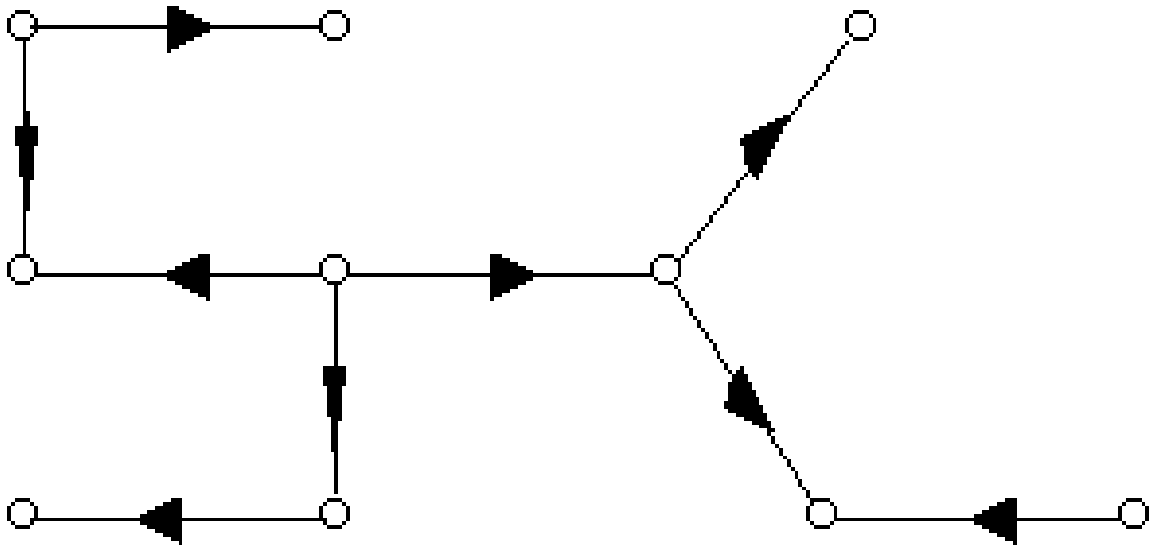
7-8 根树及其应用

有向树

前面我们讨论的树，都是一个无向图，下面我们讨论有向图的树。

定义7-8.1 如果一个有向图在不考虑边的方向时是一棵数，那么，这个有向图称为有向树。

有向树



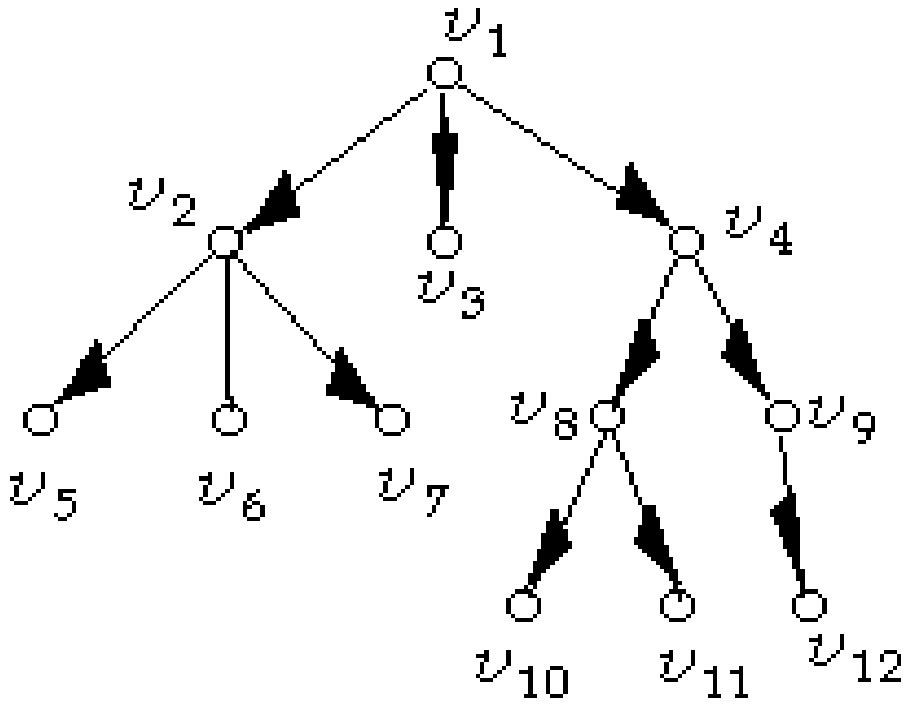
(a)

根树

- 一棵有向树，如果恰有一个结点的入度为0，其余所有结点的入度都为1，则称为根树。
- 入度为0的结点称为根。
- 出度为0的结点称为叶。
- 出度不为0的结点称为分枝点或内点。

根树

例如图表示的是一棵根树，其中**v1**为根，**v1**，**v2**，**v4**，**v8**，**v9**为分枝点，其余结点为叶。

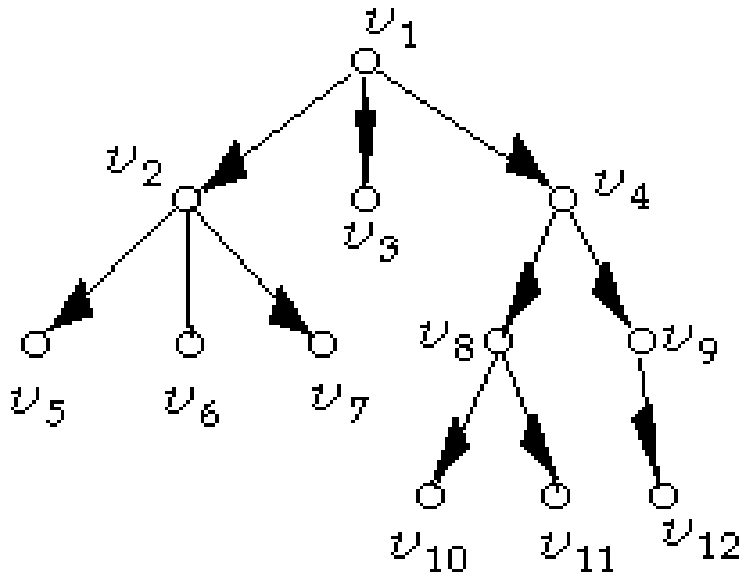


(b)

结点的层次

在根树中，任意一个结点*v*的层次，就是从根到该结点的单向通路的长度。

在下图中，有三个结点的层次为1，有五个结点的层次为2，有三个结点的层次为3。



根树

从根树的结构可以看出，树中每一个结点可以看作是原来树中的某一棵子树的根，由此可知，根树亦可递归定义为

定义7-8.3 根树包括一个或多个结点，这些结点中某一个称为根，其它所有结点被分成有限个子根树。

这个定义把*n*个结点的根树用结点数少于*n*的根树来定义，最后得到每一棵都是一个结点的根树，它就是原来那棵树的树叶。

根树

对于一棵根树，如果用图形来表示，可以有树根在下或树根在上的两种画法。

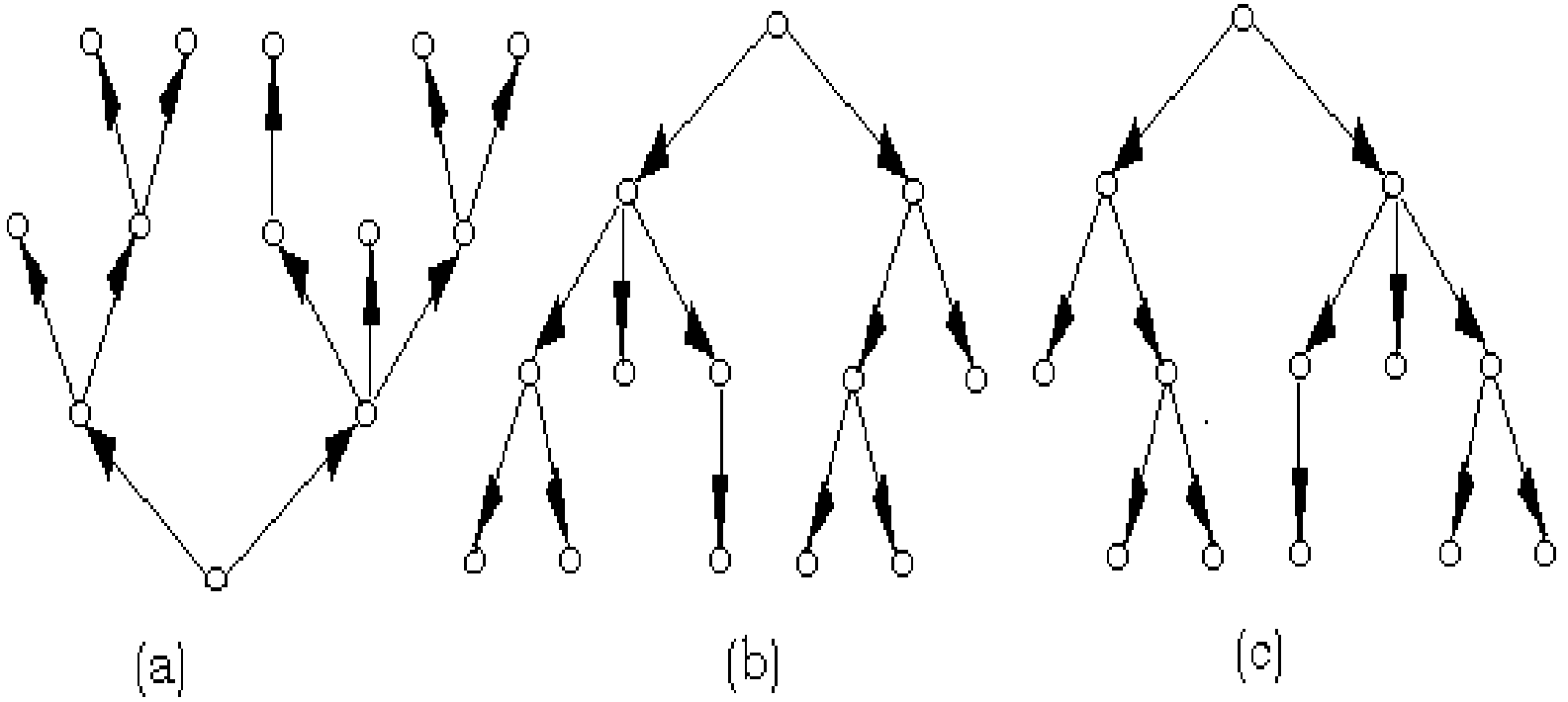


图7-8.2 根树的两种画法

有序树

图7-8.2(a)是根树自然表示法，即树从它的树根向上生长。图7-8.2(b)和(c)都是由树根往下生长，它们是同构图，其差别仅在于每一层上的结点从左到右出现的次序不同，为此今后要用明确的方式，指明根树中的结点或边的次序，这种树称为有序树。

根树（族谱）

设*a*是根树中的一个分枝点，若从*a*到*b*有一条边，则结点*a*称为结点*b*的“父亲”，而结点*b*称为结点*a*的“儿子”。

假若从*a到c有一条单向通路*，则结点*a*称为结点*c*的“祖先”，而结点*c*称为结点*a*的“后裔”。

同一个分枝点的“儿子”称为“兄弟”。

m叉树

m叉树是一种特殊的根树，在***m*=2**时，称为二叉树，它在计算机科学中有着广泛的应用。

定义7-8.4

在根树中，若每一个结点的出度小于或等于***m***，则这棵树称为***m***叉树。

若***m*=2**时，称为二叉树。

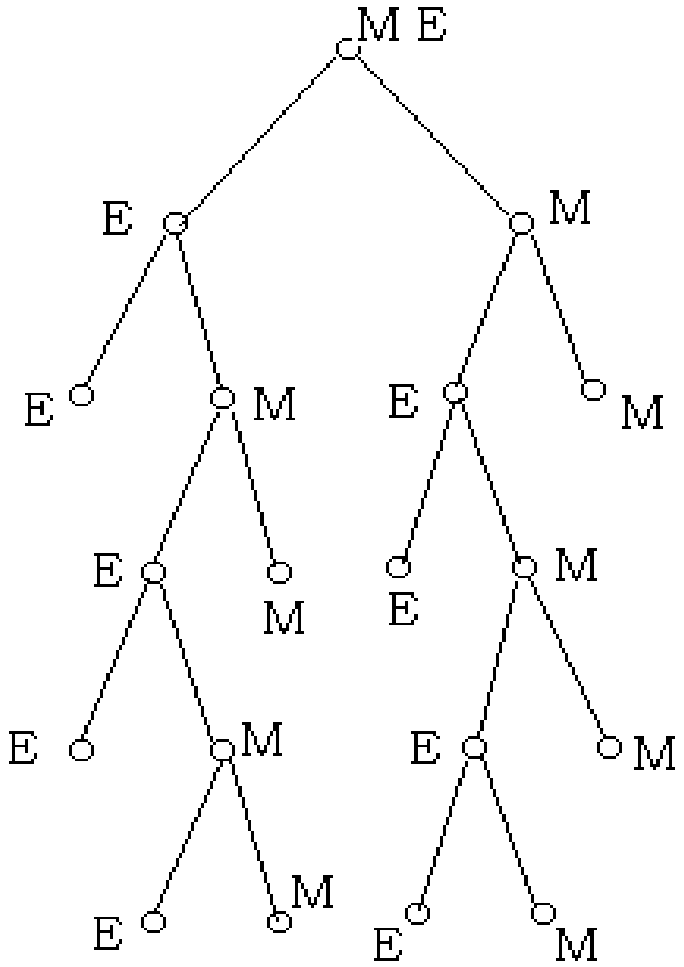
如果每一个结点的出度恰好等于***m***或零，则这棵树称为完全***m***叉树。

若所有的树叶层次相同，则这棵树称为正则***m***叉树。

有许多实际问题可用二叉树或***m***叉树来表示。

m叉树

例如M和*E*两人进行网球比赛，如果一人连胜两盘或共胜三盘就获胜，比赛结束。图7-8.3表示了比赛可能进行的各种情况，它有十片树叶，从根到树叶的每一条路对应比赛可能发生的一种情况，即：MM，M*E*MM，M*E*M*E*M，M*E*M*EE*，M*EE*，*E*MM，*E*M*E*MM，*E*M*E*M*E*，*E*M*EE*，*EE*。



完全m叉树定理

在树的实际应用中，我们经常研究完全*m*叉树。

定理7-8.1 设有完全*m*叉树，其树叶数为*t*，分枝点数为*i*，则$(m-1)i=t-1$。

证明 由假设知，该树有 $i+t$ 个结点，

由树的定义e=v-1可知，该树边数为 $i+t-1$ 。

因为所有结点出度之和等于边数所以根据完全*m*叉树的定义知，$mi = i+t-1$

即 $(m-1)i = t-1$

举例

例1. 设有28盏灯，拟共用一个电源插座，问需用多少块具有四插座的接线板。

解 将四叉树每个分枝点看作是具有四插座的接线板，树叶看作电灯，则$(4-1)i=28-1$，$i=9$，所以需要九块具有四插座的接线板。

举例

例2 假设有一台计算机， 它有一条加法指令， 可计算 3 个数之和。 如果要求 9 个数x_1， x_2， …， x_9之和， 问至少要执行几次加法指令？

解: 用 3 个结点表示 3 个数， 把 9 个数看成树叶， 将表示 3 数之和的结点作为它们的父亲结点。 这样本问题可理解为求一个完全三叉树的分枝点的个数问题。

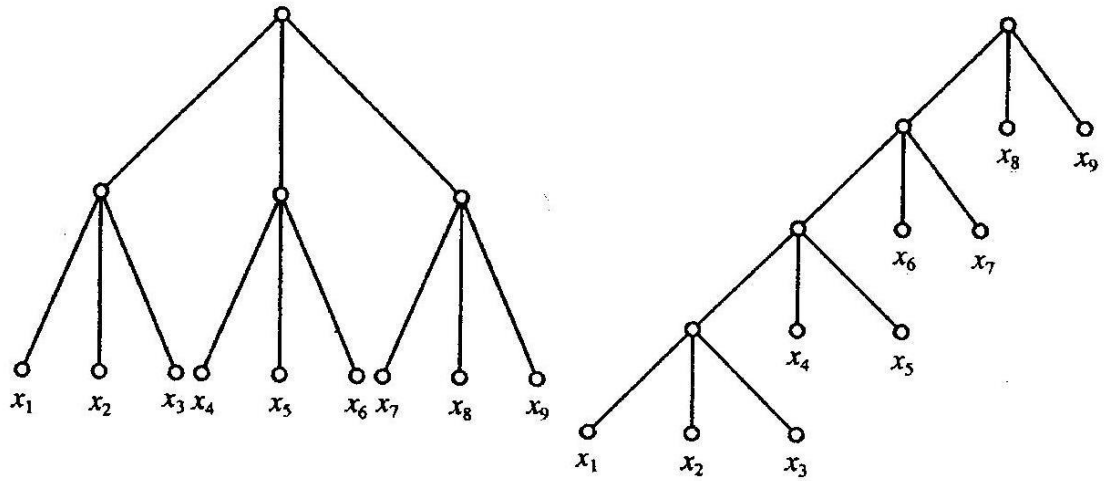
举例

由定理7.6.1知， 有 $(3-1)i = 9-1$

得 $i = 4$

所以要执行 4 次加法指令。

图7.6.5表示了两种可能的顺序。

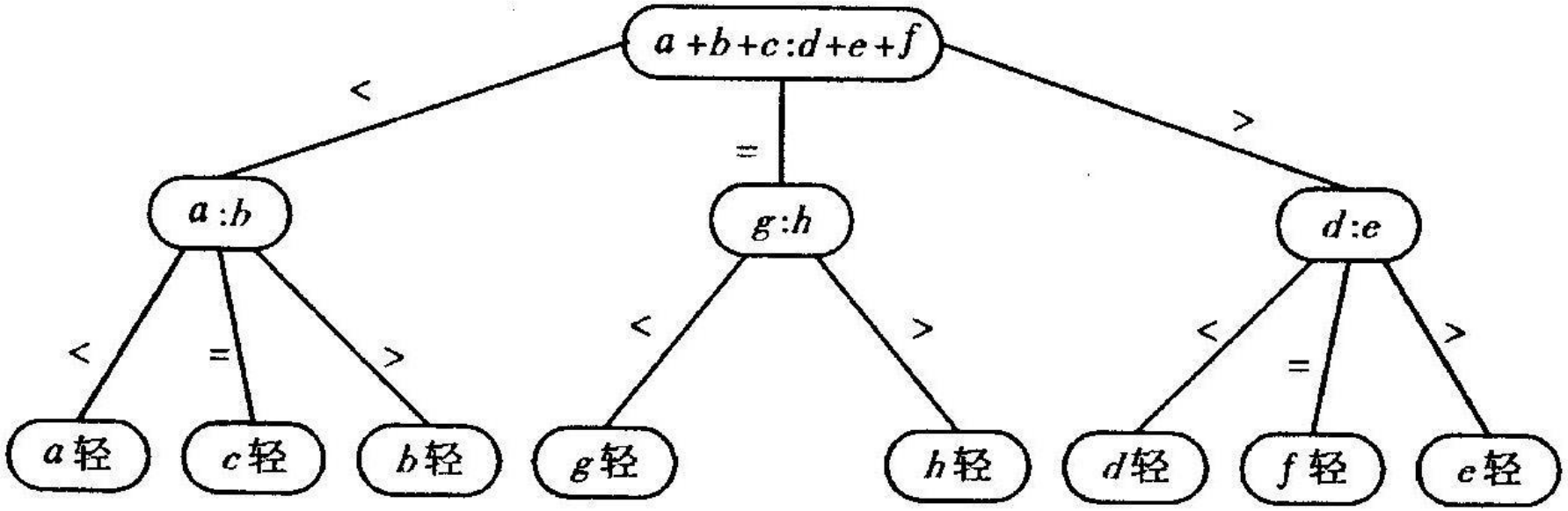


举例

例3 8 枚硬币问题。 若有 8 枚硬币*a*， *b*， *c*， *d*， *e*， *f*， *g*， *h*， 其中 7 枚重量相等， 只有 1 枚稍轻。 现要求以天平为工具， 用最少的比较次数挑出轻币来。

举例

解: 可用图所示的树表示判断过程。



从图中可知， 只需称 2 次即可挑出轻币。 这种用于描述判断过程的树叫判定树。

最优树

二叉树的一个重要应用就是最优树问题。

给定一组权w_1，w_2，…，w_t，不妨设$w_1 \le w_2 \le \ldots \le w_t$。设有一棵二叉树，共有$t$片树叶，分别带权$w_1$，$w_2$，…，$w_t$，该二叉树称为带权二叉树。

定义7-8.6 在带权二叉树中，若带权为w_i的树叶，其通路长度为$L(w_i)$，我们把$w(T)=\sum_{i=1}^{t}\omega_i L(\omega_i)$称为该带权二叉树的权。在所有带权$w_1$，$w_2$，…，$w_t$的二叉树中，找到一棵使$w(T)$最小的那棵树，称为最优树。

最优树

假若给定一组权w_1，w_2，…，w_t，为了找最优树，我们先证明下面定理：

定理7-8.3 设T为带权$w_1 \le w_2 \le \ldots \le w_t$的最优树，则

⑴带权为w_1，w_2的树叶是兄弟。

⑵以树叶V_{w_1}，V_{w_2}为儿子的分枝点，其通路长度最长。

最优树

证明　设在带权w_1，w_2，…，w_t的最优树中，v是通路长度最长的分枝点，v的儿子分别为w_x和w_y，故有

$L(w_x)\geq L(w_1)$

$L(w_y)\geq L(w_2)$

若有$L(w_x)>L(w_1)$，将w_x与w_1对调，得到新树T'，则

$w(T')-w(T)=(L(w_x)\cdot w_1+L(w_1)\cdot w_x)-(L(w_x)\cdot w_x+L(w_1)\cdot w_1)$

$=L(w_x)(w_1-w_x)+L(w_1)(w_x-w_1)=(w_x-w_1)(L(w_1)-L(w_x))<0$。

即$w(T')<w(T)$与T是最优树的假定矛盾。故$L(w_x)=L(w_1)$。

同理可证$L(w_x)=L(w_2)$。因此

$L(w_1)=L(w_2)=L(w_x)=L(w_y)$

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和w_2的树叶是兄弟。

最优树

定理7-8.4 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。

最优树

证明 根据题设，有

$w(T)=w(T')+w_1+w_2$。

若T'不是最优树，则必有另外一棵带权为w_1+w_2，w_3，…，w_t的最优树T''。对T''中带权为w_1+w_2的树叶v_{w1+w2}生成两个儿子，得到树 $\hat{T}$ ，则

$w(\hat{T})=w(T'')+w_1+w_2$。

因为T''是带权为w_1+w_2，w_3，…，w_t的最优树树，故

$w(T'')\leq w(T')$。

如果$w(T'')<w(T')$，则$w(\hat{T})<w(T)$，与T是带权为w_1，w_2，…，w_t最优树矛盾，因此

$w(T'')=w(T')$

T'是一棵带权为w_1+w_2，w_3，…，w_t的最优树。

Huffman算法

根据上面两个定理，要画一棵带有t个权的最优树，可简化为画一棵带有t-1个权的最优树，而又可简化为画一棵带t-2个权的最优树，依此类推。具体的做法是：首先找出两个最小w值，设为w_1和w_2，然后对t-1个权w_1+w_2，w_3，…，w_t求作一棵最优树，并且将这棵最优树的结点v_{w1+w2}分叉生成两个儿子v_1和v_2，依此类推。此称为**Huffman算法。**

Huffman算法

例3.　设一组权2，3，5，7，11，13，17，19，23，29，31，37，41。求相应的最优树。

解：见P334

Huffman算法

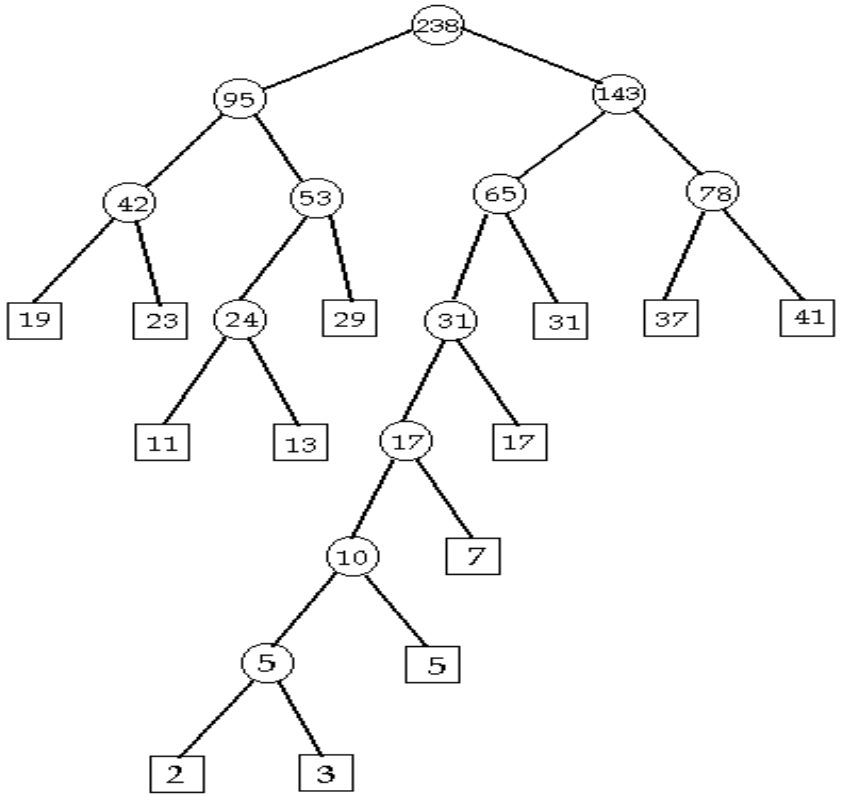


图7-8.7 最优二叉树

前缀码问题

二叉树的另一个应用，就是前缀码问题。

我们知道，在远距离通讯中，常常用0和1的字符串作为英文字母传送信息，因为英文字母共有26个，故用不等长的二进制序列表示26个英文字母时由于长度为1的序列有2个，长度为2的二进制序列有2^2个，长度为3的二进制序列有2^3个，依此类推，我们有

$2+2^2+2^3+\ldots+2^i \geq 26$

$2^{i+1}-2 \geq 26$，$i \geq 4$

前缀码问题

因此，用长度不超过四的二进制序列就可表达26个不同英文字母。但是由于字母使用的频繁程度不同，为了减少信息量，人们希望用较短的序列表示频繁使用的字母。

当使用不同长度的序列表示字母时，我们要考虑的另一个问题是如何对接收的字符串进行译码？

例如，假设

A:0,B:01,C:001

问：二进制序列**001001**如何译码？

CC? ABC? ABAB? CAB?

前缀码问题

定义7-8.7　给定一个序列集合，若没有一个序列是另一个序列的前缀，该序列集合称为前缀码。

例如{000，001，01，10}是前缀码，

而{1，0001，000}就不是前缀码。

前缀码问题

定理7-8.5 任何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，从每一个分枝点引出两条边，对左侧边标以0，对右侧边标以1，则每片树叶可以标定一个0和1的序列，它是由树根到这片树叶的通路上各边标号所组成的序列，显然，没有一片树叶的标定序列是另一片树叶的标定序列的前缀，因此，任何一棵二叉树的树叶可对应一个前缀码。

前缀码问题

定理7-8.6 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，h表示前缀码中最长序列的长度。我们画出一棵高度为h的正则二叉树，并给每一分枝点射出的两条边标以0和1，这样，每个结点可以标定一个二进制序列，它是从树根到该结点通路上各边的标号所确定，因此，对长度不超过h的每一二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去，这样得到一棵二叉树，再删去其中未标记的树叶，得到一棵新的二叉树，它的树叶就对应给定的前缀码。

前缀码问题

前缀码{000，001，01, 1}

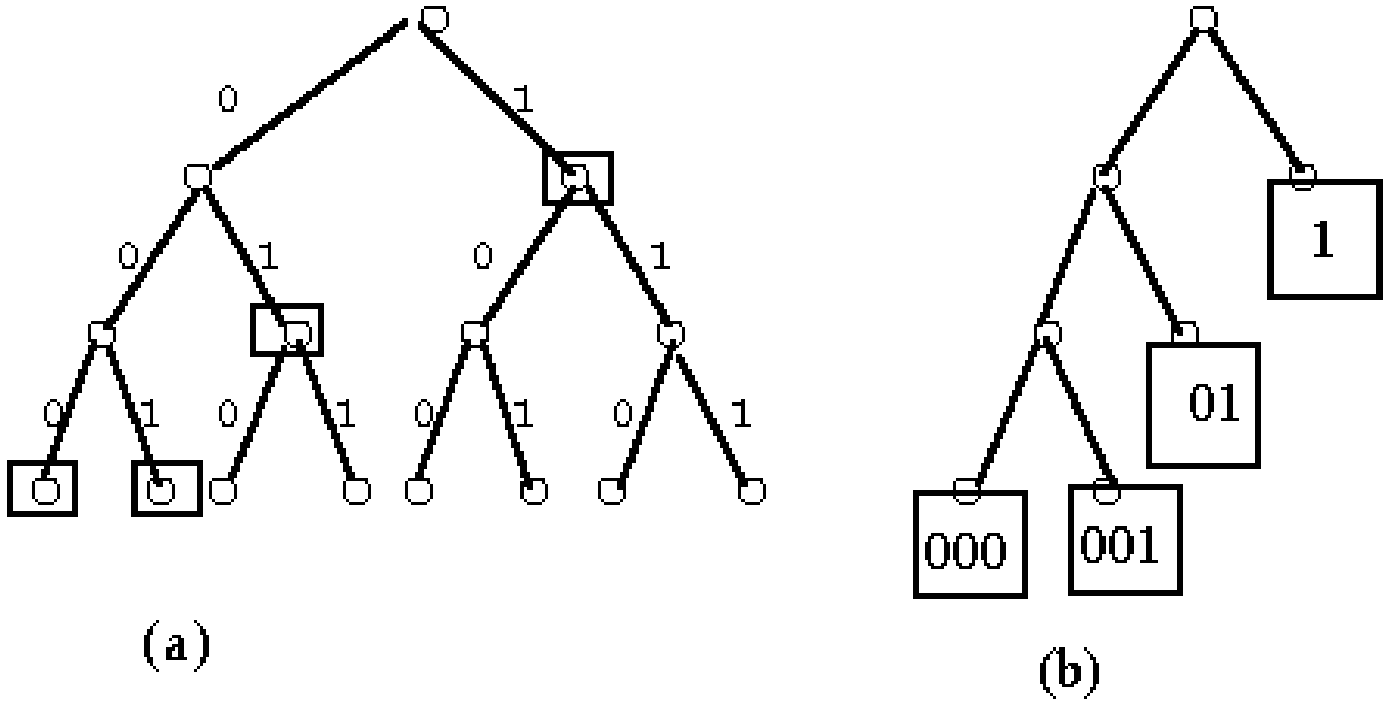


图7-8.8 二叉树对应前缀码

前缀码问题

例如图7-8.8(b)中所对应的前缀码{000,001，01，1}，可对任意二进制序列进行译码。

设有二进制序列

000100110111101001

可译为000,1,001,1,01,1，1,01,001。

如果被译的信息最后部分不能译前缀码中的序列，可约定添加0或1，直至能够译出为止。

练习

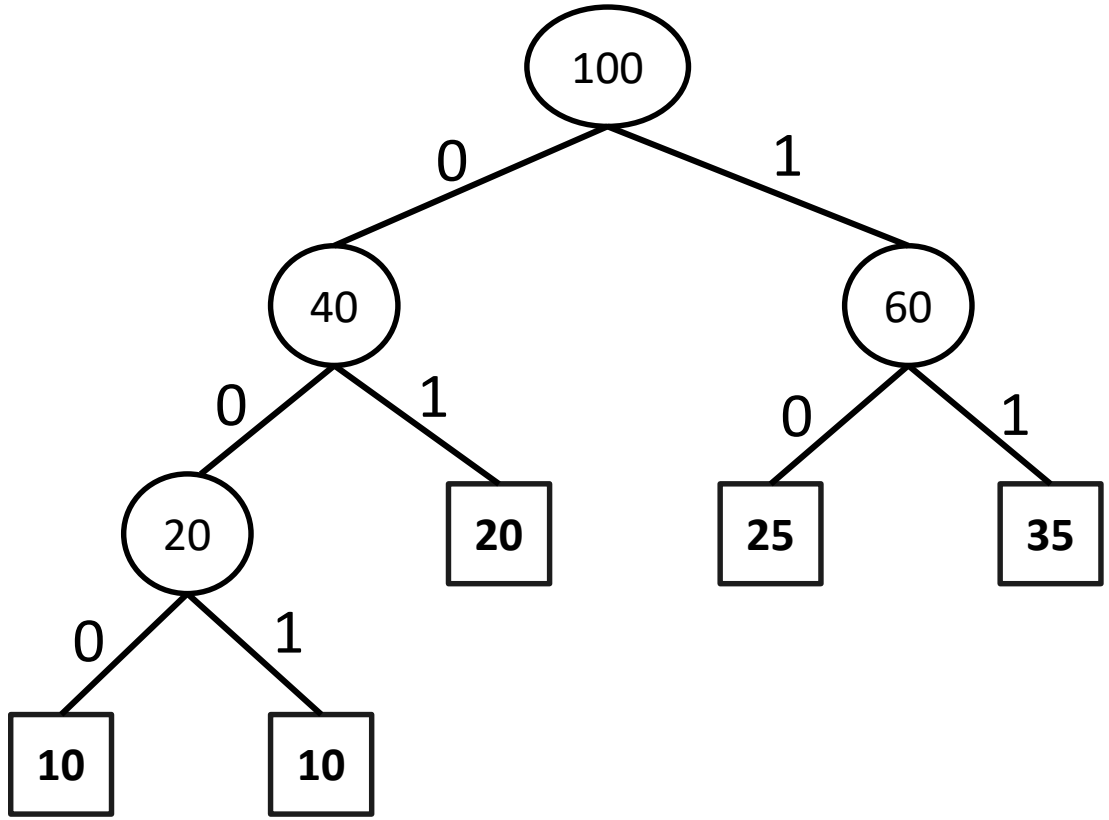
已知字母A,B,C,D,E,F出现频率如下：

A：35%,B：25%,C：20%,D：10%,E：5%, F：5%

如何选择前缀码，使得输入100个这样的字母所用的二进制位尽可能的少？并给出6个字母的最佳编码。

练习

A：35次, B：25次, C：20次, D：10次, E：10次



谢谢