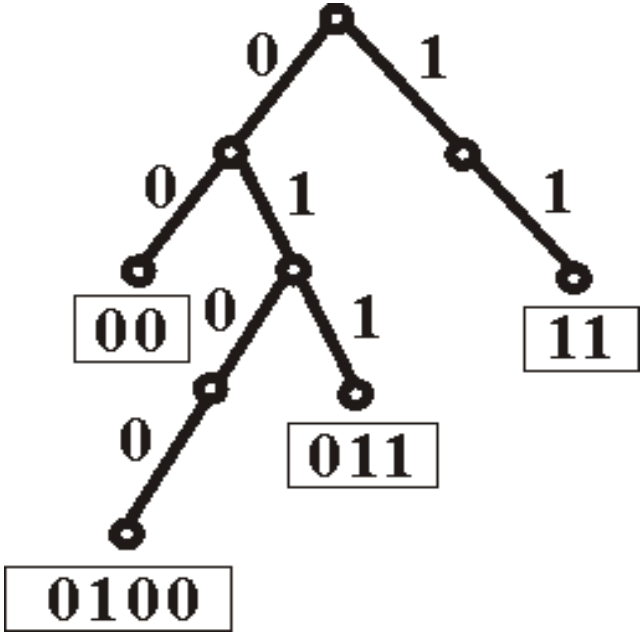
前缀码的得到

定理 任意一棵二叉树的树叶可对应一个前缀码.

证明 给定一棵二叉树,对每个分支点,若关联2条边, 则给左边标0,右边标1; 若只关联1条边,则可以给它标0(看作左边),也可以标1(看作右边). 将从树根到每一片树叶的通路上标的数字组成的字符串标定在树叶处.因为没有一片树叶的标定字符串是另一片树叶标定字符串的前缀,因此任何一棵二叉树的树叶对应一个前缀码.

实例

如下图所示。



前缀码对应二叉树

定理 任何一个前缀码都对应一棵二叉树.

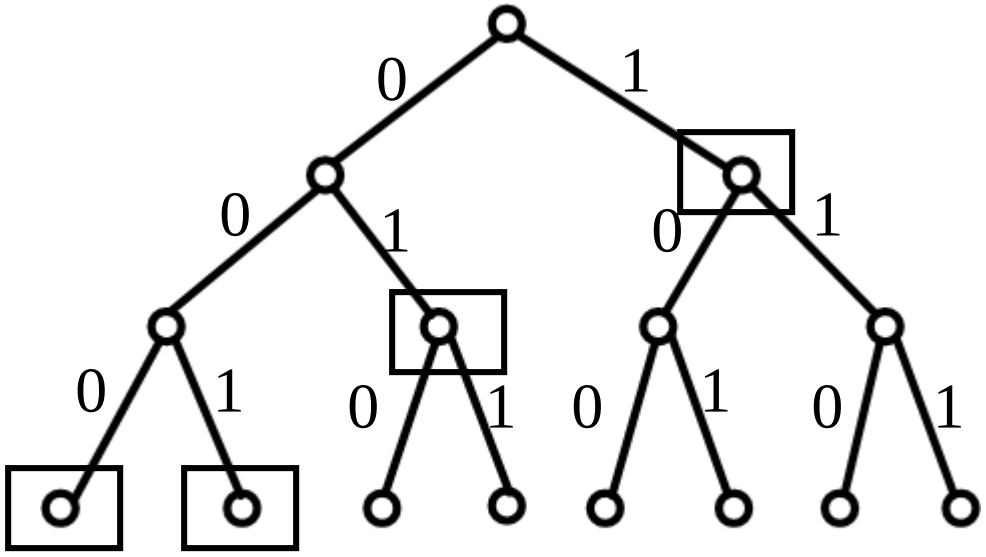
证明 设给定一个前缀码,前缀码中最长序列的长度为h.

画出一棵高度为h的完全正则二叉树,给每个分枝点射出的两条边分别标以0和1,这样树中的每个结点可以标定一个长度不超过h的二进制序列,它是从树根到该结点通路上各边的标号所确定,因此长度不超过h的每一个二进制序列必对应一个结点.

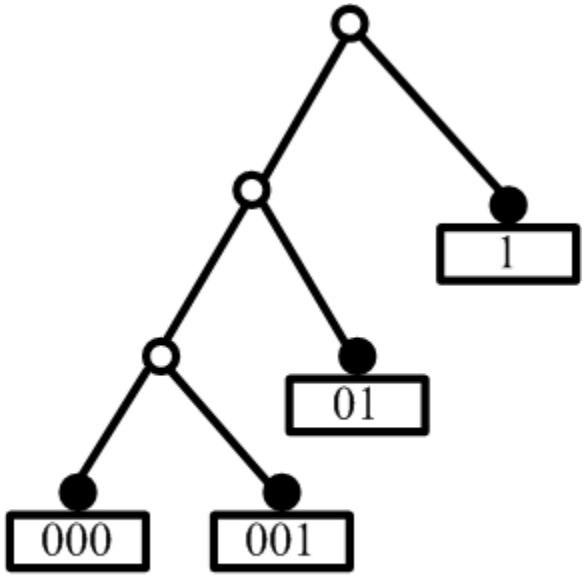
对应于前缀码中的每一序列的结点给予一个标记,并将标记结点的所有后裔和射出的边全部删掉,再删掉其中未加标记的树叶,得到一棵新的二叉树,它的树叶就对应给定的前缀码.

实例

画出与前缀码{000,001,01,1}对应的二叉树。



(a)



(b)

最佳前缀码

- **最佳前缀码:** 给定信号出现频率, 平均码字长度最短的前缀码
- **平均码字长度:** 码字长度乘以频率,求和
- **例**: {0,1,2,3}, 40%, 30%, 20%,10%,

  编码1: {00,010,011,1},

  $2\times40\%+3\times30\%+3\times20\%+1\times10\%=2.4$

  编码2: {1,00,010,011},

  $1\times40\%+2\times30\%+3\times20\%+3\times10\%=1.9$

最优二叉树

- **带权二叉树:** 每个树叶v_i都指定实数权w_i
- **带权二叉树的权:** $W(T)=\Sigma^{t}_{i=1}w_iL(v_i)$, 树叶是$v_1, v_2,\ldots, v_t$, 对应的层数是$L(v_1),L(v_2),\ldots,L(v_t)$
- **最优二叉树:** 树叶权为$w_1,w_2,\ldots,w_t$的所有二叉树中权最小的一个(不唯一)
- 求最优二叉树的算法: Huffman算法

Huffman算法

- 输入: 实数$w_1,w_2,\ldots,w_t$
- 输出: 树叶权为$w_1,w_2,\ldots,w_t$的最优二叉树
- 算法:

  1. 选择最小的2个权w_1,w_2, 连接对应的树叶得到权为w_1+w_2的分支点

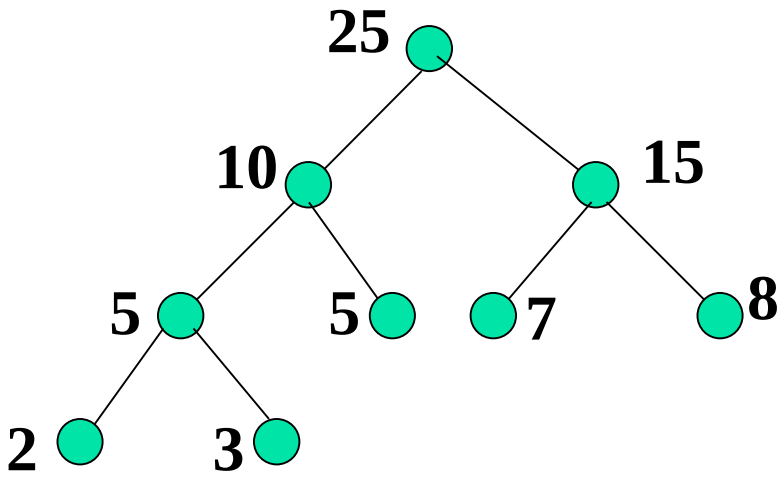
  2. 选择w_1+w_2 ,w_3,w_4, …,w_t中最小的2个权, 连接对应顶点得到新的分支点和权

  3. 同上重复进行, 直到只剩1个权为止

Huffman算法(举例)

例14.9: 2,3,5,7,8

解: 2,3,5,7,8 // 5,5,7,8 // 10,7,8 // 10,15 // 25 //



W(T)=2*3+3*3+5*2+7*2+8*2＝55.

Huffman算法(正确性)

定理14.11 设T为带权$w_1 \le w_2 \le \ldots \le w_t$的最优树,则

a)带权w_1, w_2的树叶v_{w_1}, v_{w_2}是兄弟.

b)以树叶v_{w_1}, v_{w_2}为儿子的分枝点,其通路长度最长.

证明 设在带权$w_1, w_2, \ldots, w_t$的最优树中,v是通路长度最长的分枝点,v的儿子分别带权w_x和w_y,有

$$L(v_{w_x}) \ge L(v_{w_1}) \qquad L(v_{w_y}) \ge L(v_{w_2})$$

若$L(v_{w_x}) > L(v_{w_1})$,将w_x与w_1对调,得到新树T',则

定理14.11的证明

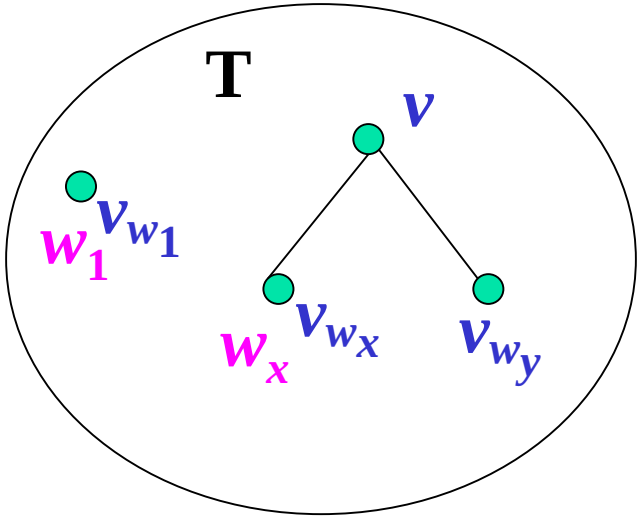
$w(T')-w(T)=(L(v_{w_x})\,w_1+L(v_{w_1})\,w_x)-(L(v_{w_x})w_x+L(v_{w_1})w_1)$

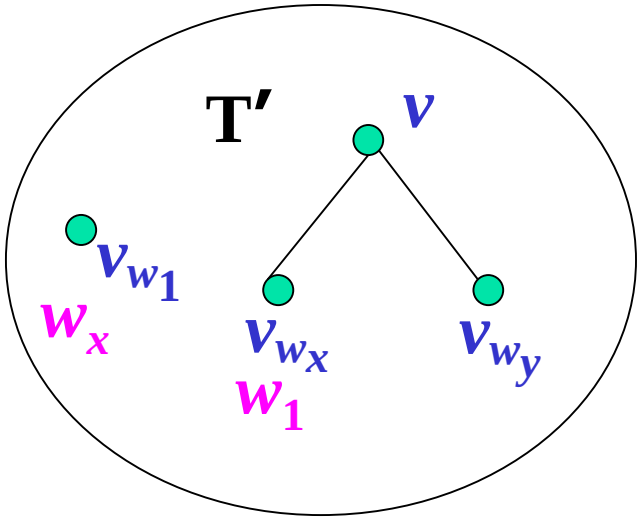
$=L(v_{w_x})(w_1-w_x)-L(v_{w_1})(w_1-w_x)$

$=(L(v_{w_x})-L(v_{w_1}))\,(w_1-w_x)<0$

那么$w(T')<w(T)$，这与T是最优树矛盾.

故$L(w_x)=L(w_1)$.





定理14.11的证明

同理可证 $L(w_x) = L(w_2)$.

因此$L(w_x) = L(w_1) = L(w_y) = L(w_2)$

分别将w_1, w_2与w_x, w_y对调得到一棵最优树，

其中带权w_1, w_2的树叶是兄弟。

定理14.12

定理14.12 设T为带权$w_1 \le w_2 \le \ldots \le w_t$的最优树,若将以带权$w_1, w_2$的树叶为儿子的分枝点改为带权$w_1+w_2$的树叶,得到一棵新树$T'$,则$T'$也是最优树.

证明 (反证法)根据题设有

$w(T)-w(T')= L(w_1)\, w_1+ L(w_2)\, w_2- (L(w_1)-1)\, (w_1+w_2)$

因为$L(w_1) = L(w_2)$,所以 $w(T)-w(T')= w_1+w_2$ (1)

若T'不是最优树,则必有另一棵带权$w_1+w_2, w_3, \ldots, w_t$的最优树$T''$. 对于$T''$中带权$w_1+w_2$的树叶生成两个儿子,得到新树$\hat{T}$,则$w(\hat{T})=w(T'')+ w_1+w_2$ (2)

定理14.12的证明

因为T''是带权$w_1+w_2, w_3, \ldots, w_t$的最优树，

故$w(T'') \le w(T')$.

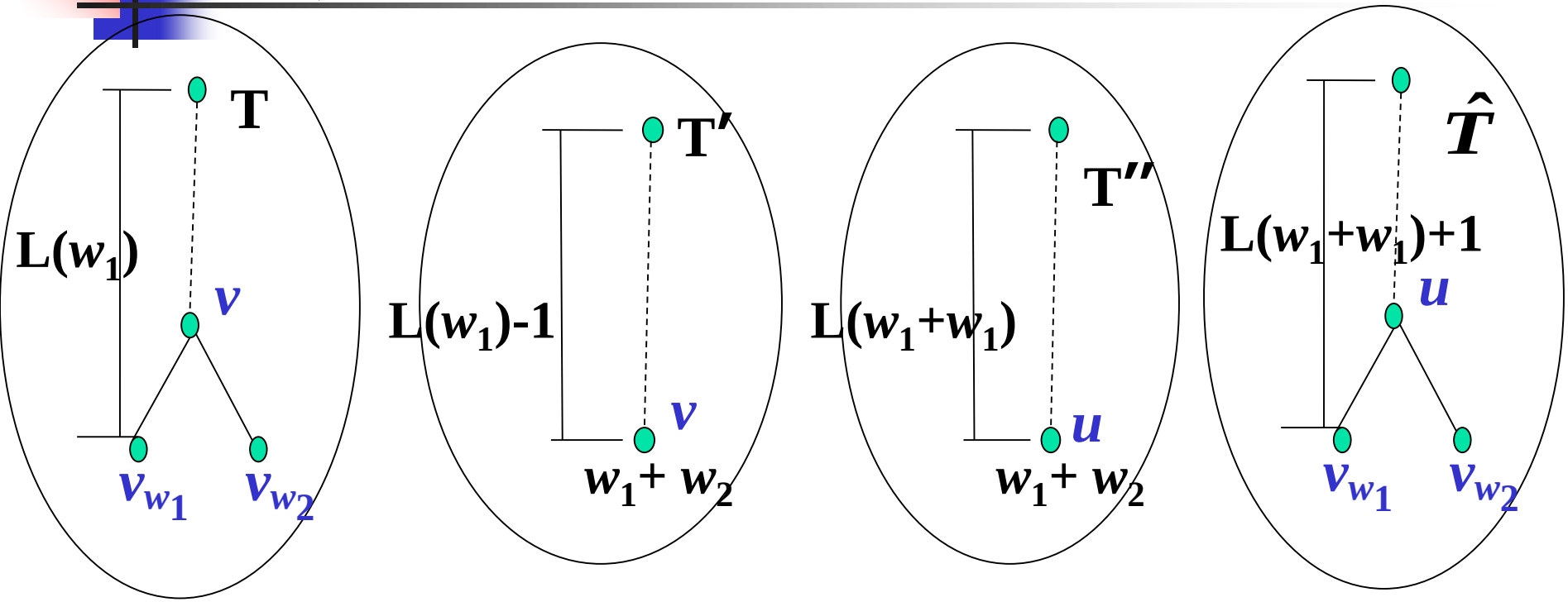
如果$w(T'') < w(T')$，由(1)(2)式得 $w(\hat{T}) < w(T)$

这与T是带权 $w_1, w_2, \ldots, w_t$的最优树相矛盾，

所以$w(T'') = w(T')$，

T'是带权$w_1+w_2, w_3, \ldots, w_t$的最优树。

证明的图例



带权$w_1 \le w_2 \le \ldots \le w_t$
的最优树T

带权$w_1+w_2,\ldots,w_t$
的最优树T″

$W(T)=W(T')+w_1+w_2$　　$W(\hat{T})=W(T'')+w_1+w_2$

$W(T'')<W(T')$,所以$W(\hat{T})<W(T)$,这与T是最优树矛盾。

实例

例 在通信中，设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%

采用二元前缀码, 求传输数字最少的二元前缀码, 并求传输$10^n(n\geq 2)$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?

解 用Huffman算法求以频率(乘以100)为权的最优2叉树. 这里 $w_1=5, w_2=5, w_3=10, w_4=10, w_5=10, w_6=15, w_7=20, w_8=25$. 最优2叉树如图所示.

实例（续）

编码： 0---01，1---11，2---001，3---100

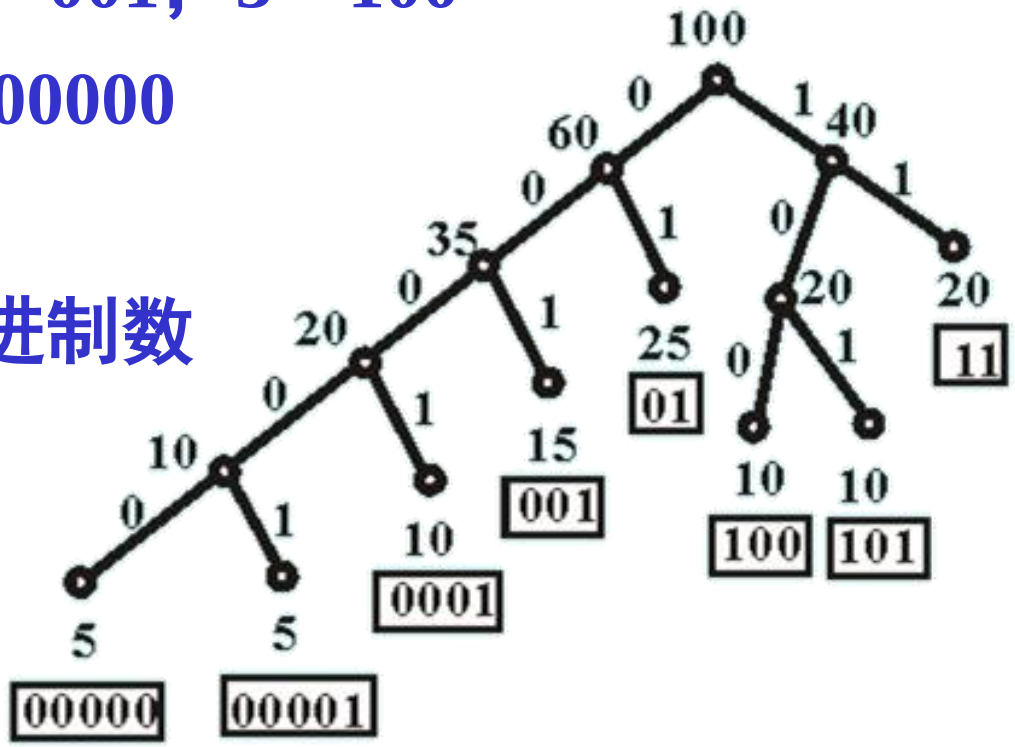
4---101，5---0001，6---00000

7---00001

传100个按比例出现的八进制数字所需二进制数字的个数为 $w(T)=285$.

传$10^n(n\geq 2)$个所用二进制数字的个数为2.85×10^n,

而用等长码(长为3)需要用 3×10^n个数字.



总结

- 根树
- 根树的周游
- 最优树, Huffman算法
- 最佳前缀码
- 作业:p155, 习题九 20,21；p220, 习题十四, 12

补充题：带有*i*个分支点的正则*r*叉树含有多个顶点？